UPPSALA
UNIVERSITET

Institutionen för informationsteknologi
Teknisk databehandling

Besöksadress:
MIC hus 2, Polacksbacken
Lägerhyddvägen 2

Postadress:
Box 337
751 05 Uppsala

Telefon:
018–471 0000 (växel)

Telefax:
018–51 19 25

Hemsida:
http:/www.it.uu.se/

Department of Information Technology
Scientific Computing

Visiting address:
MIC bldg 2, Polacksbacken
Lägerhyddvägen 2

Postal address:
Box 337
SE-751 05 Uppsala
SWEDEN

Telephone:
+46 18–471 0000 (switch)

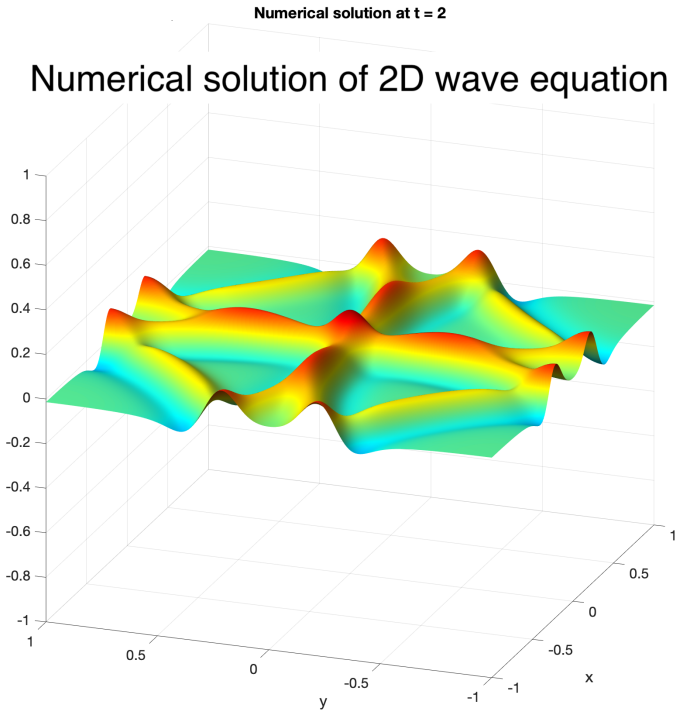
Telefax:
+46 18–51 19 25

Web page:
http:/www.it.uu.se/

Project - Numerical wave propagation

Introduction

In the present study the focus is on the SBP-Projection method to solve a scalar wave equation on second order form for various boundary conditions, i.e., an Initial-Boundary Value Problem (IBVP). Wave equations are important since they model many different types of waves in physics, such electromagnetic waves, elastic waves and acoustic waves. A 2D computation of the second order wave equation is presented in the figure below.



When solving IBVP the first step is almost always to discretize the the spatial part, leading to large ODE systems. Two common methods are: 1) the Finite Element Method (FEM), and 2) the Finite Difference Method (FDM). How to discretize the spatial part of an IBVP is one of the main topics in this course. In this project, the spatial discretisation is constructed from SBP finite difference operators combined with the Projection method to impose different types of well-posed boundary conditions. SBP operators of second, fourth and sixth order accuracy is provided as both Matlab and Python functions. The resulting ODE system will be integrated in time using the classical fourth order accurate Runge-Kutta method (RK4).

The wave equation

The second order wave equation: $\mathbf{u}_{tt} = c^2\nabla^2\mathbf{u}$, models wave-propagation, where c is the wave speed. For electromagnetic waves c equals the speed of light, and in 2D) $\mathbf{u}$ denotes either the electric or the magnetic field in one direction. For acoustic waves c equals the speed of sound and $\mathbf{u}$ denotes pressure. Here we will solve this equation on a bounded domain (in 1D and 2D), with non-periodic boundary conditions (BC). In the most general case speed of sound is a function of time and space, but here we will assume either a constant wave speed or that it depends only on space.

Initial-boundary value problems in 1D

We start with the one dimensional (1D) case, and restrict the computational domain to $-1 \leq x \leq 1$. Here $\mathbf{u} = u(x,t)$. Since the system is on second order form (second derivative in time) two initial conditions are needed: $u = f(x)$, $u_t = p(x)$, $-1 \leq x \leq 1$, where $f(x)$ and $p(x)$ are initial data. In the present project $p(x) = 0$, i.e., the system starts from rest, which is most often the case in applications. In Figure 1 we plot an initial Gaussian profile $f(x) = \exp\left(-(5\,x)^2\right)$, that will be laters used in the computations.

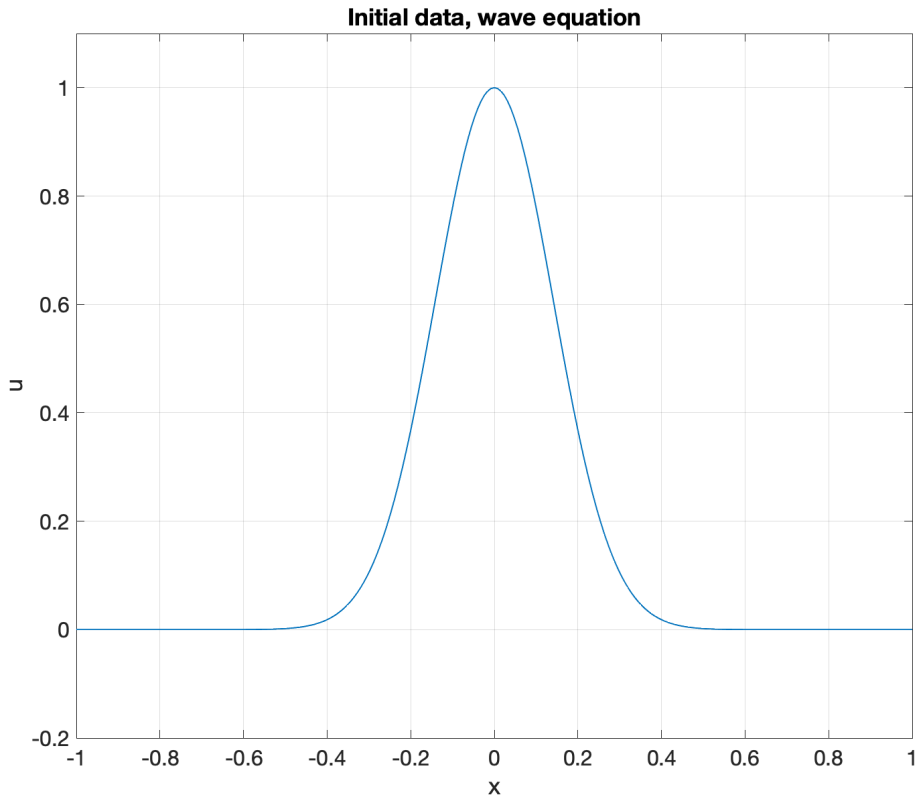


Figure 1: Initial data $u(x,0) = f(x)$ used in the computations.

A well-posed IBVP requires one BC at the left and right boundaries (two in total), denoted $L^{(l)}u = g^{(l)}(t)$ and $L^{(r)}u = g^{(r)}(t)$, respectively. There are many different types of well-posed BC for the 1D wave equation. The full problem given by,

$$\begin{array}{llll} u_{tt} = c^2 u_{xx}, & & -1 \leq x \leq 1 & t \geq 0, \\ L^{(l)}u = g^{(l)}(t), & & x = -1, & t \geq 0, \\ L^{(r}u = g^{(r)}(t), & & x = 1, & t \geq 0, \\ u = f(x), & u_t = 0, & -1 \leq x \leq 1, & t = 0, \end{array} \tag{1}$$

is referred to as an IBVP.

Now consider the following IBVP,

$$\begin{array}{lll} u_{tt} = c^2 u_{xx}, & -1 \leq x \leq 1 & t \geq 0, \\ L^{(l)}u = \alpha^{(l)} u_t + \beta^{(l)} u + \gamma^{(l)} u_x = 0, & x = -1, & t \geq 0, \\ L^{(r)}u = \alpha^{(r)} u_t + \beta^{(r)} u + \gamma^{(r)} u_x = 0, & x = 1, & t \geq 0, \\ u = f(x), \quad u_t = 0, & -1 \leq x \leq 1, & t = 0, \end{array} \tag{2}$$

where $c = c(x) > 0$, and $\alpha^{(l)}$, $\beta^{(l)}$, $\gamma^{(l)}$, $\alpha^{(r)}$, $\beta^{(r)}$, $\gamma^{(r)}$ are some constants.

Assignment 1: Derive conditions such that (2) is a well-posed IBVP for the special case when $\alpha^{(l,r)} = 0$. Then present 3 different well-posed boundary conditions.

Assignment 2: Derive conditions such that (2) is a well-posed IBVP assuming that $\alpha^{(l,r)} = 1$. Show that the special case when $\beta^{(l,r)} = 0$, $\gamma^{(l)} = -c$, $\gamma^{(r)} = c$ is well-posed and leads to damping of energy at the two boundaries. (In literature, this set of BC are referred to as absorbing boundary conditions.)

Analytic solutions

The following Gaussian profiles,

$$\theta^{(1)}(x,t) = \exp\left(-\left(\frac{x - c\,t}{r_*}\right)^2\right), \; \theta^{(2)}(x,t) = -\exp\left(-\left(\frac{x + c\,t}{r_*}\right)^2\right), \tag{3}$$

are introduced where r_* defines the width of the Gaussian. Here assuming that c is constant.

Let $r_* = 0.2$, where the domain is restricted to $-1 \leq x \leq 1$. If initial data are set to:

$$u(x,0) = \theta^{(1)}(x,0), \quad u_t(x,0) = 0\,, \tag{4}$$

analytic solutions to (1) using either homogeneous Neumann or Dirichlet

BC are given by,

$$\begin{aligned} u^{(1)}(x,t) &= +\tfrac{1}{2}\theta^{(1)}(x+2,t) - \tfrac{1}{2}\theta^{(2)}(x-2,t), \quad \text{Neumann BC} \\ u^{(2)}(x,t) &= -\tfrac{1}{2}\theta^{(1)}(x+2,t) + \tfrac{1}{2}\theta^{(2)}(x-2,t), \quad \text{Dirichlet BC} \end{aligned} \tag{5}$$

after the Gaussian pulses have been reflected at the boundaries, i.e., approximately when $c\,t \in [1.75,\, 2.25]$. (When doing a numerical convergence study, avoid using end time excactly $T = 2$.)

Spatial discretization

The spatial domain ($-1 \le x \le 1$) is discretized using the following m equidistant grid points (with grid-spacing h):

$$x_j = -1 + (j-1)\,h, \quad j = 1, 2, \ldots, m, \quad h = \tfrac{2}{m-1}\,.$$

The approximate solution at grid point x_j is denoted u_j, and the discrete solution vector $u = [u_1, u_2, \ldots, u_m]^T$ (where superscript ^{T} denotes transpose, i.e., u is a column vector of length m). A semi-discrete SBP approximation of (2) can be written

$$\begin{aligned} &u_{tt} = c^2 D_2\, u, && t \ge 0, \\ &Lw = 0, && t \ge 0, \\ &u = f, \quad u_t = 0, && t = 0, \end{aligned} \tag{6}$$

where $w^T = [u^T\,, u_t^T]$ and D_2 is a second order derivative SBP operator. To solve this using the classical 4th order Runge-Kutta method (RK4) we introduce $v = u_t$ and rewrite this to first order form,

$$\begin{aligned} &\begin{bmatrix} u \\ v \end{bmatrix}_t = \begin{bmatrix} 0 & I \\ c^2 D_2 & 0 \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix}, && t \ge 0, \\ &L \begin{bmatrix} u \\ v \end{bmatrix} = 0, && t \ge 0, \\ &\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} f \\ 0 \end{bmatrix}, && t = 0, \end{aligned} \tag{7}$$

where I is the $m \times m$ unit matrix (`eye(m)` in Matlab).

Assignment 3: Derive an SBP-Projection approximation of (2) for the well-posed boundary conditions derived in the first assignment, and prove stability. Let P denote the projection operator. Here the semi-discrete SBP-Projection approximation can be written,

$$\begin{aligned} &u_{tt} = A\, u, && t \ge 0, \\ &u = f, \quad u_t = 0, && t = 0, \end{aligned} \tag{8}$$

where the 1D discretization matrix is given by $A = Pc^2 D_2 P$. This is because the boundary conditions only involve u and u_x.

Assignment 4: Derive an SBP-Projection approximation of (2) for the well-posed boundary conditions derived in the second assignment and prove stability. Since the boundary conditions now also involve u_t, we can not cast the semi-discrete SBP-Projection approximation into the form given by (8). The SBP-Projection approximation now has to start from the first order (in time) form given by (7).

Assignment 5: Write a MATLAB or Python program that solves the SBP-Projection approximation derived in the previous two assignments (3 and 4), using the classical RK4 method for time-integration. Hence, the program should handle all types of well-posed boundary conditions. It should be easy to change number of grid-points (m) time-step (k), end time (T), initial data, and type of boundary condition. Plot the solution at $t = 0.2, 0.5, 0.7, 1.8$ using $m = 101$ and $k = 0.01$ for Dirichlet, Neumann and absorbing boundary conditions (see assignment 2). Initialize the solutions using (4). **Hint**: It can be instructive to simulate the solution. (To generate a movie in Matlab the function **VideoWriter** is useful.)

Assignment 6: Do a convergence study for the cases with Dirichlet and Neumann BC, based on the analytic solutions given by (5). Do this for the second, fourth and sixth order accurate SBP operators. Initialize the solution using (4). Use for example $m = 51$, $m = 101$, $m = 201$ and $m = 301$ grid-points. Use $k = 0.1\,h$ for the time step to make sure the solution is stable. Measure the error at for example $T = 1.8$ or $T = 2.1$. Present a table (for each type of BC) with the discrete l_2 norm of the error and convergence rate q. Do you obtain the expected convergence behaviour?

The convergence rate is calculated as

$$q = \frac{\log_{10}\left(\frac{\|u-v^{(m_1)}\|_h}{\|u-v^{(m_2)}\|_h}\right)}{\log_{10}\left(\frac{m_2}{m_1}\right)}\,, \tag{9}$$

where u is the analytic solution, and $v^{(m_1)}$ the corresponding numerical solution with m_1 grid points, and

$$\|u - v^{(m_1)}\|_h \equiv \frac{1}{\sqrt{m_1}}\sqrt{\sum_{j=1}^{m_1}(u(x_j) - v_j)^2}\,,$$

is the discrete l_2 norm of the error. In Matlab we can use

`sqrt(h)*norm(u-v)` to compute the discrete l_2 norm of the error, where v is the numerical solution vector (at a certain time-level) and u the corresponding analytic solution (also a vector of length m). (Note that $h = \frac{2}{m-1}$, which is why $h \simeq 1/m$.)

The two dimensional problem

A 2D extension of (1) with Neumann BC onto the square domain restricted by $-1 \leq x,\, y \leq 1$ is given by

$$\begin{array}{llll} u_{tt} = c^2(u_{xx} + u_{yy}), & -1 \leq x,\, y \leq 1, & t \geq 0, \\ u_{x,\,y} = 0, & x,\, y \in \delta\Omega & t \geq 0, \\ u = f(x,y), \quad u_t = 0, & -1 \leq x,\, y \leq 1, & t = 0, \end{array} \tag{10}$$

where $\delta\Omega$ represent the boundary of the computational domain (here defined by the square). An equivalent 2D extension of (1) with Dirichlet BC is given by,

$$\begin{array}{llll} u_{tt} = c^2(u_{xx} + u_{yy}), & -1 \leq x,\, y \leq 1, & t \geq 0, \\ u = 0, & x,\, y \in \delta\Omega & t \geq 0, \\ u = f(x,y), \quad u_t = 0, & -1 \leq x,\, y \leq 1, & t = 0, \end{array} \tag{11}$$

To simplify the 2D notation we will make use of the Kronecker product:

$$B \otimes C = \begin{bmatrix} b_{1,1}\,C & \cdots & b_{1,m}\,C \\ \vdots & & \vdots \\ b_{m,1}\,C & \cdots & b_{m,m}\,C \end{bmatrix},$$

where B is a $m \times m$ matrix and C is an $n \times n$ matrix. To simplify notation (without any restriction) we assume the same dimension ($n = m$) in both the x- and y-direction. The square domain is discretized with an $m \times m$-point equidistant grid defined as

$$\begin{aligned} x_j &= -1 + (j-1)\,h, \quad j = 1, 2, \ldots, m, \quad h = \tfrac{2}{m-1}, \\ y_i &= -1 + (i-1)\,h, \quad i = 1, 2, \ldots, m, \quad h = \tfrac{2}{m-1}. \end{aligned}$$

The numerical approximation at grid point $(x_j,\, y_i)$ is denoted $u_{j,i}$. The discrete solution vector is now defined as a "vector of vectors"

$$u = [u_{1,1},\, u_{1,2}, \ldots, u_{1,m},\, u_{2,1},\, u_{2,2}, \ldots, u_{2,m}, \ldots, \ldots, u_{m,1},\, u_{m,2}, \ldots u_{m,m}]\,.$$

Let A denote the discretisation matrix for the 1D problem, presented in (8). The semi-discrete SBP-Projection approximation of the 2D problem (10) or (11) can be written $(A \otimes I) + (I \otimes A)$, where I is the unit matrix of

size $m \times m$. Hence, the semi-discrete approximation of the 2D problem is given by,

$$\begin{aligned} u_{tt} &= ((A \otimes I) + (I \otimes A))\, u, \quad t \geq 0, \\ u &= f, \quad u_t = 0, \qquad\qquad t = 0. \end{aligned} \tag{12}$$

Hint: In Matlab you have the function `kron`, i.e., the 2D solution matrix is given by $kron(A, I) + kron(I, A)$. To make the computations faster and lower the memory storage make sure you store the matrices in `sparse` format.

(If you would like Dirichlet BC at the left and right boundaries (x-direction) and Neumann BC at the top and bottom boundaries (y-direction), the 2D discretisation matrix is given by $(AD \otimes I) + (I \otimes AN)$, where AD (and AN) denotes the 1D discretization matrix using Dirichlet (and Neumann) boundary conditions).

Assignment 7: Discretise (12) with RK4. Write a Matlab or Python program that simulates the wave propagation using $u(x, y, 0) = f(x, y) = e^{-100(x^2+y^2)}$ and $u_t(x, y, 0) = 0$ as initial data. Solve the problem with both Dirichlet and Neumann BC (and a mix if you prefer). Present plots of the solution at $t = 0$, $t = 1$, $t = 2$ and $t = 10$. (Use $m = 201$.) In Figure 2 the initial Gaussian and numerical solution at $t = 0.5$ is presented with 201×201 grid points (and at $t = 1.2$ using Neumann BC, on first page).

Hint: To plot the solution vector u as a function of x, y in Matlab. Use: $[X, Y] = meshgrid(x, y)$, `U=reshape(u,m,m);surface(X,Y,U)`.

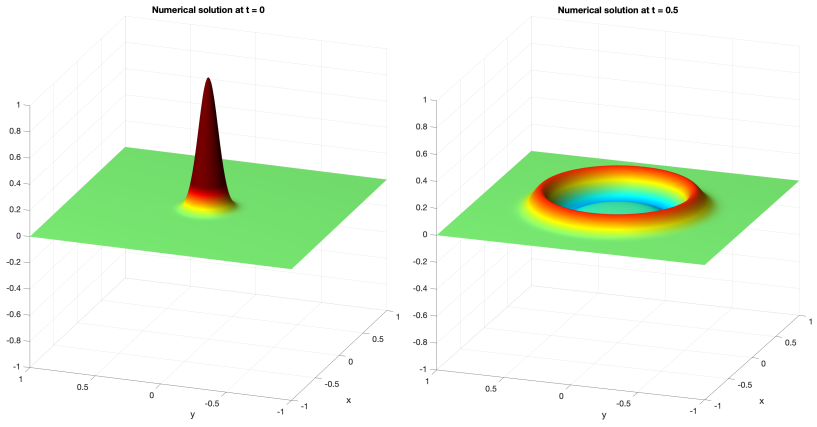


Figure 2: 2D wave equation. Left: initial Gaussian ($t = 0$). Right: numerical solution at $t = 0.5$. Using MATLAB function `surface` to plot.