UPPSALA
UNIVERSITET

Institutionen för informationsteknologi
Beräkningsvetenskap

Besöksadress:
Lägerhyddsvägen 1,
hus 10, Uppsala

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:
Lägerhyddsvägen 1,
hus 10, Uppsala

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/

Lab - Introduction to Finite Difference Methods

The idea here is to get a feeling for three important properties when it comes to numerical approximations of Partial Differential Equations (PDE): *numerical error*, *stability* and *convergence*. Convergence proofs rely on a combination of stability and numerical error analysis. For non-linear PDEs, stability and convergence is non-trivial and most often requires advanced numerical tools. For linear problems, stability and convergence is more or less straightforward with the proper tools (that we will focus on in this course).

Tightly coupled to numerical stability is a property of the underlying PDE, referred to as *well-posedness*. Without a proper understanding of well-posedness, it is not meaningful to discuss and analyze stability and convergence. To achieve well-posedness, it is necessary to impose the correct type and number of boundary conditions. A problem that is not well-posed, is referred to as *ill-posed*. A PDE can also be ill-posed (independent of the boundary conditions) if the parameters have incorrect signs. An example is the following PDE: $u_t = \mu u_{xx}$. For $\mu < 0$ this problem is ill-posed (why?).

A few of the more well-known PDEs are: Navier-Stokes equations, Euler equations, Maxwell's equations, the Schrödinger equation, acoustic wave equation, heat equation, Poisson equation, Helmholtz equation, Dirac equation and Einstein field equations. There are essentially 3 different types of PDEs: Hyperbolic, Parabolic and Elliptic. Some equations however (such as the Schrödinger equation) are neither of these three types. Here we will focus on hyperbolic PDEs. We offer the lab in both Matlab and Python. Start by downloading
`Lab1_Matlab.zip` or `Lab1_Python.zip` and unpack.

Part 1: Numerical error

In this section you need the following programs: `Lab11_SC.m`, `Plotta_1D.m`, `Initial_P1.m`, `Initial_P2.m`, `RHS_P.m`, `Periodic_E_SC`, `Periodic_S6_SC`, `Periodic_AD_SC` and put them in the same catalog. The corresponding Python files have identical file-names but with the extensions `.py`.

The simplest hyperbolic PDE is given by

$$u_t + au_x = 0 \ ,$$

with periodic boundary conditions. As initial data we will set $u(0,t) = f(x)$ (here some periodic function). The analytic solution can in this case be written $u(x,t) = f(x - a\,t)$. In Figure 1 two examples of initial data (smooth and non-smooth) are shown.

This equation models wave-propagation in one direction. The parameter a defines the wave speed. This equation can be solved numerically using for example the finite difference (FD) method. The most widely used FD stencil is the central second order accurate approximation, but higher order stencils are usually more efficient. For non-linear problems (for example when $a = u$) or when we have non-smooth parameters or data some type of *artificial dissipation* is needed to avoid oscillations.

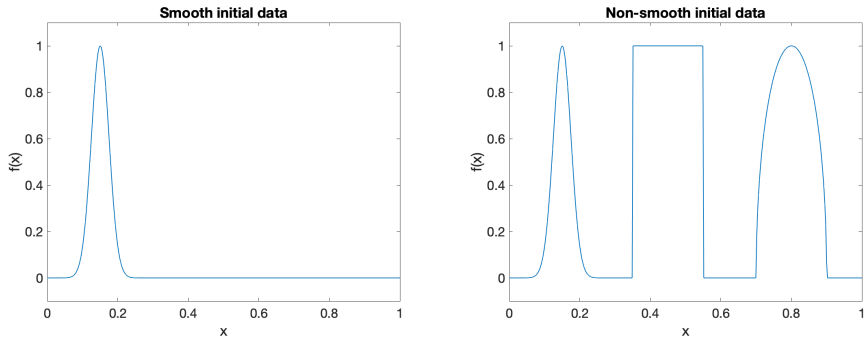

Figure 1: Initial data $f(x)$, smooth (left) and non-smooth (right).

Smooth initial data

In the first test we will use smooth initial data where

$$f(x) = \begin{cases} e^{-300(2x-0.3)^2} & \text{if } |2x-0.3| \leq 0.25, \\ 0 & \text{otherwise.} \end{cases}$$

Here we will test different types of central finite difference stencils, both explicit (of orders 2, 4, 6, 8, 10, 12) and an implicit spectral accurate finite difference stencil. The grid-spacing is given by $h = \frac{1}{m}$, where m denotes the number of grid intervals. The numerical solution is integrated until $t = 25$.

Start by typing *Lab11_SC* in the Matlab window. The numerical solution is integrated in time to $t = 25$ using the fourth order accurate Runge-Kutta method and time-step $k = \frac{h}{10}$ to make the time-integration errors negligible. You will have to specify the name of the avi-file of the generated movie, the number of grid intervals m (i.e, the resolution in space) and the type of finite difference stencil. Observe the behavior of the numerical solution (the blue line) compared to the analytic (exact) solution (the red

line) when grid intervals is set to $m = 50$. Run the program for different orders of accuracy, starting from 2nd and ending with the 12th order explicit finite difference stencil. Then compare the result when using instead the implicit spectral accurate finite difference stencil. What do you observe? The difference in wave speed (between the exact and numerical solution) is refereed to as the *dispersion error*. The discrete l_2-norm of the error is defined as $\sqrt{h}\sum_{i=1}^{m}|u(x_j)-v_j|$, where v_j is the numerical solution at grid-point x_j and $u(x_j)$ the corresponding exact value. This is written out when the program ends.

What happens when you increase the number of grid intervals m for a given finite difference stencil? Does the numerical approximation improve?

What happens when you increase the order of accuracy for a given resolution (m) , does the numerical approximation improve?

How many grid intervals m are required to have an l_2-error smaller than 0.001 for the 2nd, 6th, 12th and the spectral accurate finite difference stencils?

Non-smooth initial data

In the second test we will use non-smooth initial data (see Figure 1) where

$$u_0(x) = \begin{cases} e^{-300(2x-0.3)^2} & \text{if } |2x-0.3| \leq 0.25, \\ 1 & \text{if } |2x-0.9| \leq 0.2, \\ \left(1-\left(\frac{2x-1.6}{0.2}\right)^2\right)^{\frac{1}{2}} & \text{if } |2x-1.6| \leq 0.2, \\ 0 & \text{otherwise.} \end{cases}$$

Again we will test different types of central finite difference difference stencils, both explicit (of orders 2, 4, 6, 8, 10, 12) and an implicit spectral accurate finite difference stencil. Here we will introduce something referred to as *Artificial Dissipation* (AD). AD is sometimes necessary to damp spurious oscillations that are generated by non-smooth features, such as non-smooth data, or strong nonlinear phenomena such as shocks. Spurious oscillations can also occur for under resolved features (meaning that the numerical errors are large with the current number of grid intervals), or when solving ill-posed problems.

Start by typing *Lab12_SC* in the Matlab window. The numerical solution is integrated in time to $t = 10$ using the fourth order accurate Runge-Kutta method and time-step $k = \frac{h}{25}$ to make the time-integration errors negligible. You will have to specify the name of the avi-file of the generated movie, the number of grid intervals and the type of finite difference stencil. Here you can chose to add AD. Observe the behavior of the numerical

solution (the blue line) compared to the analytic (exact) solution (the red line) when grid intervals is set to $m = 50$. Run the program for different orders of accuracy with and without AD.

What happens when you run the program without AD?

What happens when you run the program with AD?

What happens when you increase the number of grid intervals m for a given finite difference stencil? Does the numerical approximation improve?

What happens when you increase the order of accuracy for a given resolution (m) , does the numerical approximation improve?

Part 2: A non-linear model

In this section you need the following programs: `Lab13_SC`, `Lab14_SC`, `Plotta_1D.m`, `Initial_P1`, `RHS_P2`, `RHS_P3` `Periodic_E_SC_Variable`, `BDF ResidualViscosity`, `Periodic_E_SC_Variable` and all files in folder **+implimentations**. The corresponding Python files have identical file-names but with the extensions `.py`.

The nonlinear hyperbolic model:

$$u_t + u\, u_x = 0 \ ,$$

is referred to as inviscid Burgers' equation. This can also be written in conservative form

$$u_t + \left(\frac{u^2}{2}\right)_x = 0 \ .$$

Here the wave speed is given by $a = u$. This model will give rise to non-smooth solutions in finite time (even with smooth initial data). Without sufficient AD the numerical solution does not converge to the analytic solution. How to construct AD for non-linear problems are non-trivial and is not the focus in this course. Here we will test two different methods. The first method (referred to as *upwind*) does not work well for this problem, while the second method (referred to as *residual viscosity*) produce an accurate solution, with sufficiently many grid intervals m. The numerical solution is integrated in time to $t = 2$ using the fourth order accurate Runge-Kutta method and time-step $k = \frac{h}{25}$.

The initial data is given by

$$f(x) = \begin{cases} \frac{1}{2} + e^{-300(2x-0.3)^2} & \text{if } |2x-0.3| \leq 0.25, \\ 0 & \text{otherwise.} \end{cases}$$

Upwind

Here we will test different types of non-central (upwind) finite difference difference stencils, both explicit (of orders 1, 3, 5, 7, 9, 11) and an implicit spectral accurate finite difference stencil. (In fact, these upwind finite difference stencils are identical to the schemes with the addition of AD tested in the previous section.)

Start by typing *Lab13_SC* in the Matlab window. Here you can chose to add AD (to produce upwind) or try without AD.

What happens when you run the program without addition of AD?

What happens when you run the program with addition of AD?

What happens when you increase the number of grid intervals m for a given finite difference stencil? Does the numerical approximation improve?

What happens when you increase the order of accuracy for a given resolution (m) , does the numerical approximation improve?

Residual viscosity

Here we will test different types of residual viscosity finite difference difference stencils of orders 2, 4, 6, 8, 10, 12. This typ of AD is highly efficient (state of the art). Start by typing *Lab14_SC* in the Matlab window. Here we will test central finite difference difference stencils of orders 2, 4, 6, 8, 10 and 12.

What happens when you increase the number of grid intervals m for a given finite difference stencil? Does the numerical approximation improve?

What happens when you increase the order of accuracy for a given resolution (m) , does the numerical approximation improve?

Part 3: Maxwell's equations

In this section you need the following programs: `Maxwell_1D_SC_PDE`, `Maxwell_1D_Interface_SC_PDE`, `Maxwell_2D_Interface_SC_PDE`, `Plotta_Maxwell_1D`, `Plotta_Maxwell_1D_Interface_SC`, `Plotta_Maxwell_2D_Interface_4B`. You also need the files in folder **SBP**.

Assuming no free charges or currents, the 2D Maxwell's equations (in Cartesian coordinates systems) can be written

$$\mathbf{C}\mathbf{u}_t = \mathbf{A}\mathbf{u}_x + \mathbf{B}\mathbf{u}_y \ , \tag{1}$$

where

$$\mathbf{u} = \begin{bmatrix} E^{(x)} \\ H^{(z)} \\ E^{(y)} \end{bmatrix} \tag{2}$$

denotes the components of the electric and magnetic fields. (Here $E^{(x)}$ is for example the electric field in the x-direction and $H^{(z)}$ the magnetic field in the z-direction.) The matrices $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{C}$ are given by

$$\mathbf{A} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & -1 & 0 \end{bmatrix} \ , \mathbf{B} = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \ , \mathbf{C} = \begin{bmatrix} \epsilon & 0 & 0 \\ 0 & \mu & 0 \\ 0 & 0 & \epsilon \end{bmatrix} \ . \tag{3}$$

The permittivity ϵ and permeability μ will in general depend on the spatial location within the medium (but here assumed to be time-independent). Almost all the difficulties that arise in finite difference approximations of Maxwells's equations are due to material interfaces and boundary conditions. Another difficulty arise due to complex geometries, but that is something we will not address in the present study.

Boundary conditions in 1D

Here we will solve Maxwell's equations in 1D given for example by removing the $E^{(x)}$ component. We will consider the following 1D model of Maxwell's equations,

$$\mathbf{C}\mathbf{u}_t = \mathbf{A}\mathbf{u}_x \ , \tag{4}$$

where now

$$\mathbf{u} = \begin{bmatrix} E \\ H \end{bmatrix} \ , \mathbf{A} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \ , \mathbf{C} = \begin{bmatrix} \epsilon & 0 \\ 0 & \mu \end{bmatrix} \ . \tag{5}$$

E and H signifies the mutually perpendicular tangential electric and magnetic field components. We will consider the non-dimensional form of Maxwell's equations. The normalized quantities in Eq. (4) are related to the physical quantities, $\tilde{t}$, $\tilde{x}$, $\tilde{E}$, and $\tilde{H}$ as

$$t = \frac{c\,\tilde{t}}{L_*} \ , \ x = \frac{\tilde{x}}{L_*} \ , \ E = \frac{\tilde{E}}{Z_0} \ , \ H = \tilde{H} \ ,$$

where $c = \frac{1}{\sqrt{\epsilon_0 \mu_0}}$ is the vacuum speed of light with ϵ_0 and μ_0 being the vacuum permitivity and permeability, respectively, and $Z_0 = \sqrt{\frac{\mu_0}{\epsilon_0}}$ is the vacuum impedance. The expression L_* is an appropriate length scale, usually taken as the mean wavelength, λ, of the electromagnetic wave. In this

setting, Eq. (4) describes waves propagating at unit speed and with lengths measured in units of wavelengths.

There are a variety of possible boundary conditions for Maxwell's equations. When providing boundary conditions it it is important that we obey the physics, as well as the mathematical properties (referred to as Well-posedness) of this system. Roughly speaking: well-posedness requires that we provide the correct number of boundary conditions, but also such that we achieve a bounded solution. Here we will test 4 different sets of boundary conditions for (4)

1. Specify E at the two boundaries.
2. Specify H at the two boundaries.
3. Specify a linear combination $E + \beta\, H$ and $E - \beta\, H$ at the left and right boundary, respectively.
4. Specify both E and H at the two boundaries.

Start by typing *Maxwell_1D_SC_PDE* in the Matlab window. The numerical solution is integrated in time to $t = 1.8$ using the fourth order accurate Runge-Kutta method and time-step $k = \frac{h}{10}$ to make the time-integration errors negligible. You will have to specify the name of the avi-file of the generated movie, the number of grid intervals m (i.e, the resolution in space) and the order of accuracy of the SBP finite difference operator. You will then chose between the 4 types of boundary conditions listed above. For boundary condition type 3, you will also specify the value of β. Try different values of β including ± 1 and ± 2. For the special case when $\beta = 0$ we get back the first choice of boundary condition (specifying the electric component).

In the numerical tests it is recommended to use at least $m = 100$ grid-points. The grid-spacing h is here given by $\frac{2}{m-1}$.

What happens when β is positive or -1?

What happens when you specify boundary condition typ 4, using for example SBP operators 6 and 7?

Hint: *boundary conditions type 4 and type 3 (with positive β) are ill-posed.*

Coupling of two different media in 1D

Consider Maxwell's equations in 1D in the presence of a dielectric boundary, i.e., an internal boundary between two different materials. The result-

ing system can be written,

$$\begin{aligned} \mathbf{C}^{(l)}\mathbf{u}_t^{(l)} &= \mathbf{A}\mathbf{u}_x^{(l)} \ , x \in [x_l,\, x_I] \\ \mathbf{C}^{(r)}\mathbf{u}_t^{(r)} &= \mathbf{A}\mathbf{u}_x^{(r)} \ , x \in [x_I,\, x_r] \end{aligned} \tag{6}$$

where

$$\mathbf{C}^{(l,r)} = \begin{bmatrix} \epsilon^{(l,r)} & 0 \\ 0 & \mu \end{bmatrix} \ , \tag{7}$$

and $\epsilon^{(l)} \neq \epsilon^{(r)}$. Hence, at the media interface $x = x_I$ there is a discontinuous jump in the permittivity. The refractive index of the two media is denoted $\eta^{(l,r)} = \sqrt{\epsilon^{(l,r)}}$.

Assume that we initiate the electric (and magnetic) field with a Gaussian profile in the left sub-domain where the refractive index is $\eta^{(l)}$ and the wave speed $c^{(l)} = \frac{c}{\eta^{(l)}}$. (In the present study we consider the non-dimensional form of Maxwell's equations and hence $c = 1$.) For $t > 0$ the initial Gaussian profile splits into left-going and right-going Gaussian waves propagating with speed $c^{(l)}$. The right-going wave hits the media interface at $x = x_I$ and splits into a reflected wave and a transmitted wave. The transmission (T) and reflection (R) coefficients are given by

$$T = \frac{2\,\eta^{(l)}}{\eta^{(l)}+\eta^{(r)}}, \quad R = \frac{\eta^{(l)}-\eta^{(r)}}{\eta^{(l)}+\eta^{(r)}} \ .$$

Let E_I denote the incident electric wave. The transmitted electric wave, here denoted E_T, propagate to the right with wave speed $c^{(r)} = \frac{c}{\eta^{(r)}}$ and is given by $E_T = T \cdot E_I$. The reflected electric field, here denoted E_R, propagate to the left with wave speed $c^{(l)}$ and is given by $E_R = R \cdot E_I$.

Start by typing *Maxwell_1D_Interface_SC* in the Matlab window. The numerical solution is integrated in time to $t = 1.7$ using the fourth order accurate Runge-Kutta method and time-step $k = \frac{h}{10}$ to make the time-integration errors negligible. You will have to specify the name of the avi-file of the generated movie, the number of grid intervals m (i.e, the resolution in space) and the order of accuracy of the SBP finite difference operator. You will then chose the value of $\epsilon_r > 0$. For the special case when $\epsilon_r = 1$ we have the same medium on both sides of the interface.

In the numerical tests it is recommended to use at least $m = 100$ grid-points. The grid-spacing h is here given by $\frac{2}{m-1}$.

What happens when $\epsilon_r > 1$?

What happens when $\epsilon_r < 1$ (but still positive)?

Compare the results SBP operators 6 and 7 when $\epsilon_r = 5$ and $m = 150$. What are your observations?

What happens when you increase the number of grid intervals m for a given finite difference stencil? Does the numerical approximation improve?

What happens when you increase the order of accuracy for a given resolution (m) , does the numerical approximation improve?

Coupling of two different media in 2D

Consider Maxwell's equation in 2D (1) with the computational setup presented in Figure 2. The computational domain consists of two different media, where the computational domain is split into 4 adjacent blocks. This setup is solved using the finite difference method presented in this course. Let $\mu_1 = 1$ and $\epsilon_1 = 1$ denote the physical parameters in medium 1. In medium 2, the parameters are denoted $\mu_2 = 1$ and ϵ_2. The boundaries of the computational domain is between -2 and 2 in both dimensions. The divergence free initial data is given by $H^{(z)} = \exp\left(-\left(\frac{x-x_0}{r_*}\right)^2 - \left(\frac{y-y_0}{r_*}\right)^2\right)$ where $r_* = 0.1$, $x_0 = -0.5$ and $y_0 = 0.5$. The electric field is initially set to zero.

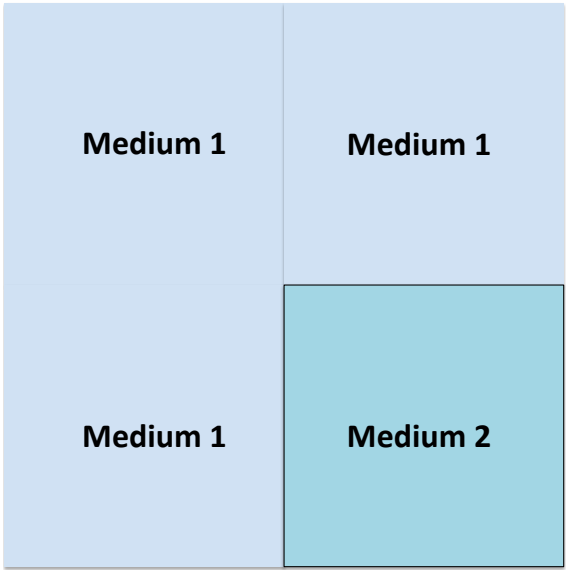

Figure 2: Computational domain in 2D with two different media.

Here we will test 3 different sets of boundary conditions for (1)

1. Specify the tangential electric field (i.e., either $E^{(x)}$ or $E^{(y)}$).
2. Specify the magnetic field $H^{(z)}$.
3. Specify non-reflecting boundary conditions (particular linear combinations of electric and magnetic fields).

UPPSALA
UNIVERSITET

Start by typing *Maxwell_2D_Interface_SC_PDE* in the Matlab window. The numerical solution is integrated in time to $t = 4$ using the fourth order accurate Runge-Kutta method and time-step $k = \frac{h}{10}$ to make the time-integration errors negligible. You will have to specify the name of the avi-file of the generated movie, the number of grid intervals m (i.e, the resolution in space) and the order of accuracy of the SBP finite difference operator. You will then chose between the 3 types of boundary conditions listed above. You will also specify the value for $\epsilon_2 > 0$ in the second medium (in medium 1 $\epsilon_1 = 1$). If $\epsilon_2 > 1$ means that the speed of light is slower in medium 2 (compared to medium 1).

In the numerical tests it is recommended to use at least $m = 100$ (in each dimension and block, which translates to $4 \times m^2$ grid-points in total), but not more than roughly $m = 200$ (since you might run out of memory). The grid-spacing h is here given by $\frac{2}{m-1}$.

What happens to the solution when you chose non-reflecting boundary conditions?

What happens when $\epsilon_2 > 1$?

What happens when $\epsilon_2 < 1$ (but still positive)?