Institut für Strömungsmechanik und Umweltphysik im Bauwesen

Assignment 2

1D transient mass transport equation - analytical and numerical solutions (by using finite difference method)

Date of issue: 02.05.2024

Date of submission: 05.06.2024, at 13:00

Information

- working groups of 2-3 people
- all codes and plots have to be submitted via e-mail to krishna@hydromech.uni-hannover.de
- the present document has to be signed digitally
- codes have to have headers stating the date, names and matriculation numbers of all participants
- tasks marked with a * are optional
- add comments to your codes (at least one comment for each individual information)
- all plots have to have labeled axes and legends
- always use the given variables and functions' names

Declaration of Independence

I hereby declare that the present assignment was worked on independently without help from a third party. No codes or other results were taken from other homework.

Surname	First name	Matriculation number	Signiture
Tersteegen	Marie	10061302	M Tersteegen
Duvvuru	Lokesh	10063226	D Lokesh
Toumi	Younes Abdeldjalil	10064473	[illegible]

1 Solute transport in a river

A river (see Figure 1) with a mean flow velocity of $v = 0.1$m/s flows through the landscape. Turbulent mixing in the water is expressed with the dispersion coefficient $D = 0.01\text{m}^2/\text{s}$.

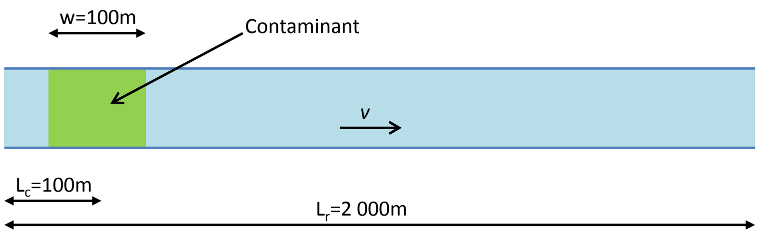


Figure 1: Contaminant in the river (Bird view)

A contaminant solute is injected into the river. The solute is spread instantaneously over the whole width and depth of the river. The concentration of the solute can be initially described as a rectangular pulse with a width of $w = 100$m. Therefore, the initial condition reads

$$c(t=0,x) = \begin{cases} 0 & \text{if} \quad x < L_c - \frac{w}{2} \\ c_{ini} & \text{if} \quad L_c - \frac{w}{2} \leq x \leq L_c + \frac{w}{2} \\ 0 & \text{if} \quad x > L_c + \frac{w}{2} \end{cases} \tag{1}$$

where $c\,(mg/m^3)$ is the solute concentration in the river which is a function of space and time, and $c_{ini}\,(mg/m^3)$ is the initial (maximum) concentration, and $L_c\,(m)$ is the coordinate of the center of the pulse.

Solute transport in a river can be modelled with the one-dimensional advection-dispersion equation

$$\frac{\partial c}{\partial t} + v\frac{\partial c}{\partial x} - D\frac{\partial^2 c}{\partial x^2} = 0. \tag{2}$$

2 Analytical solution

The following equation states the analytical solution for the problem described above:

$$c(x,t) = \frac{c_{ini}}{2}\left(\text{erf}\left(\frac{x + \frac{w}{2} - L_c - vt}{\sqrt{4Dt}}\right) - \text{erf}\left(\frac{x - \frac{w}{2} - L_c - vt}{\sqrt{4Dt}}\right)\right) \tag{3}$$

with $erf(\,)$ the error function. You can find the description of the error function by using Matlab command: `help erf` .

1. Write the function `[c] = transient_ana(x,t,c_ini,w,L_c,v,D)`. This function calculates the vector `c` with a given time `t` [s].

   (Tip: The error function in matlab can be called by `erf( )`.)

2. Write the script `river_ana.m`. The script shall plot the analytical solution associates with the following parameters:

   – initial condition $c_{ini} = 1kg/m^3$

– grid size $\Delta x = 10m$

– plot the concentration profiles at t=[0, 6000, 12000] s in one plot

– save the plot as `river_ana.png`

3 Numerical solution

3.1 Spatial discretization

Finite difference method is used for the spatial discretization. Applying the general finite difference scheme, which has been introduced in Assignment 1, to discretize equation 2 yields

$$\frac{\partial \vec{c}}{\partial t} = \left(\underbrace{\frac{D}{\Delta x^2}\mathbf{K} - \frac{v}{\Delta x}\left(\alpha \mathbf{P_b} + (\mathbf{1} - \alpha)\,\mathbf{P_f}\right)}_{\Lambda} \right) \vec{c} = \mathbf{\Lambda} \vec{c}, \tag{4}$$

where the matrices $\mathbf{K}$, $\mathbf{P}_b$ and $\mathbf{P}_f$ are the same as presented in the Assignment 1.

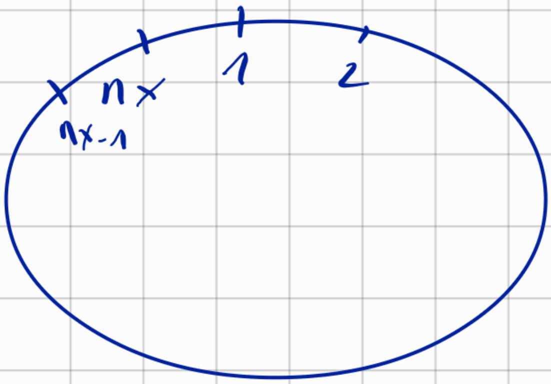
1. Fill in the entries of the matrices $\mathbf{K}$, $\mathbf{P_b}$ and $\mathbf{P_f}$. You can leave the first and last row of these matrices blank.

$$\mathbf{K} = \begin{bmatrix} -2 & 1 & 0 & \cdots & 0 & 1 \\ 1 & -2 & 1 & & & \\ & 1 & -2 & 1 & 0 & \\ & & 1 & -2 & 1 & \\ & 0 & & \ddots & \ddots & \ddots \\ & & & 1 & -2 & 1 \\ 1 & 0 & \cdots & 0 & 1 & -2 \end{bmatrix}, \quad \vec{c} = \begin{Bmatrix} c_1 \\ c_2 \\ \vdots \\ \vdots \\ \vdots \\ c_{nx-1} \\ c_{nx} \end{Bmatrix} \tag{5}$$

$$\mathbf{P_b} = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 & -1 \\ -1 & 1 & 0 & & & \\ & -1 & 1 & 0 & 0 & \\ & & -1 & 1 & 0 & \\ & 0 & & \ddots & \ddots & \ddots \\ & & & -1 & 1 & 0 \\ 0 & \cdots & \cdots & 0 & -1 & 1 \end{bmatrix} \quad \mathbf{P_f} = \begin{bmatrix} -1 & 1 & 0 & \cdots & \cdots & 0 \\ 0 & -1 & 1 & & & \\ & 0 & -1 & 1 & 0 & \\ & & 0 & -1 & 1 & \\ & 0 & & \ddots & \ddots & \ddots \\ & & & 0 & -1 & 1 \\ 1 & 0 & \cdots & \cdots & 0 & -1 \end{bmatrix} \tag{6}$$

$c_i - c_{i-1}$ $\qquad$ $c_{i+1} - c_i$ $\qquad$ $c_{i+1} - 2c_i + c_{i-1}$

P_b $\qquad$ P_f $\qquad$ K



$P_b^1 = [1\ 0 - - - 0\ -1]$ $\qquad$ $P_b^{n_x} = [0 - \dots - 0\ -1\ 1]$

$P_f^1 = [-1\ 1\ 0 \dots - - \quad]$ $\qquad$ $P_f^{n_x} = [1\ 0 \dots - - \ 0\ -1]$

$K^1 = [-2\ 1\ 0 - \dots - +1]$ $\qquad$ $K^{n_x} = [1\ 0 \dots - 0\ 1\ -2]$

3.2 Boundary Conditions

If a domain is very long, the spatial discretization can cause high computational cost. The periodic boundary condition is usually applied for overcoming such a problem.

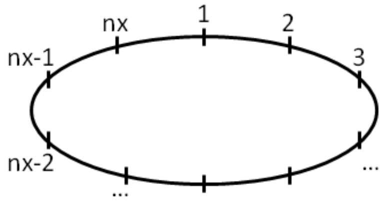


Figure 2: Cyclic boundary conditions

Periodic boundary conditions are created by defining the first node as the neighbor of the last node (see Figure 2).

$$c_i = c_{i+nx}, \qquad \forall\, i \in \mathbb{Z} \tag{7}$$

Now the problem can be solved only using the nodes $i = 1...nx$. Node 1 has the position $x_1 = 0$ and node nx has the position $x_{nx} = 1$.

1. Apply the periodic boundary conditions for the problem, and fill in the first and last row of matrices $\mathbf{K}$, $\mathbf{P_b}$ and $\mathbf{P_f}$.

3.3 Time Discretization

The initial value problem is given with an arbitrary function $f\left(c\left(t\right)\right)$

$$\frac{\mathrm{d}c\left(t\right)}{\mathrm{d}t} = f\left(c\left(t\right)\right) \tag{8}$$

with initial value $c(t_0) = c^0$. We are searching for the value c^1 at timestep t_1. This is done by finding the correct equation to calculate the value c^{n+1} at time t_{n+1} from the known value of c^n at time t_n. Integrating equation (8) from t_n to t_{n+1} yields

$$\int_{t_n}^{t_{n+1}} \frac{\mathrm{d}c}{\mathrm{d}t}\mathrm{d}t = \int_{t_n}^{t_{n+1}} f\left(c\right)\mathrm{d}t \tag{9}$$

The term on the left hand side of the equation (9) can be calculated analytically. The right hand side is integrated numerically by interpolating between $f\left(c^n\right)$ and $f\left(c^{n+1}\right)$. The values at the old and the new time can be weighted by a coefficient θ (see Figure 3). The scheme for the discretization in time can then be written as follows:

$$\frac{c^{n+1} - c^n}{\Delta t} = \left(1-\theta\right) f\left(c^n\right) + \theta f\left(c^{n+1}\right) + \mathcal{O}\left(\Delta t^p\right) \tag{10}$$

$$p = \begin{cases} 2 & \theta = 0.5 \text{ “Trapezoidal”} \\ 1 & 0 \leq \theta \leq 1,\, \theta \neq 0.5 \end{cases} \tag{11}$$

For $\theta = 0$ this is an explicit Euler scheme, that only takes values from the old time step. for $\theta = 0.5$ this is a Crank-Nicolson scheme and for $\theta = 1$ this becomes the implicit Euler scheme which only uses values from the new time step. $\mathcal{O}$ is the truncation error operator.

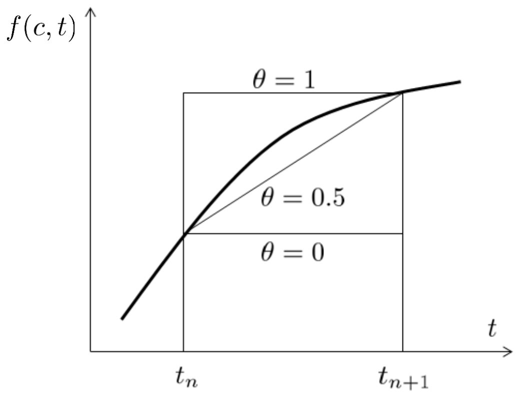


Figure 3: Numerical integration

1. Deduce the time-space finite difference discretization scheme for solving the transient 1D mass transport equation by inserting the spatial discretization (Equation (4)) into the time integration scheme (Equation (10)).

$$\frac{c^{n+1}-c^{n}}{\Delta t} = (1-\theta)\Lambda c^{n} + \theta \Lambda c^{n+1} + \mathcal{O}(\Delta t^{p}) \quad | \cdot \Delta t$$

$$\Leftrightarrow c^{n+1}-c^{n} = \Delta t\left[(1-\theta)\Lambda c^{n} + \theta \Lambda c^{n+1}\right]$$

2. Simplify the equation by inserting, Neumann-Number Ne and Courant-Number CFL. They are defined as

$$Ne = \frac{D\Delta t}{\Delta x^2} \qquad and \qquad CFL = \frac{v\Delta t}{\Delta x} \tag{12}$$

$$\vec{c}^{\,n+1}-\vec{c}^{\,n} = (1-\theta)\left(Ne\,K - CFL\left(\alpha B_b + (1-\alpha) B_f\right)\right)\vec{c}^{\,n} + \theta\left(Ne\,K - CFL\left(\alpha B_b + (1-\alpha) B_f\right)\right)\vec{c}^{\,n+1}$$

3. What do $\theta = 0$, $\theta = 0.5$ and $\theta = 1$ stand for? Write down the order of each scheme.

$\theta = 0 \Rightarrow$ only first term with $\vec{c}^{\,n}$ $\Rightarrow$ explicit Euler scheme $\mathcal{O}(\Delta t)$

$\theta = 0.5 \Rightarrow$ mixed terms $\Rightarrow$ Cranck-Nicolson scheme $\mathcal{O}(\Delta t^2)$

$\theta = 1 \Rightarrow$ only second term with $\vec{c}^{\,n+1}$ $\Rightarrow$ implicit Euler scheme $\mathcal{O}(\Delta t)$

4. Rearrange the equation to

$$\mathbf{A}_l \overrightarrow{c}^{n+1} = \mathbf{A}_r \overrightarrow{c}^{n}. \tag{13}$$

Define matrices $\mathbf{A}_l$ and $\mathbf{A}_r$ with help of the general scheme (4) with matrices $\mathbf{K}$, $\mathbf{P_b}$, $\mathbf{P_f}$ and identity $\mathbf{I}$. Tip: $\overrightarrow{c} = \mathbf{I}\overrightarrow{c}$.

$$\Longleftrightarrow \vec{c}^{\,n+1} - \theta\left(Ne\,K - CFL(\alpha P_b + (1-\alpha)P_f\right)\vec{c}^{\,n+1} = (1-\theta)\left(Ne\,K - CFL(\alpha P_b + (1-\alpha)P_f\right)\cdot\vec{c}^{\,n} + \vec{c}^{\,n}$$

$$\Rightarrow \underbrace{\left[I - \theta(Ne\,K - CFL(\alpha P_b + (1-\alpha)P_f)\right]}_{A_l}\vec{c}^{\,n+1} = \underbrace{\left\{(1-\theta)(Ne\,K - CFL(\alpha P_b + (1-\alpha)P_f) + I\right\}}_{A_r}\vec{c}^{\,n}$$

$$\Longrightarrow \quad A_l := I - \theta\left(Ne\,K - CFL(\alpha P_b + (1-\alpha)P_f\right)$$

$$A_r := (1-\theta)\left(Ne\,K - CFL(\alpha P_b + (1-\alpha)P_f\right) - I$$

5. By using the implicit scheme, the discretized system of equation

$$\mathbf{A}_l \overrightarrow{c}^{n+1} = \overrightarrow{c}^{n} \tag{14}$$

is solved. On the other hand,

$$\overrightarrow{c}^{n+1} = \mathbf{A}_r \overrightarrow{c}^{n}. \tag{15}$$

is solved when the explicit scheme is considered.

What has to be done differently to calculate $\overrightarrow{c}^{n+1}$ from equation (14) than solving equation (15)? Which calculation needs more computational effort?

In order to solve (14) (implicit scheme) we need to invert A_l so it takes more computational effort than (15) (explicit scheme) which does not require inversion of A_r.

6. How many unknown nodes exist for the new time t_{n+1}? What is the dimension of matrix $\mathbf{A}_r$ and $\mathbf{A}_l$? What is the length of vector $\vec{c}$? Is the initial condition a scalar, vector or matrix?

The length of vector $\vec{c}$ is nx.

The dimensions of A_r and A_l are $nx \cdot nx$.

For new time all nodes have to be calculated from t_n, therefore t_{n+1} has nx unknown nodes

The initial condition $\vec{c}(t_0) = \vec{c}^0$ is a vector which contains information of the state of the system (at every node) at the start time t_0

3.4 Implementation in Matlab

1. Complete the flowchart in Figure (4) for solving the time dependent transport equation with labels from Table (1). Read the description of the following exercise and mark in your flowchart if the process is part of the function or the script. Use your decision when writing the code.

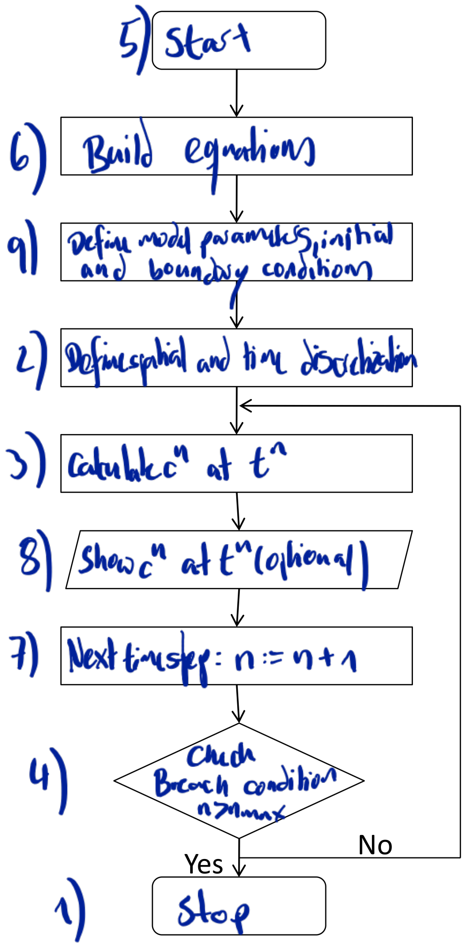


Figure 4: Flowchart

1	Stop
2	Define spatial and time discretization
3	Calculate c^n at time step t^n
4	Check break condition $n \geq n_{max}$
5	Start
6	Build equations
7	Next timestep $n := n + 1$
8	Show c^n at time step t^n (optional)
9	Define model parameters, initial and boundary conditions

Table 1: Parts of flowchart

2. Write the function `[c tend] = transient_cyc(c0, x, dt, CFL, Ne, Nt, alpha, theta, Nplot)` which calculates the numerical solution of Eq. ((2)) with periodic boundary conditions using the discretization scheme derived earlier.

With input values...	c0	vector containing initial condition
	x	vector containing the coordinates of $\vec{c}$
	dt	time step size
	Ne	Neumann number
	CFL	Courant number
	Nt	number of time steps
	alpha	parameter defining the spatial discretization
	theta	parameter defining the time discretization
	Ntplot	plotting interval Ntplot = 1: plot every time step Ntplot = 10 : plot first time step and then every tenth

With output values...	c	approximate solution $\vec{c}$ after time step `Nt`
	tend	time of last time step ($t_{end} = dt * Nt$)

Tips:

- Generate the matrices $\mathbf{K}$, $\mathbf{P_b}$ and $\mathbf{P_f}$ with your function `tridiagcyc`.
- `length`, `numel` and `size` are functions that return the size of vectors and matrices. Use one of these commands to determine the size of the matrices from the initial conditions.

3. Write the script `transient_river.m` that compares the analytical and the numerical solution using your function `transient_cyc()`. Follow the instructions below:

- use the function `init` (on Stud.IP) to calculate the initial condition for your numerical solution. Call the function by writing `init(2)` to use a uniform initial condition as described in equation 1. It also gives you vector x.
- calculate the numerical solution dependent on `Nt`, then calculate the analytical solution at `tend`
- calculate flow velocity v and dispersion coefficient D from CFL and Ne respectively
- plot the analytical solution versus the numerical solution and the initial condition over x

4. Use your script `transient_river.m` with the following settings:

Numerical		**Analytical**	
c0	init(2)	cini	1
CFL	0.5	w	0.5
alpha	1	lc	0.5
theta	1		
Ne	0.1		
dt	1		

Firstly, create a plot for Nt = 3 and save it as `transientNt3.png`, then create a plot for Nt = 30 and save it as `transientNt30.png`. (Note: Only save the last time-step plot.)

5. * Create the script `transient_river2.m` to compare the numerical solutions by applying different combinations of alpha and CFL numbers. Base your script on `transient_river.m` and follow the instructions below:

   - set $\theta = 1$ and $Ne = 0.1$ and $Nt = 200$
   - loop over $\alpha = (0, 0.5, 1)$ and $CFL = (0.5, 1, 10)$
   - take all other parameters from the previous task
   - plot the solution for each combination in a new subplot
   - save the plot as `transient_river2.png`

Based on transient_river2.m we created investigation.m for this exercise and the following

3.4.1 Advection and implicit time integration

Use your script `transient_river.m` or your script `transient_river2.m` for the following investigation. Choose θ such that implicit time integration is used and set $Ne = 0.0$.

1. Investigate the numerical solution for $\alpha = 0$, $\alpha = 0.5$ and $\alpha = 1$ as well as for $CFL = 0.5$, $CFL = 1.0$ and $CFL = 11.0$. Mark the parameter combinations in the following table that are unstable. Which Courant number shows the strongest/weakest numerical diffusion?

we used Nt=30 for this investigation

CFL	$\alpha = 0$	$\alpha = 0.5$	$\alpha = 1$
0.5	unstable	unstable	stable
1	stable	unstable	stable
11	stable	unstable	stable

Courant number 0.5 shows the weakest numerical diffusion and courant number 11 show the strongest numerical diffusion.

2. How can you tell if a scheme is stable or unstable? How can you recognize numerical diffusion? Which scheme for the spatial discretization leads to the best results when using implicit time integration? Which CFL number leads to the smallest/strongest numerical diffusion? Is implicit time integration always stable? When not?

- A scheme is unstable if it exibits oscillations not present in the true solution. Otherwise it is stable.
- Numerical diffusion can be recognized by looking at edges. If numerical diffusion happens, edges become smooth.
- Best results when using implicit time integration are achieved with backward scheme ($\alpha=1$)!

  Numerical diffusion: weakest for CFL=0.5, strongest CFL=11

- Implicit time integration is not always stable. It is not stable for some CFL numbers and not when using central difference scheme for spatial discretization

3.4.2 Advection and explicit time integration

also investigation.m

Use your script `transient_river.m` or your script `transient_river2.m` for the following investigation. Choose θ such that explicit time integration is used and set $Ne = 0.0$.

1. Investigate the numerical solution for $\alpha = 0$, $\alpha = 0.5$ and $\alpha = 1$ as well as for $CFL = 0.5$, $CFL = 1.0$ and $CFL = 11.0$. Mark the parameter combinations in the following table that are unstable. Which Courant number shows the strongest/weakest numerical diffusion?

CFL	$\alpha = 0$	$\alpha = 0.5$	$\alpha = 1$
0.5	unstable	unstable	stable
1	unstable	unstable	stable
11	unstable	unstable	unstable

$CFL_{strongest} = 0.5$

$CFL_{weakest} = 1$

others are unstable

2. Which scheme for the spatial discretization leads to the best results when using explicit time integration? Which CFL number leads to the smallest/strongest numerical diffusion? Is explicit time integration always stable? When not?

- All schemes have problems, but $\alpha = 1$ (backwards schemes) works best.
- Num. diffusion strongest for CFL = 0.5, weakest for CFL = 1
- explicit time integration is only stable for $\alpha = 1$ with CFL = 0.5 and CFL = 1

3. Is the numerical diffusion smaller when using implicit than explicit time integration?

When looking at only the stable solutions (CFL = 0.5 and CFL = 1 for $\alpha = 1$), numerical diffusion is smaller when using explicit time integration

3.4.3 Advection and diffusion

From here on we use investigation2.m based on transient_river2.m

1. Choose backward differences for the spatial discretization and explicit time integration. Test the solution for CFL = [-1,0,1] and Ne = [0,0.5, 1]. Use the following table to mark the unstable parameter combinations.

CFL	$Ne = 0$	$Ne = 0.5$	$Ne = 1$
-1	unstable	unstable	stable
0	stable	stable	unstable
1	stable	unstable	unstable

2. Choose central differences for the spatial discretization and explicit time integration. Test the solution for CFL = [-1,0,1] and Ne = [0,0.5, 1]. Use the following table to mark the unstable parameter combinations.

CFL	$Ne = 0$	$Ne = 0.5$	$Ne = 1$
-1	unstable	stable	unstable
0	stable	stable	unstable
1	unstable	stable	unstable

3. Choose backward Euler scheme for the spatial discretization and implicit time integration. Test the solutions corresponding to CFL = [-1,0,1] and Ne = [0,0.5, 1] respectively. Use the following table to mark the unstable parameter combinations.

CFL	$Ne = 0$	$Ne = 0.5$	$Ne = 1$
-1	stable	unstable	stable
0	stable	stable	stable
1	stable	stable	stable

$$\Delta P_{noFilm}/\Delta P_{film} = (\lambda_{film})/\lambda_{noFilm}) \times (e - e_{film})/e \tag{16}$$

2. Please solve this for a nice movie :)