Partial differential equations

Prof.dr.ir. Martin van Sint Annaland, Dr.ir. Ivo Roghair

m.v.sintannaland@tue.nl

Chemical Process Intensification,
Eindhoven University of Technology

Today's outline

Overview

Main question

How to solve parabolic PDEs like:

$$\frac{\partial c}{\partial t} = \mathcal{D}\frac{\partial^2 c}{\partial x^2} - u\frac{\partial c}{\partial x} + R$$

with

$$\begin{aligned} t = 0; 0 \le x \le \ell \quad & \Rightarrow c = c_0 \\ t > 0; x = 0 \quad & \Rightarrow -\mathcal{D}\frac{\partial c}{\partial x} + uc = u_{\text{in}} c_{\text{in}} \\ t > 0; x = \ell \quad & \Rightarrow \frac{\partial c}{\partial x} = 0 \end{aligned}$$

accurately and efficiently?

What is a PDE?

Partial differential equation

An equation containing a function and their derivatives to multiple independent variables.

Order of PDE

The highest derivative appearing in the PDE

General second order ODE:

$$A\frac{\partial^2 f}{\partial x^2} + B\frac{\partial^2 f}{\partial x \partial y} + C\frac{\partial^2 f}{\partial y^2} + D\frac{\partial f}{\partial x} + E\frac{\partial f}{\partial y} + Ff = G$$

- Linear equation: Coefficients $A, B, \ldots, G$ do not depend on x and y.
- Non-linear equation: Coefficients $A, B, \ldots, G$ are a function of x and y.

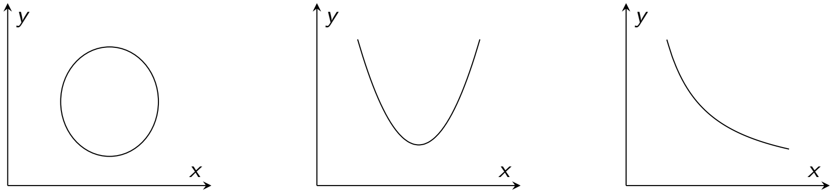
Classification of PDE's

$$A\frac{\partial^2 f}{\partial x^2} + B\frac{\partial^2 f}{\partial x \partial y} + C\frac{\partial^2 f}{\partial y^2} + D\frac{\partial f}{\partial x} + E\frac{\partial f}{\partial y} + Ff = G$$

The discriminant Δ of a quadratic polynomial is computed as (note: only the higher order coefficients are important):

$\Delta = B^2 - 4AC$

- $\Delta < 0 \Rightarrow$ Elliptic equation
  (e.g. Laplace equation for stationary diffusion in 2D)
- $\Delta = 0 \Rightarrow$ Parabolic equation
  (e.g. instationary heat penetration in 1D)
- $\Delta > 0 \Rightarrow$ Hyperbolic equation
  (e.g. wave equation)



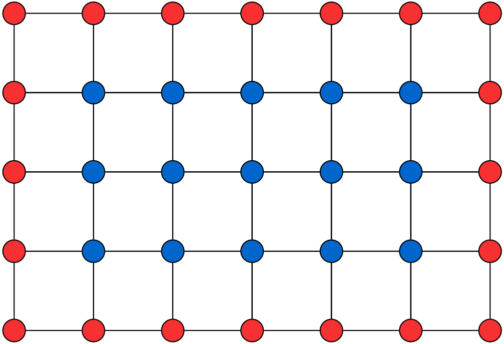
Importance of classification

Different PDE types require different solution techniques because of the difference in range of influence:

- *Characteristics*
  Curves in xy-domain along with signal propagation takes place

- *Domain of dependence of point P*
  points in xy-domain which influence the value of f in point P
- *Range of influence of point P*
  points in xy-domain which are influenced by the value of f in point P

Example elliptic PDE (boundary value problems: BVP)



Grid point at which dependent variable has to be computed

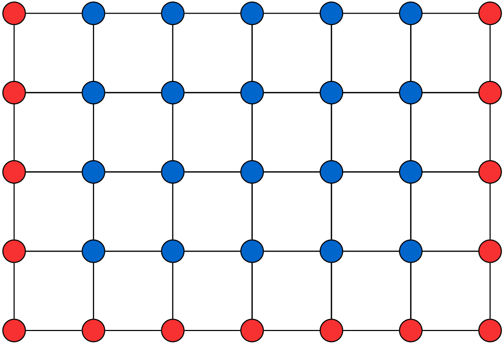
Grid point at which boundary condition is specified

Typical example: Poisson equation

$$\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} = f(x, y)$$

Efficiency (memory requirements, CPU time) of the numerical method is of crucial importance.

Example parabolic PDE (initial value problem: IVP)



Grid point at which dependent variable has to be computed

Grid point at which boundary condition is specified

Typical example: Poisson equation

$$\frac{\partial c}{\partial t} + u\frac{\partial c}{\partial x} = \mathcal{D}\frac{\partial^2 c}{\partial x^2} + R$$

Stability (in numerical sense) of the numerical method is of crucial importance.

Boundary conditions

- Dirichlet or fixed condition: prescribed value of f at boundary

$$f = f_0 \qquad f_0 \text{ is a known function}$$

- Neumann condition: prescribed value of derivative of f at boundary

$$\frac{\partial f}{\partial n} = q \qquad q \text{ is a known function}$$

- Mixed or Robin condition: relation between f and $\frac{\partial f}{\partial n}$ at boundary

$$a\frac{\partial f}{\partial n} + bf = c \qquad a,\ b \text{ and } c \text{ are known functions}$$

Numerical solution method

Finite differences (method of lines, MOL):

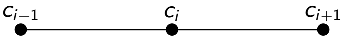
1. Discretize spatial domain in discrete grid points
2. Find suitable approximation for the spatial derivatives
3. Substitute approximations in PDE, which gives a system of ODE's, one for every grid points
4. Advance in time with a suitable ODE solver

Alternative methods: collocation, Galerkin or Finite Element methods

Instationary diffusion equation (Fick's second law)

$$\frac{\partial c}{\partial t} = \mathcal{D}\frac{\partial^2 c}{\partial x^2}, \quad \text{with} \quad \begin{array}{l} t = 0; 0 \leq x \leq \ell \Rightarrow c = c_0 \\ t > 0; x = 0 \Rightarrow c = c_L \\ t > 0; x = \ell \Rightarrow c = c_R \end{array}$$

Second derivative $\dfrac{\partial^2 c}{\partial x^2}$



$$c_{i+1} = c_i + \left.\frac{\partial c}{\partial x}\right|_i \Delta x + \frac{1}{2}\left.\frac{\partial^2 c}{\partial x^2}\right|_i \Delta x^2 + \frac{1}{6}\left.\frac{\partial^3 c}{\partial x^3}\right|_i \Delta x^3 + \ldots$$

$$c_{i-1} = c_i - \left.\frac{\partial c}{\partial x}\right|_i \Delta x + \frac{1}{2}\left.\frac{\partial^2 c}{\partial x^2}\right|_i \Delta x^2 - \frac{1}{6}\left.\frac{\partial^3 c}{\partial x^3}\right|_i \Delta x^3 + \ldots$$

$$c_{i+1} + c_{i-1} = 2c_i + \left.\frac{\partial^2 c}{\partial x^2}\right|_i \Delta x^2 + \mathcal{O}(\Delta x^4)$$

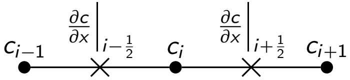
$$\Rightarrow \left.\frac{\partial^2 c}{\partial x^2}\right|_i = \frac{c_{i+1} - 2c_i + c_{i-1}}{\Delta x^2} + \mathcal{O}(\Delta x^2)$$

Due to symmetric discretization: second order (central discretization).

Instationary diffusion equation (Fick's second law)

An alternative discretization:

$$\left.\frac{\partial^2 c}{\partial x^2}\right|_i = \frac{\left.\frac{\partial c}{\partial x}\right|_{i+\frac{1}{2}} - \left.\frac{\partial c}{\partial x}\right|_{i-\frac{1}{2}}}{\Delta x} + \mathcal{O}(\Delta x^2)$$



$$= \frac{\frac{c_{i+1} - c_i}{\Delta x} - \frac{c_i - c_{i-1}}{\Delta x}}{\Delta x} = \frac{c_{i+1} - 2c_i + c_{i-1}}{\Delta x^2}$$

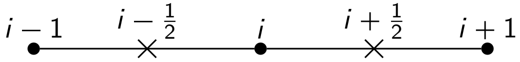
This is convenient for the derivation of $\frac{\partial}{\partial x}\left(\mathcal{D}\frac{\partial c}{\partial x}\right)$:

$$\frac{\partial}{\partial x}\left(\mathcal{D}\frac{\partial c}{\partial x}\right) = \frac{\mathcal{D}_{i+\frac{1}{2}}\left.\frac{\partial c}{\partial x}\right|_{i+\frac{1}{2}} - \mathcal{D}_{i-\frac{1}{2}}\left.\frac{\partial c}{\partial x}\right|_{i-\frac{1}{2}}}{\Delta x} = \frac{\mathcal{D}_{i+\frac{1}{2}}\frac{c_{i+1} - c_i}{\Delta x} - \mathcal{D}_{i-\frac{1}{2}}\frac{c_i - c_{i-1}}{\Delta x}}{\Delta x}$$

$$= \frac{\mathcal{D}_{i+\frac{1}{2}} c_{i+1} - \left(\mathcal{D}_{i+\frac{1}{2}} + \mathcal{D}_{i-\frac{1}{2}}\right) c_i + \mathcal{D}_{i-\frac{1}{2}} c_{i-1}}{(\Delta x)^2}$$

Instationary diffusion equation (Fick's second law)

$$\frac{\partial^2 f}{\partial x^2}$$



$$f_{i+\frac{1}{2}} = f_i + \frac{1}{2}\Delta x \left.\frac{\partial f}{\partial x}\right|_i \Delta x + \frac{1}{2}\left(\frac{1}{2}\Delta x\right)^2 \left.\frac{\partial^2 f}{\partial x^2}\right|_i + \mathcal{O}(\Delta x^3)$$

$$f_{i-\frac{1}{2}} = f_i - \frac{1}{2}\Delta x \left.\frac{\partial f}{\partial x}\right|_i \Delta x + \frac{1}{2}\left(\frac{1}{2}\Delta x\right)^2 \left.\frac{\partial^2 f}{\partial x^2}\right|_i + \mathcal{O}(\Delta x^3)$$

$$f_{i+\frac{1}{2}} - f_{i-\frac{1}{2}} = \Delta x \frac{\partial f}{\partial x} + \mathcal{O}(\Delta x^3)$$

$$\Rightarrow \left.\frac{\partial f}{\partial x}\right|_i = \frac{f_{i+\frac{1}{2}} - f_{i-\frac{1}{2}}}{\Delta x} + \mathcal{O}(\Delta x^2)$$

Symmetric discretization yields second order!

Instationary diffusion equation: spatial discretization

Substitution of spatial derivatives yields:

$$\frac{dc_i}{dt} = \mathcal{D}\frac{c_{i-1} - 2c_i + c_{i+1}}{\Delta x^2} \quad \text{for } i = 0, \ldots, N$$

For example, using 6 (ridiculously low number!) grid points:

c_0 c_1 c_2 c_3 c_4 c_5

$$c_0 = c_L$$

$$\frac{dc_1}{dt} = \mathcal{D}\frac{c_0 - 2c_1 + c_2}{\Delta x^2}$$

$$\frac{dc_2}{dt} = \mathcal{D}\frac{c_1 - 2c_2 + c_3}{\Delta x^2}$$

$$\frac{dc_3}{dt} = \mathcal{D}\frac{c_2 - 2c_3 + c_4}{\Delta x^2}$$

$$\frac{dc_4}{dt} = \mathcal{D}\frac{c_3 - 2c_4 + c_5}{\Delta x^2}$$

$$c_5 = c_R$$

Instationary diffusion equation: boundary conditions

Two options:

1. Keep boundary conditions as additional equations:

$$c_0 = c_L, \frac{dc_1}{dt} = \mathcal{D}\frac{c_0 - 2c_1 + c_2}{\Delta x^2}, \frac{dc_2}{dt} = \mathcal{D}\frac{c_1 - 2c_2 + c_3}{\Delta x^2},$$
$$\frac{dc_3}{dt} = \mathcal{D}\frac{c_2 - 2c_3 + c_4}{\Delta x^2}, \frac{dc_4}{dt} = \mathcal{D}\frac{c_3 - 2c_4 + c_5}{\Delta x^2}, c_5 = c_R$$

2. Substitute boundary conditions to reduce number of equations:

$$\frac{dc_1}{dt} = \mathcal{D}\frac{c_L - 2c_1 + c_2}{\Delta x^2}, \frac{dc_2}{dt} = \mathcal{D}\frac{c_1 - 2c_2 + c_3}{\Delta x^2},$$
$$\frac{dc_3}{dt} = \mathcal{D}\frac{c_2 - 2c_3 + c_4}{\Delta x^2}, \frac{dc_4}{dt} = \mathcal{D}\frac{c_3 - 2c_4 + c_R}{\Delta x^2}$$

Instationary diffusion equation: temporal discretization

$$\frac{dc_i}{dt} = \mathcal{D}\frac{c_{i-1} - 2c_i + c_{i+1}}{\Delta x^2}$$

Time discretization: forward Euler (explicit)

$$\frac{c_i^{n+1} - c_i^{n}}{\Delta t} = \mathcal{D}\frac{c_{i-1}^{n} - 2c_i^{n} + c_{i+1}^{n}}{\Delta x^2}$$

$$\Rightarrow c_i^{n+1} = Foc_{i-1}^n + (1-2Fo)c_i^n + Foc_{i+1}^n \quad \text{with } Fo = \frac{\mathcal{D}\Delta t}{\Delta x^2}$$

Straightforward updating (explicit equation), simple to implement in a program but stability constraint $Fo = \frac{\mathcal{D}\Delta t}{\Delta x^2} < \frac{1}{2}$!

Small $\Delta x \Rightarrow$ small $\Delta t \Rightarrow$ patience required ☹

Instationary diffusion equation: temporal discretization

$$\frac{dc_i}{dt} = \mathcal{D}\frac{c_{i-1} - 2c_i + c_{i+1}}{\Delta x^2}$$

Time discretization: backward Euler (implicit)

$$\frac{c_i^{n+1} - c_i^{n}}{\Delta t} = \mathcal{D}\frac{c_{i-1}^{n+1} - 2c_i^{n+1} + c_{i+1}^{n+1}}{\Delta x^2}$$

$$\Rightarrow -Fo c_{i-1}^{n+1} + (1+2Fo) c_i^{n+1} - Fo c_{i+1}^{n+1} = c_i^n \quad \text{with } Fo = \frac{\mathcal{D}\Delta t}{\Delta x^2}$$

Requires the solution of a system of linear equations, but no stability constraints!

Note: extension to higher order schemes (with time step adaptation) straightforward. Often second or third order optimal, because for each Euler-like step in the additional order an often large system needs to be solved (not treated in this course).

Solving the instationary diffusion equation: example

Solve the diffusion problem using explicit discretization:

$$\frac{\partial c_i}{\partial t} = \mathcal{D}\frac{\partial^2 c}{\partial x^2} \quad \text{with} \quad \begin{array}{l} 0 \leq x \leq \delta,\ \delta = 5 \cdot 10^{-3}\ \mathrm{m} \\ \delta/\Delta x = 100 \text{ grid cells} \\ \mathcal{D} = 1 \cdot 10^{-8}\ \mathrm{m^2\,s^{-1}} \\ t_{\mathrm{end}} = 5000\ \mathrm{s} \\ c_{\mathrm{L}} = 1\ \mathrm{mol\,m^{-3}}\ \ c_{\mathrm{R}} = 0\ \mathrm{mol\,m^{-3}} \end{array}$$

$$c_i^{n+1} = Fo c_{i-1}^n + (1 - 2Fo)c_i^n + Fo c_{i+1}^n \quad \text{with } Fo = \frac{\mathcal{D}\Delta t}{\Delta x^2}$$

1. Initialise variables
2. Compute time step so that $Fo \leq \frac{1}{2} \Rightarrow \Delta t = 0.125\mathrm{s}$!
3. Compute 40000 time steps times 100 grid nodes!
4. Store solution

Solving the diffusion equation implicitly

Linear system $A\boldsymbol{x} = \boldsymbol{b}$ from $-Foc_{i-1}^{n+1} + (1+2Fo)c_i^{n+1} - Foc_{i+1}^{n+1} = c_i^n$

$$\begin{pmatrix} 1 & 0 & 0 & 0 & \cdots & 0 \\ -Fo & (1+2Fo) & -Fo & 0 & \cdots & 0 \\ 0 & -Fo & (1+2Fo) & -Fo & \cdots & 0 \\ 0 & 0 & -Fo & (1+2Fo) & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & 1 \end{pmatrix} \cdot \begin{pmatrix} c_0^{n+1} \\ c_1^{n+1} \\ c_2^{n+1} \\ c_3^{n+1} \\ \vdots \\ c_m^{n+1} \end{pmatrix} = \begin{pmatrix} c_0^n \\ c_1^n \\ c_2^n \\ c_3^n \\ \vdots \\ c_m^n \end{pmatrix}$$

$1 \times c_0^{n+1} = c_0^n$ (boundary condition)

$-Foc_0^{n+1} + (1+2Fo)c_1^{n+1} - Foc_2^{n+1} = c_1^n$

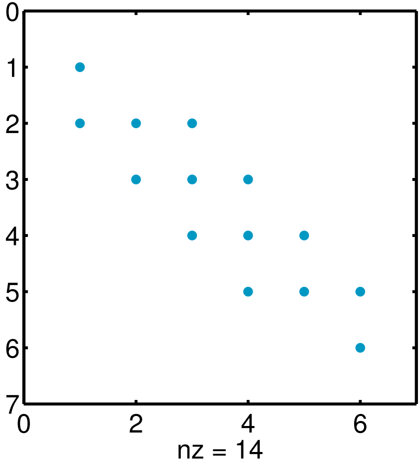
$-Foc_1^{n+1} + (1+2Fo)c_2^{n+1} - Foc_3^{n+1} = c_2^n$

$-Foc_2^{n+1} + (1+2Fo)c_3^{n+1} - Foc_4^{n+1} = c_3^n$

$1 \times c_m^{n+1} = c_m^n$ (boundary condition)

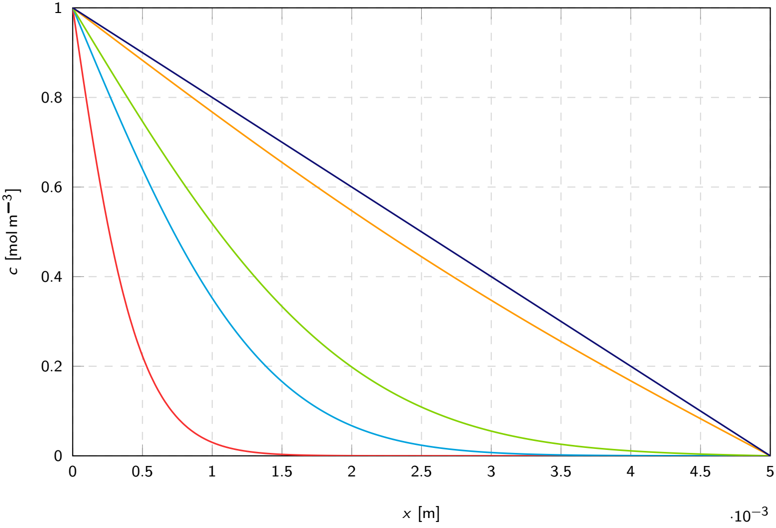
Solving the diffusion equation implicitly in Matlab

The command `spy(A)` shows a figure with the non-zero positions.



Solving the diffusion equation implicitly in Matlab

Plotting the solution at $t = \{12.5, 62.5, 125, 625, 5000\}$ s.



About explicit vs. implicit solutions

- Explicit solution:
  - Easy to implement
  - Very small time steps required.
  - This problem took about 0.5 s.
- Implicit solution:
  - Harder to implement, needs sparse matrix solver
  - No stability constraint
  - This problem took about 0.05 s
- The difference will become much larger for systems with e.g. more grid nodes!

Extension with non-linear source terms

$$\frac{\partial c}{\partial t} = \mathcal{D}\frac{\partial^2 c}{\partial x^2} + R(c) \quad \text{with} \quad \begin{array}{l} t = 0; 0 \leq x \leq \ell \Rightarrow c = c_0 \\ t > 0; x = 0 \Rightarrow c = c_L \\ t > 0; x = \ell \Rightarrow c = c_R \end{array}$$

- Forward Euler (explicit): simply add to right-hand side

$$\frac{c_i^{n+1} - c_i^n}{\Delta t} = \mathcal{D}\frac{c_{i-1}^{n} - 2c_i^{n} + c_{i+1}^{n}}{\Delta x^2} + R(c_i^{n})$$

$$\Rightarrow c_i^{n+1} = Foc_{i-1}^n + (1-2Fo)c_i^n + Foc_{i+1}^n + R_i^n \Delta t$$

- Backward Euler (implicit): linearization required

$$R(c_i^{n+1}) = R(c_i^n) + \left.\frac{dR}{dc}\right|_i^n (c_i^{n+1} - c_i^n)$$

$$\frac{c_i^{n+1} - c_i^n}{\Delta t} = \mathcal{D}\frac{c_{i-1}^{n+1} - 2c_i^{n+1} + c_{i+1}^{n+1}}{\Delta x^2} + R(c_i^{n+1})$$

$$\Rightarrow -Foc_{i-1}^{n+1} + (1 + 2Fo - \left.\frac{dR}{dc}\right|_i^n \Delta t)c_i^{n+1} - Foc_{i+1}^{n+1} = c_i^n + \left(R_i^n - \left.\frac{dR}{dc}\right|_i^n c_i^n\right)\Delta t$$

Extension with convection terms

$$\frac{\partial c}{\partial t} = \mathcal{D}\frac{\partial^2 c}{\partial x^2} - u\frac{\partial c}{\partial x} + R$$

Discretization of first derivative $\frac{dc}{dx}$,
looks simple but is numerical headache!

Central discretization:

$$\frac{dc}{dx} = \frac{c_{i+1} - c_{i-1}}{2\Delta x}$$

$\Rightarrow$ simple and easy, too bad it doesn't work: yields unstable solutions if convection dominated.

Central difference scheme of 1st derivative

Unsteady convection:

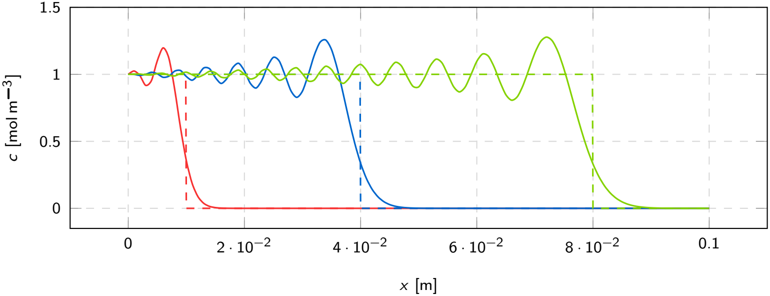
$$\frac{\partial c}{\partial t} = -u\frac{\partial c}{\partial x}$$

Central difference for first derivative:

$$\frac{dc}{dx} = \frac{c_{i+1} - c_{i-1}}{2\Delta x}$$

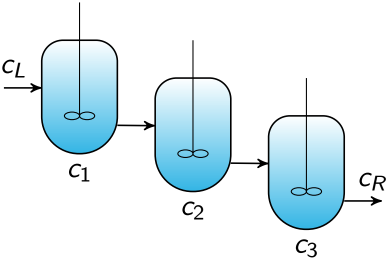
Forward Euler discretization of temporal and spatial domain:

$$\frac{c_i^{n+1} - c_i^n}{\Delta t} = -u\frac{c_{i+1} - c_{i-1}}{2\Delta x} \Rightarrow c_i^{n+1} = c_i^n - u\frac{c_{i+1}^n - c_{i-1}^n}{2\Delta x}\Delta t$$



Extension with convection terms

Solution: upwind discretization, like CSTR's in series:



First order upwind: $$-u\left.\frac{dc}{dx}\right|_i = \begin{cases} -u\dfrac{c_i - c_{i-1}}{\Delta x} & \text{if } u \geq 0 \\[2ex] -u\dfrac{c_{i+1} - c_i}{\Delta x} & \text{if } u < 0 \end{cases}$$

Stable if $Co = \frac{u\Delta t}{\Delta x} < 1$ (with Co the Courant number). However, only 1[st] order accurate (large smearing of concentration fronts). Higher order upwind requires TVD schemes (trick of the trade)...

First order upwind scheme of 1st derivative

Unsteady convection:

$$\frac{\partial c}{\partial t} = -u\frac{\partial c}{\partial x}$$

Upwind scheme for first derivative:

$$-u\frac{dc}{dx}\bigg|_i = \begin{cases} -u\dfrac{c_i - c_{i-1}}{\Delta x} & \text{if } u \geq 0 \\[2ex] -u\dfrac{c_{i+1} - c_i}{\Delta x} & \text{if } u < 0 \end{cases}$$

Forward Euler discretization of temporal and spatial domain:

$$\frac{c_i^{n+1} - c_i^n}{\Delta t} = -u\frac{c_{i+1} - c_{i-1}}{2\Delta x}$$

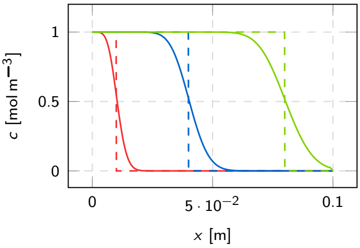
$$\Rightarrow c_i^{n+1} = \begin{cases} c_i^n - u\dfrac{c_i - c_{i-1}}{\Delta x} & \text{if } u \geq 0 \\[2ex] c_i^n - u\dfrac{c_{i+1} - c_i}{\Delta x} & \text{if } u < 0 \end{cases}$$

Upwind scheme: example

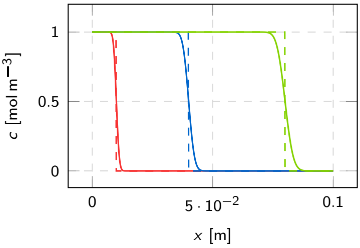
Unsteady convection through a pipe:

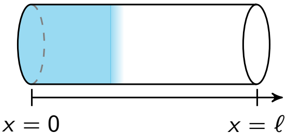
$$\frac{\partial c}{\partial t} = -u\frac{\partial c}{\partial x} \quad \text{with} \quad u = 0.1\,\mathrm{m\,s^{-1}} \Rightarrow c_i^{n+1} = c_i^n - u\frac{c_i - c_{i-1}}{\Delta x}\Delta t$$

Using 100 grid cells



Using 1000 grid cells





Extension to systems of PDE's

- Explicit methods: straightforward extension
- Implicit methods: yields block-tridiagonal matrix (note ordering of equations: all variables per grid cell)

Extension to 2D or 3D systems

Spatial discretization in 2 directions — different methods available:

- Explicit
- Fully implicit
  - 1D gives tri-diagonal matrix
  - 2D gives penta-diagonal matrix
  - 3D gives hepta-diagonal matrix

  Use of dedicated matrix solvers (e.g. ICCG, multigrid, ...)
- Alternating direction implicit (ADI)
  - Per direction implicit, but still overall unconditionally stable

Further extensions for parabolic PDEs

- Higher order temporal discretization (multi-step) with time step adaptation
- Non-uniform grids with automatic grid adaptation
- Higher-order discretization methods, especially higher order TVD (flux delimited) schemes for convective fluxes (e.g. WENO schemes)
- Higher-order finite volume schemes (Riemann solvers)

Summary

- Several classes of PDEs were introduced
  - Elliptic, Parabolic, Hyperbolic PDEs
- Diffusion equation: discretization of temporal and spatial domain was discussed
  - Solutions of the diffusion equation using explicit and implicit methods
  - How to add non-linear source terms
- Convection: upwind vs. central difference schemes