Mathematical Modelling Project Report

Simulation of Wave Equations in 2D

Teacher: M. Nyffeler

Matteo Delucchi, ACLS, 2. Semester, *delucmat@students.zhaw.ch*

Peer-reviewed by *Eldhose Poulose*

Zürich, January 10, 2019

Matteo Delucchi

1. Finite Difference Methods for 2D wave equations

A wide range of physical processes lead to wave motion. Signals are then propagated through a medium in space and time with normally little or no permanent movement of the medium itself. Many types of wave motion can be described by the hyperbolic partial differential equation

$$u_{tt} = \nabla \cdot (c^2 \nabla u) + f. \tag{1}$$

where t is the time variable, ∇^2 is the spatial Laplacian, c a fixed constant and $u = u(x_1, x_2, ..., x_n; t)$ the scalar function modelling for example the displacement of the wave with the spacial variables $x_1, x_2, ..., x_n$. Waves in a one-dimensional space can be ~~used to describe~~ the oscillation of a string i.e. of a guitar. If the string is of length L and fixed at both ends, the constant c describes the velocity of the wave and f represents the known starting position. To provide an illustrative example in two-dimensional space, one can consider a rock falling into water, where the propagation of the waves in the water can be described by 2D wave equations.

compare d with

1.1. Differential Model

We're further considering the 2D wave equation

$$u_{tt} = c^2 \nabla^2 u \tag{2}$$

with

$$\nabla^2 u = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \tag{3}$$

and u prescribed as a gaussian pulse

$$u_{inital} = e^{-0.1((x-x_0)^2+(y-y_0)^2)}. \tag{4}$$

1.2. Mesh & Notation

A mesh in time and space is introduced. The mesh in time consists of time points $t_0 = 0 < t_1 < ... < t_{N_t}$, with a constant spacing $\Delta t = t_{n+1} - t_n$. We implement the finite difference model in a box-shaped domain with separate mesh points in x- and y-dimension with $x_0 < x_1 < ... < x_{N_x}$ and $y_0 < y_1 < ... < y_{N_y}$ with a constant and equal mesh spacing $\Delta x = x_{n+1} - x_n$ and $\Delta y = y_{n+1} - y_n$ where $\Delta x = \Delta y$. For $n = 0$ we have the initial conditions $u = u_{initial}$ and $u_t = 0$. At the boundaries $i = 0, N_x$ we apply different specific boundary conditions stated in the implementation section according to the exercise description in appendix A. The unknown u at mesh point (x_i, y_j, t_n) is denoted by $u^n_{i,j}$.

1.3. Discretization

The 2D wave equations are discretized by assembling building blocks for discretization of 1D wave equations, because the multidimensional versions just contain terms of the same type as those in 1D. The building blocks were taken from the finite difference toolbox in [1].

$$\frac{u^{n+1}_{i,j} - 2u^n_{i,j} + u^{n-1}_{i,j}}{\Delta t^2} = c^2 \frac{u^n_{i+1,j} - 2u^n_{i,j} + u^n_{i-1,j}}{\Delta x^2} + c^2 \frac{u^n_{i,j+1} - 2u^n_{i,j} + u^n_{i,j-1}}{\Delta y^2} + f^n_{i,j} \tag{5}$$

Let us turn the wave equation into a system of partial differential equations with only first order time derivatives by introducing $v = u_t$

$$\begin{cases} u_t = v \\ v_t = c^2 u_{xx} u_{yy} \end{cases} \tag{6}$$

Assuming that all values at time levels n and $n-1$ are known, we can solve for the only unknown $u_{i,j}^{n+1}$ in a simple forward Euler scheme

$$\begin{aligned} u_{i,j}^{n+1} &= u_{i,j}^n + \Delta t \cdot v_{i,j}^n \\ v_{i,j}^{n+1} &= v_{i,j}^n + \frac{\Delta t c^2 (u_{i+1,j}^n - 2u_{i,j}^n + u_{i-1,j}^n)}{\Delta x^2} + \frac{\Delta t c^2 (u_{i,j+1}^n - 2u_{i,j}^n + u_{i,j-1}^n)}{\Delta y^2} + f_{i,j}^n \end{aligned} \tag{7}$$

2. Implementation

We shall now describe various R [2] implementations for solving standard 2D linear wave equations. This is done with help of the R-package `ReacTran` [4] which originally was designed to solve reactive transport equations but can be used to model general partial differential equations of order ≤ 2 [3] and enables us to easily display the model in perspective plots.
After setting up the model space with the function `setup.grid.1D` the wave equation is implemented as `wavefunction` with the corresponding boundary conditions using `tran.2D` where the boundaries are defined by

`C.x.up`	=	$u_{i+1,j}$	=	E
`C.x.down`	=	$u_{i-1,j}$	=	W
`C.y.up`	=	$u_{i,j+1}$	=	N
`C.x.down`	=	$u_{i,j-1}$	=	S

where E, W, N, S abbreviate the cardinal directions. The system of 2D partial differential equations, is beeing solved with the function `deSolve::ode.2D` [5] using `wavefunction`, the initial values for the ODE system defined by the initial model space and the discrete time steps as arguments.

2.1. Periodic Boundary conditions and Gaussian Pulse

Project I.4 exercise 1

We start with a constant wave velocity and periodic boundary conditions. The wave equation is to be solved in the space-time domain $\Omega \times (0, T]$ where $\Omega = (0, L_x) \times (0, L_y)$ with $L_x = L_y = 100$ forming a rectangular spatial domain. The complete initial-boundary value problem is defined by

$$u_{tt} = c^2(u_x x + u_y y) + f(x, y, t), \qquad (x, y) \in \Omega, t \in (0, T], \tag{8}$$

$$u_{(x,y,0)} = e^{-0.1((x-x_0)^2+(y-y_0)^2)}, \qquad (x, y) \in \Omega \tag{9}$$

$$u_{t(x,y,0)} = 0, \qquad (x, y) \in \Omega \tag{10}$$

$$u = 0, \qquad (x, y) \in \partial\Omega, t \in (0, T] \tag{11}$$

where $\partial\Omega$ is the boundary of Ω. In this case the four sides of the rectangle. Of this set of partial differential equations, we get the explicit updating formula for the implementation in a program by

$$u^{n+1} = -u_{i,j}^{n+1} + 2u_{i,j}^n + C_x^2(u_{i+1,j}^n - 2u_{i,j}^n + u_{i-1,j}^n) + C_y^2(u_{i,j+1}^n - 2u_{i,j}^n + u_{i,j-1}^n) + \Delta t^2 f_{i,j}^n \quad (12)$$

for all interior mesh points. The constants are defined as

$$C_x = c\frac{\Delta t}{\Delta x};\;\; C_y = c\frac{\Delta t}{\Delta y}. \quad (13)$$

The periodic boundary conditions were implemented by setting $u_{i,j}^{N+1} = u_{i,j}^0$ and $u_{i,j}^{-1} = u_{i,j}^N$for $i = 0, j = 0, ..., N_y$; $j = 0, i = 0, ..., N_x$; and $i = N_x, j = 0, ..., N_y$; $j = N_y, i = 0, ..., N_x$.
For the first step, $n = 0$, (12) is combined with the discretization of the initial condition $u_t = 0$, to obtain a special formula for $u_{i,j}^1$ at the interior mesh points:

$$u^1 = u_{i,j}^0 + \Delta t \cdot 0 + \frac{1}{2}C_x^2(u_{i+1,j}^0 - 2u_{i,j}^0 + u_{i-1,j}^0) + \frac{1}{2}C_y^2(u_{i,j+1}^0 - 2u_{i,j}^0 + u_{i,j-1}^0) + \frac{1}{2}\Delta t^2 f_{i,j}^0 \quad (14)$$

We know have all we need to define the algorithm:

1. Set initial condition $u_{i,j}^0 = e^{-0.1((x-x_0)^2+(y-y_0)^2)}$
2. Compute $u_{i,j}^1$ from (12)
3. Set the boundary condition $u_{i,j}^{N+1} = u_{i,j}^0$ and $u_{i,j}^{-1} = u_{i,j}^N$ for $i = 0, N_x,\;\; j = 0, N_y$
4. For $n = 1, 2, ..., N_t$:
   a) Find $u_{i,j}^{n+1}$ from (12) for all internal mesh points.
   b) Set $u_{i,j}^{n+1} = 0$ with $(x_0, y_0) = (50, 50)$ for the boundaries $i = 0, N_x,\;\; j = 0, N_y$

A simulation for $t = (0, 70)$ by steps of $\Delta t = 5$ leads us to an output as in figure 1 and figure 5 from the appendix (as well as the animation `I_4_1.gif` not included in this report).

2.2. Flat Initial Configuration and Various Sources

Project I.4 exercises 2 & 4

We continue with the same configurations as above, but with additional wave sources of the form

$$u_t = v + f_{(x,y,t)} \;\text{ with }\; f_{(x,y,t)} = sin(2\pi kt) \cdot e^{-0.1((x-x_0)^2+(y-y_0)^2)} \quad (15)$$

where k is the frequency of the sinus wave. Having the second source at $(x_0^{'}, y_0^{'}) = (50, 70)$ results in figure 2 with $k = 0.1$. With a change of the frequency, the height of the second source and the impact on the first source (at $(x_0^{''}, y_0^{''}) = (50, 50)$) is reduced as shown in the appendix figure (6) for $k = 0.005$. Further we added a third and fourth source at $(x_0^{'''}, y_0^{'''}) = (50, 25)$ and $(x_0^{\text{iv}}, y_0^{\text{iv}}) = (25, 25)$ respectively, of which the output is shown in appendix figures (7, 8).

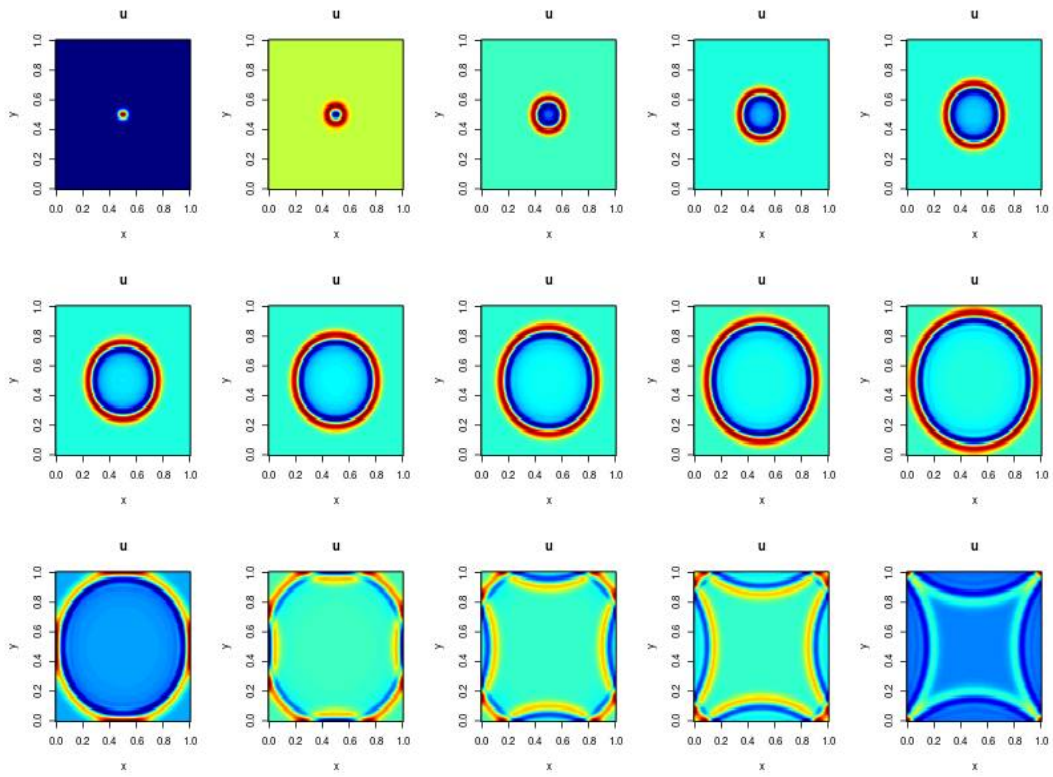

Figure 1: Simulation of the 2D wave equation with periodic boundary conditions, an initial gaussian pulse and $u_t = 0$ at the beginning.

2.3. Different Boundary Conditions for the Cardinal Points

Project I.4 exercise 5

Taking the implementation of section 2.1, we show the behaviour of the wave for different boundary conditions on the system borders with Dirichlet boundary conditions on the North and South by imposing the value for the update function on the edges to

$$\begin{array}{lcll} \texttt{C.y.up} & = & u_{i,j+1} & = & 0 \\ \texttt{flux.y.up} & = & u_{y(i,j+1)} & = & 0 \\ \texttt{C.y.down} & = & u_{i,j-1} & = & 0 \\ \texttt{flux.y.up} & = & u_{y(i,j-1)} & = & 0 \end{array}$$

And Neumann boundary conditions were applied on the East and West by imposing the flux $u_x = u_y = 0$ through the edge

$$\begin{array}{lcll} \texttt{C.x.up} & = & u_{i+1,j} & = & 0 \\ \texttt{flux.x.up} & = & u_{x(N_x+1,j)} & = & u_{0,j} \\ \texttt{C.x.down} & = & u_{i-1,j} & = & 0 \\ \texttt{flux.x.up} & = & u_{x(-1,j)} & = & u_{Nx,j} \end{array}$$

which simply means that these edges are isolated and no flux passes these edges of the system.

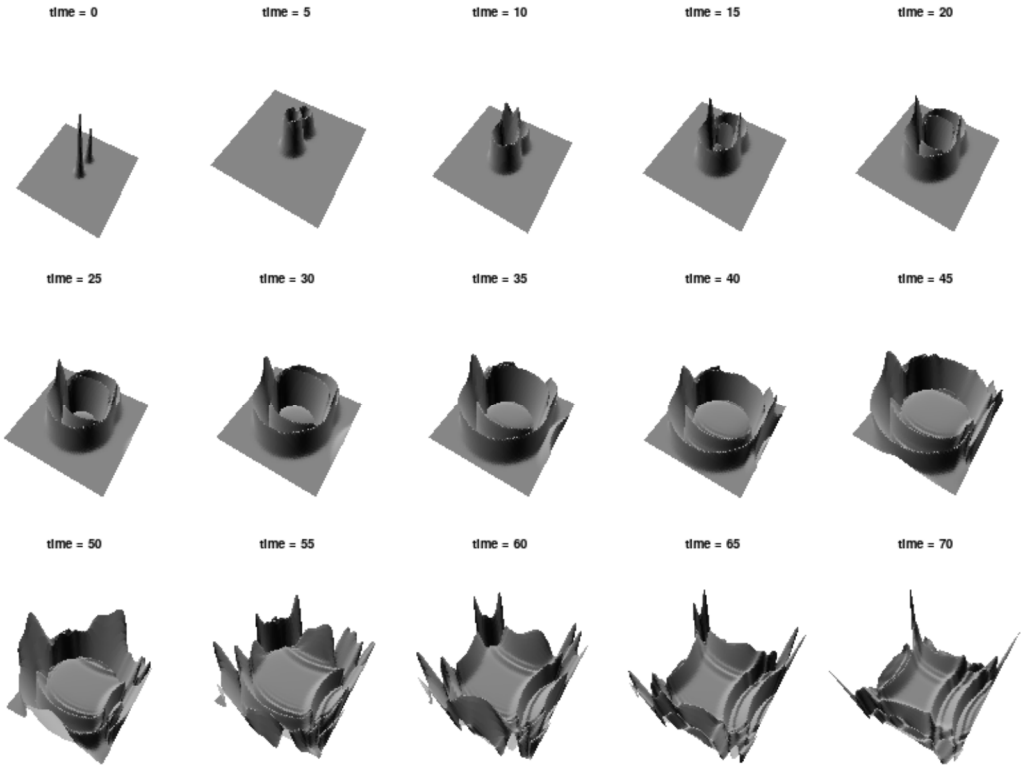

Figure 2: Simulation of the 2D wave equation with periodic boundary conditions, an initial gaussian pulse and $u_t = 0$ at the beginning plus a second source in form of a gaussian shaped sinus wave. The frequency of (15) $k = 0.1$.

The behaviour of the wave is displayed in figure 3 where it's clearly shown, that on one side of the system space the wave is absorbed and on the other its reflected.

3. Convergence towards continuous Solution

We check if the simulation of section 2.1 converges towards the continuous solution for $t = [1, 10]$. Which in other words means that we're interested on the behaviour of $u^n_{i,j}$ and the relationship between $\Delta x = \Delta y$ and Δt for $\Delta x = \Delta y \to 0$ and $\Delta t \to 0$. Therefore we compare the numerical result $u^n_{i,j}$ with the continuous solution $U_{t,x,y} = U^n_{i,j}$ by the relative error

$$E_{(t)} = \frac{||e_{(t,x,y)}||}{||u_{t,i,j}||} \tag{16}$$

using the infinity norm $||f_{(x)}|| = max|f_{(x)}|$ and with $e_{(t,x,y)} = |U_{t,x,y} - u_{t,x,y}|$.
But since we don't know the exact solution for $U^n_{t,x,y}$, we apply Runge's Rule. We first introduce

$$h = \Delta t = \Delta x^2 \cdot c = \Delta y^2 \cdot c \tag{17}$$

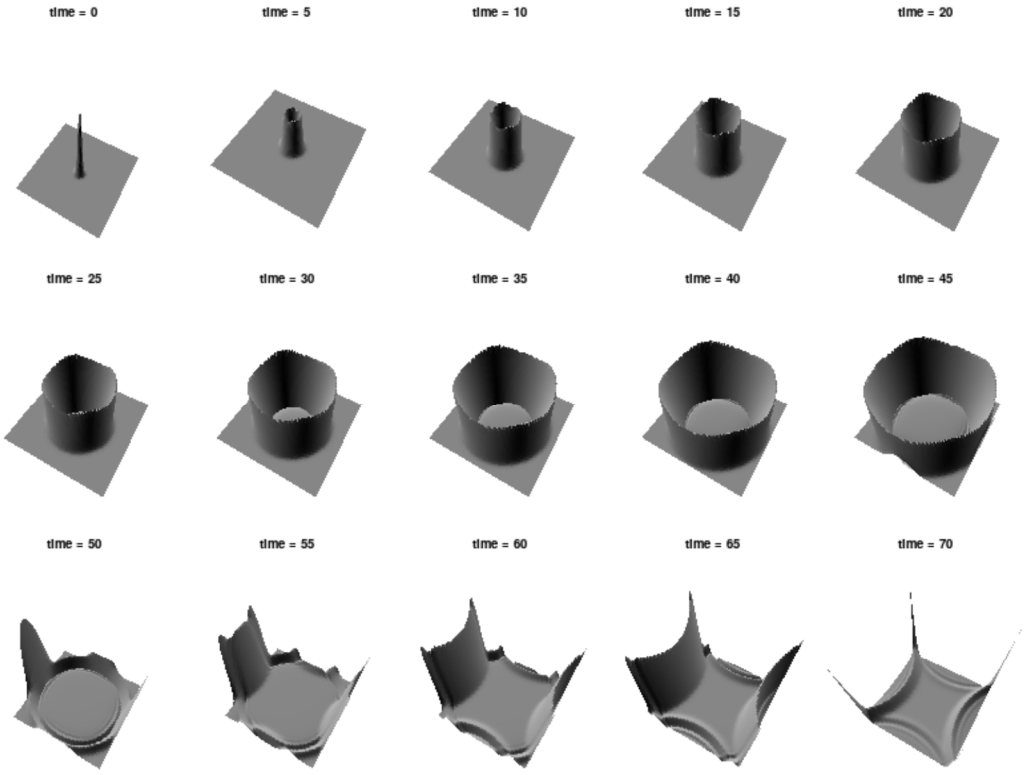

Figure 3: Simulation of the 2D wave equation with Dirichlet boundary conditions on the North and South side and Neumann boundary conditions on East and West side with one initial gaussian pulse.

with c as a constant and assuming that if $u_{h\to 0} = U$ with $u_h = u_{(t,x,y)}$ for some h. Runge's Rule says (in short) that if

$$e_{h_{(t)}} = ||u_h - u_{h/2}|| \tag{18}$$

is small, then $||u_h - U||$ is also small. To approximate $h \to 0$ we stepwise divide h by two and we know then that for small h (18) is allometric. With this we can approximate $||u_h - U||$ leading to a geometric series resulting in

$$||u_h - U|| \leq 2 \cdot e_{h_{(t)}} = 2 \cdot bh^a \quad \text{with} \quad a \geq 1 \tag{19}$$

and the relative error becomes:

$$E_{(t)} \leq 2 \cdot \frac{||u_h - u_{h/2}||}{||u_h||} \tag{20}$$

3.1. Implementation of the Error Estimate

Project I.4 exercise 6

We applied for each h algorithm 2.1 to calculate $u_{i,j}$ at time steps $t = [1, 2, ..., 10]$. To calculate $||u_h||$ and $e_{h_{(t)}}$ we used the R function `norm(x, type = "i")` with `"i"` specifying the maximum absolute row sum.

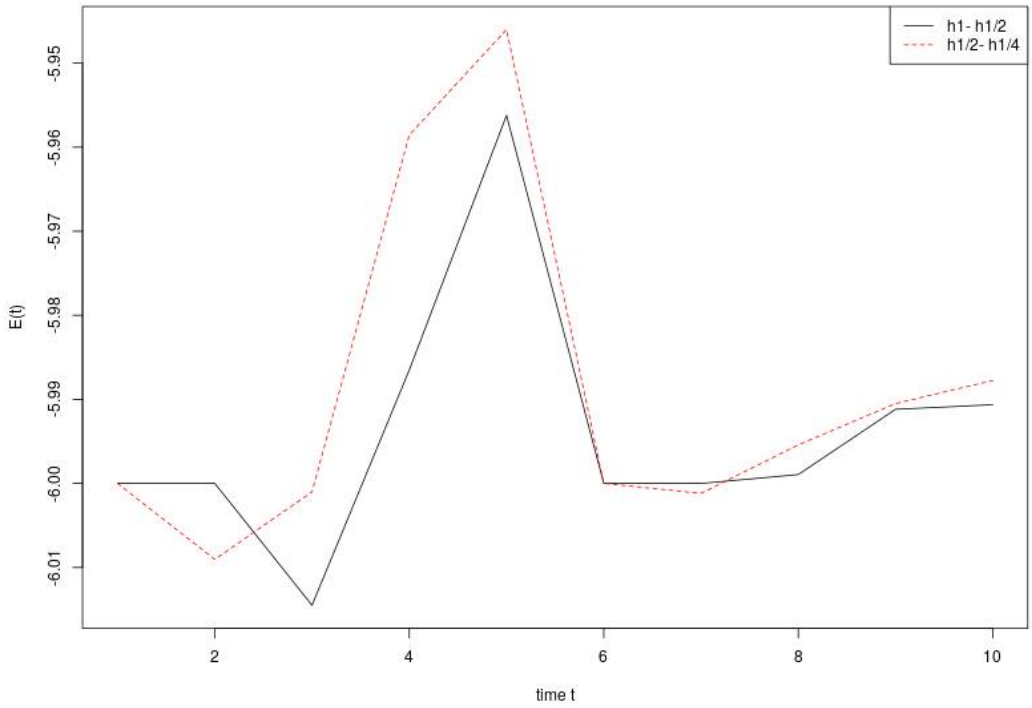

Figure 4: Relative error development over time for each h in a different color and shape. It is recommended to refere to the last section 4 for a statement about the significance of this graphic.

4. Conclusion

In figure 1 of section 2.1 one clearly sees the propagation of the bell-shaped wave through space in time. The periodic boundary conditions let the waves exit on one side of the rectangular plane and enter on the other again - in an illustrative point-of view. In a topological sense, this means that the space Ω forms into a torus. In the perspective plot from figure 5 in the appendix, one can observe, most obviously at $time = 70$, that the boundary conditions couldn't beeing correctly implemented for the corners.
Modelling multiple wave sources at once, gives an interesting overlay of the waves as displayed in figure 2. However based on this graphic alone, it can't be finally shown, if the wave inference is modelled correctly. We suggest for an improvement to enlarge the model-space and apply with Dirichlet boundary conditions for all edges.
Unfortunately the error estimatation from section 3.1 couldn't properly beeing implemented with `ReacTran`. For a smaller h the grid size is refined and hence the matrix $u_{i,j}$ has a different size for each h. This becomes a problem when calculating $e_{h_{(t)}}$. To work around this issue, we assumed that $||u_h - u_{h/2}|| = ||u_h|| - ||u_{h/2}||$ which is not proofen to be true. Furthermore, with simulations for $h < 1/4$ the R-session crashed. This was not due to calculation power reasons, as a multicore implementation of the for-loop over each `h` didn't solved the issue either.

References

[1] D. M. Causon, C. G. Mingham *Introductory Finite Difference Methods for PDEs,* Ventus Publishing ApS, 2010.

[2] R Core Team (2016) *R: A language and environment for statistical computing.* R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0, URL http://www.R-project.org. R version 3.2.2

[3] Karline Soetaert and Filip Meysman *Solving partial differential equations, using R package ReacTran* Centre for Estuarine and Marine Ecology, Netherlands Institute of Ecology, The Netherlands

[4] Soetaert, Karline and Meysman, Filip, 2012. *Reactive transport in aquatic ecosystems: Rapid model prototyping in the open source software* R Environmental Modelling & Software, 32, 49-60.

[5] Karline Soetaert, Thomas Petzoldt, R. Woodrow Setzer (2010). *Solving Differential Equations in R: Package deSolve.* Journal of Statistical Software, 33(9), 1–25. `http://dx.doi.org/10.18637/jss.v033.i09`