CEGM1000
Modelling, Uncertainty and Data for Engineers

Week 2.1
Numerical methods for PDEs

Marcel Zijlema

TU Delft

Learning Objectives

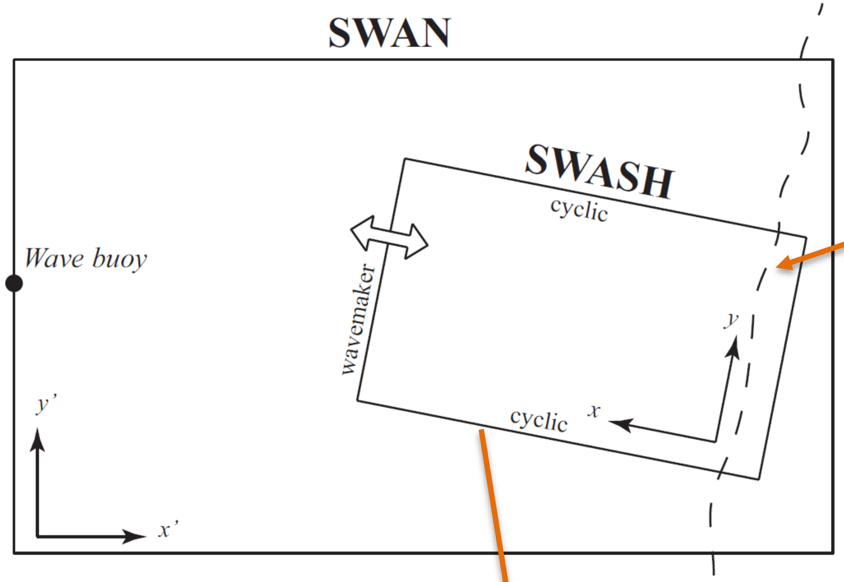
at the end of this lecture, you should be able to

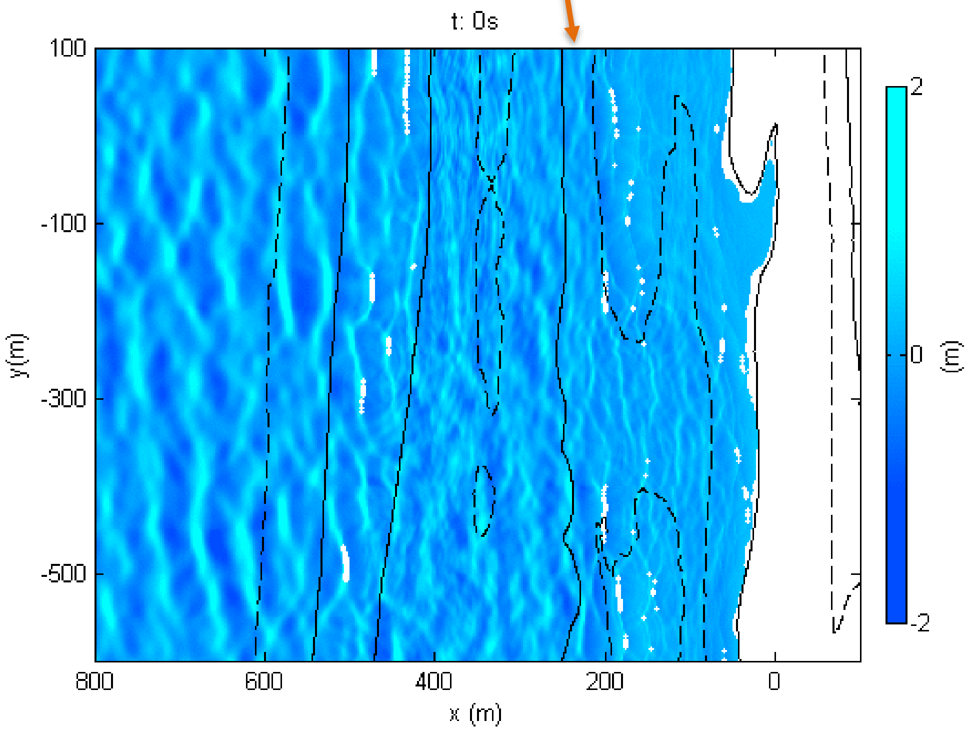
- identify various PDEs based on their properties
- explain the Finite Difference Method
- apply central differences and upwind schemes
- discuss the pros and cons of these numerical schemes with respect to
    - accuracy,
    - stability, and
    - numerical artefacts

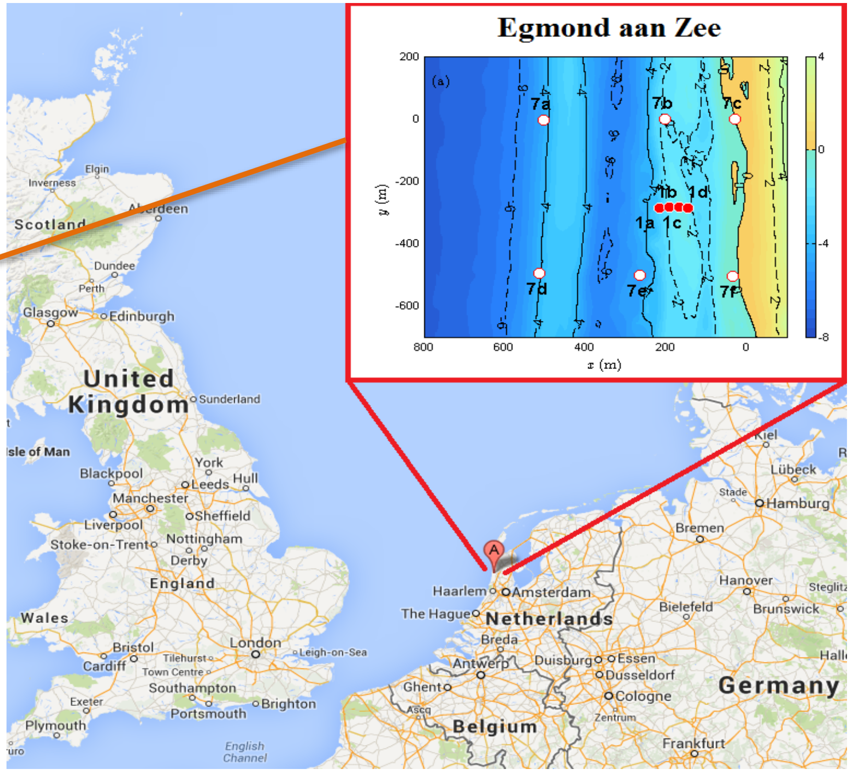
this lecture

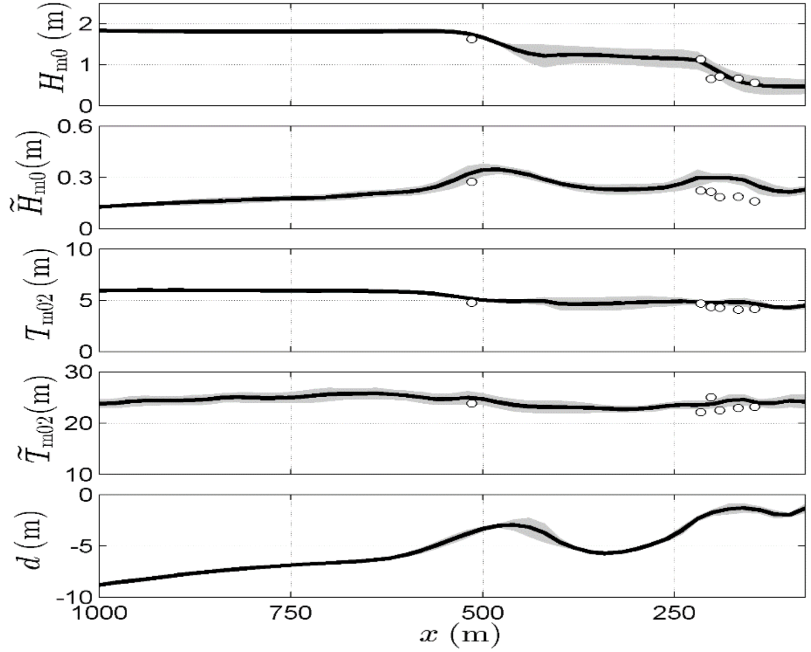
- today's lecture follows directly on from week 2 of Q1 (*Numerical Modelling*)
  - week 1.2 treated numerical methods for ODEs
  - week 2.1 (this week) will treat numerical methods for PDEs
- nature of PDEs
- Finite Difference Method (FDM)
  - central difference scheme – FTCS
  - upwind scheme – FTBS
- consistency, stability and convergence
  - truncation error
- numerical artefacts
  - numerical diffusion
  - wiggles

wave dynamics in a barred coastal region (Rijnsdorp et al., 2015)









Delft's wave models (at gitlab.tudelft.nl)

- SWAN
  (Booij et al, 1999)

  spectral wave model

$$\frac{\partial E}{\partial t} + c_{g,x}\frac{\partial E}{\partial x} + c_{g,y}\frac{\partial E}{\partial y} + c_\theta \frac{\partial E}{\partial \theta} = S(x,y,\theta)$$

energy balance equation
(see CIEM3000-25)

a bunch of partial differential equations (PDEs)

- SWASH
  (Zijlema et al, 2011)

  non-hydrostatic
  wave-flow model

$$\frac{\partial \zeta}{\partial t} + \frac{\partial hu}{\partial x} + \frac{\partial hv}{\partial y} = 0$$

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y} + g\frac{\partial \zeta}{\partial x} + c_f\frac{|u|u}{h} = \nu_h\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right)$$

$$\frac{\partial v}{\partial t} + u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y} + g\frac{\partial \zeta}{\partial y} + c_f\frac{|v|v}{h} = \nu_h\left(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2}\right)$$

shallow water equations
(see modules A and B and
2nd year elective modules)

TU Delft

nature of PDEs

- **elliptic**: steady, the perturbation in one location affects the rest of domain instantaneously

- **parabolic**: unsteady, the information propagates in all directions at an infinite speed

- **hyperbolic**: unsteady, the information propagates along a characteristic at a finite speed

nature of PDEs

- **elliptic**: steady, the perturbation in one location affects the rest of domain instantaneously

Poisson equation: $$\frac{\partial^2 \phi}{\partial x^2} + \frac{\partial^2 \phi}{\partial y^2} = f(x, y)$$ 2D, non-homogeneous, 2nd order linear PDE

- **parabolic**: unsteady, the information propagates in all directions at an infinite speed

diffusion equation: $$\frac{\partial T}{\partial t} = \kappa \frac{\partial^2 T}{\partial x^2}$$ 1D, homogeneous, 2nd order linear PDE

- **hyperbolic**: unsteady, the information propagates along a characteristic at a finite speed

advection equation: $$\frac{\partial c}{\partial t} + u \frac{\partial c}{\partial x} = 0$$ 1D, homogeneous, 1st order linear PDE

***the physical behaviour of PDEs requires a suitable numerical scheme to solve them*!**

TU Delft

Delft's wave models

- SWAN

$$\frac{\partial E}{\partial t}+c_{g,x}\frac{\partial E}{\partial x}+c_{g,y}\frac{\partial E}{\partial y}+c_\theta\frac{\partial E}{\partial \theta}=S(x,y,\theta)$$

3D, non-homogeneous, 1st order nonlinear PDE of hyperbolic type

- SWASH

$$\frac{\partial \zeta}{\partial t}+\frac{\partial hu}{\partial x}+\frac{\partial hv}{\partial y}=0$$

$$\frac{\partial u}{\partial t}+u\frac{\partial u}{\partial x}+v\frac{\partial u}{\partial y}+g\frac{\partial \zeta}{\partial x}+c_f\frac{|u|u}{h}=\nu_h\left(\frac{\partial^2 u}{\partial x^2}+\frac{\partial^2 u}{\partial y^2}\right)$$

$$\frac{\partial v}{\partial t}+u\frac{\partial v}{\partial x}+v\frac{\partial v}{\partial y}+g\frac{\partial \zeta}{\partial y}+c_f\frac{|v|v}{h}=\nu_h\left(\frac{\partial^2 v}{\partial x^2}+\frac{\partial^2 v}{\partial y^2}\right)$$

system of 2D hyperbolic, homogeneous, 2nd order nonlinear PDEs

TU Delft

well-posed problems

- a boundary value problem (BVP) consists of
  - the PDE itself
  - the enclosed spatial domain Ω on which the PDE is required to be satisfied
  - the boundary conditions that the solution must be met at the boundaries of Ω
  - in case of a time-dependent problem, initial condition(s) must be included as well
- a BVP is said to be **well-posed** if
  - a solution to the problem **exists**,
  - the solution is **unique**, and
  - the solution is **not sensitive** to boundary / initial conditions (closely related to **stability** of the PDE)

we will be dealing with the following PDEs this week …

- the diffusion equation
  - today's lecture
  - Programming Assignment 2.1 (Tuesday)

$$\frac{\partial T}{\partial t} = \kappa \frac{\partial^2 T}{\partial x^2}$$

- the advection equation
  - today's lecture
  - Workshop Assignment 2.1 (Wednesday)

$$\frac{\partial c}{\partial t} + u\frac{\partial c}{\partial x} = 0$$

- the advection-diffusion equation
  - today's lecture
  - Group Assignment 2.1 (Friday)

$$\frac{\partial c}{\partial t} + u\frac{\partial c}{\partial x} - \kappa \frac{\partial^2 c}{\partial x^2} = 0$$

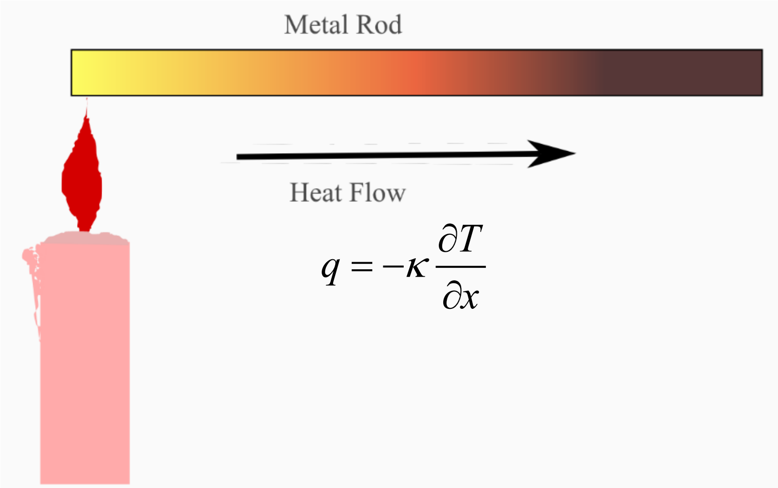
1D, homogeneous,
2nd order linear PDE
of parabolic type

the diffusion equation

- we consider the following equation

$$\frac{\partial T}{\partial t} = \kappa \frac{\partial^2 T}{\partial x^2}$$

- an example of application is the heat transfer in a rod
  - the associated PDE is known as the **heat equation**
  - Fourier's law of thermal conduction: heat flow q is product of thermal diffusivity κ and *negative* temperature gradient (from high temperature to low temperature)

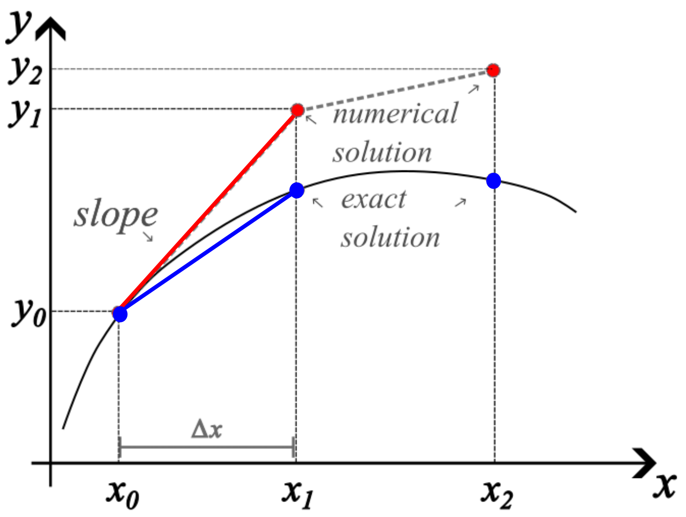


$$\frac{\partial T}{\partial t} + \frac{\partial q}{\partial x} = 0$$

conservation law

numerical approach

- Finite Difference Method (FDM)
  - a derivative of an unknown is approximated by means of finite differences
  - use Taylor series expansion to approximate the derivative to a certain order of accuracy
  - difference between exact derivative and finite difference formula determines the truncation error



forward difference quotient:

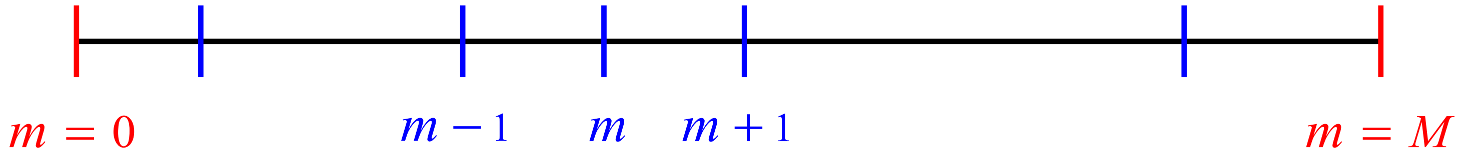
$$\left.\frac{dy}{dx}\right|_{x_0} \approx \frac{y(x_1)-y(x_0)}{x_1-x_0} \approx \frac{y_1-y_0}{\Delta x}$$

$$\boxed{\tau_{\Delta x}} = \left.\frac{dy}{dx}\right|_{x_0} - \frac{y_1-y_0}{\Delta x} = \boxed{O(\Delta x)}$$

TU Delft

semi discretization of diffusion equation

- Method of Lines (MOL)
  - discretize the PDE in <u>space</u> first, leading to a system of ODEs – <u>semi discretization</u>
  - then integrate the system of ODEs over time
- semi discretization by means of forward, backward or central difference scheme
  - second derivatives are always approximated using central differences
- define 1D <u>equidistant grid</u> with *Δx* the mesh size



- the diffusion equation is approximated as follows (ignore boundary points):

$$\frac{\partial T}{\partial t} = \kappa \frac{\partial^2 T}{\partial x^2} \quad \Rightarrow \quad \frac{dT_m}{dt} = \kappa \frac{T_{m+1} - 2T_m + T_{m-1}}{\Delta x^2}, \quad m = 1, 2, \cdots, M-1$$

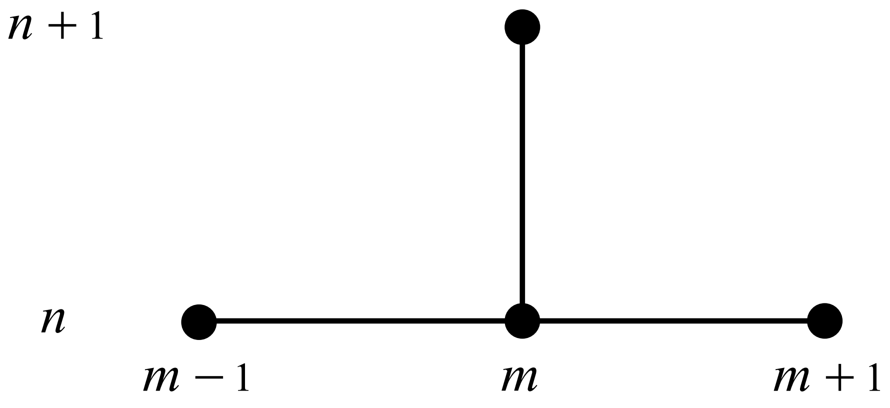
these are inner grid points

time integration

- use the forward Euler scheme to integrate the semi-discrete equation, as follows

$$\frac{dT_m}{dt} = \kappa \frac{T_{m+1} - 2T_m + T_{m-1}}{\Delta x^2} \quad \Rightarrow \quad \boxed{\frac{T_m^{n+1} - T_m^n}{\Delta t} = \kappa \frac{T_{m+1}^n - 2T_m^n + T_{m-1}^n}{\Delta x^2}}$$

  with n the time level and Δt a (fixed) time step (= difference between two successive time instants t_{n+1} and t_n)
- this finite difference scheme is called the Forward in Time and Central in Space (FTCS) scheme
- the FTCS scheme is called **explicit** since the new solution depends only on the solutions at the *preceding* time step
- its stencil displays the links between different unknowns and looks like



TU Delft

Python implementation

- The FTCS *scheme* can be rewritten in the following numerical *recipe*

$$T_m^{n+1} = T_m^n + \kappa \frac{\Delta t}{\Delta x^2}\left(T_{m+1}^n - 2T_m^n + T_{m-1}^n\right)$$

- in turn, this recipe can be implemented as an *algorithm* on a computer by means of a high-level computer language
  - Fortran, C, Matlab, Java, Python
  - ```python
    # loop through time
    for n in range(nt-1):

        # solution in the interior
        for m in range(1,nx-1):
            T[n+1,m] = T[n,m] + kappa * dt/dx/dx * (T[n,m+1]-2*T[n,m]+T[n,m-1])
    ```
    ```

truncation error

exact solution

- the truncation error of the FTCS scheme is given by

$$\tau_{\Delta t,\Delta x} = \frac{T(x_m,t_{n+1})-T(x_m,t_n)}{\Delta t} - \kappa\frac{T(x_{m+1},t_n)-2T(x_m,t_n)+T(x_{m-1},t_n)}{\Delta x^2}$$

- use Taylor expansion series to show that

$$\tau_{\Delta t,\Delta x} = O\left(\Delta t,\Delta x^2\right)$$

note that $x_{m+1} = x_m + \Delta x$ and $t_{n+1} = t_n + \Delta t$

- rule: the solution error, defined as $e_m^n = T(x_m,t_n)-T_m^n$, and the truncation error $\tau_{\Delta t,\Delta x}$ are of the same order
  - when both Δt and Δx are reduced by a factor of 2, the solution error will be reduced by a factor of 8 (= 2 x 4)

TU Delft

consistency and convergence

- a numerical scheme is called **consistent** if and only if

$$\lim_{\Delta x, \Delta t \to 0} \tau_{\Delta t, \Delta x} = 0$$

- a numerical scheme is said to be **convergent** if

$$\lim_{\Delta x, \Delta t \to 0} e_m^n = 0$$

- a basic rule is the following

$$\boxed{\text{consistency} + \text{stability} \rightarrow \text{convergence}}$$

numerical solution is bounded

TU Delft

stability

- a numerical scheme is said to be **stable** if and only if

$$\left| T_m^n \right| < K, \quad \forall m\,, \quad \forall n$$

where Δt and Δx are fixed(!) and K is a constant independent of Δt and Δx

- a common method to analyze the stability of the scheme is based on Fourier decomposition and is known as the Von Neumann stability analysis method
  - this method provides a theoretical basis to proof the correctness of the stability condition
  - another method that can correctly determine the stability of the scheme is the matrix method

difficult and elaborated

- simple but heuristic stability methods are also possible but usually do not prove the correctness of the stability
  - such methods are typically PDE dependent

TU Delft

stability of FTCS

- the true solution of the diffusion equation is always non-negative since
  - the concentration (e.g., heat, salt) or the associated mass is from physical point of view non-negative
  - the initial and boundary conditions are for this reason non-negative
  - the diffusion process is conserving: it does not create nor destroy amount of mass within a given volume – if you start with a non-negative amount of mass, you cannot end up with a negative amount
- requirement: the numerical solution generated by a numerical scheme must be non-negative, that is,

$$T_m^n \geq 0\,, \quad \forall m = 0, \cdots, M\,, \quad \forall n = 0, 1, 2, \cdots$$

- we derive heuristically the following (prototype) stability limit of the FTCS scheme:

$$\boxed{\frac{\kappa\,\Delta t}{\Delta x^2} \leq \frac{1}{2}}$$

correct according to
Von Neumann analysis

- the FTCS scheme is conditionally stable: a choice of Δx will limit the time step!
- in practice: Δt is often too small with respect to accuracy!

the advection equation

- we consider the following equation

$$\frac{\partial c}{\partial t} + u\frac{\partial c}{\partial x} = 0$$

- it describes the propagation of a perturbation (e.g., wave, cloud of pollutant) along the domain $x \in [0,L]$

the advection equation

- we consider the following equation

$$\frac{\partial c}{\partial t} + u \frac{\partial c}{\partial x} = 0$$

- it describes the propagation of a perturbation (e.g., wave, cloud of pollutant) along the domain $x \in [0,L]$

the advection equation

- we consider the following equation

$$\frac{\partial c}{\partial t} + u\frac{\partial c}{\partial x} = 0$$

- it describes the propagation of a perturbation (e.g., wave, cloud of pollutant) along the domain $x \in [0,L]$

the advection equation

- we consider the following equation

$$\frac{\partial c}{\partial t} + u \frac{\partial c}{\partial x} = 0$$

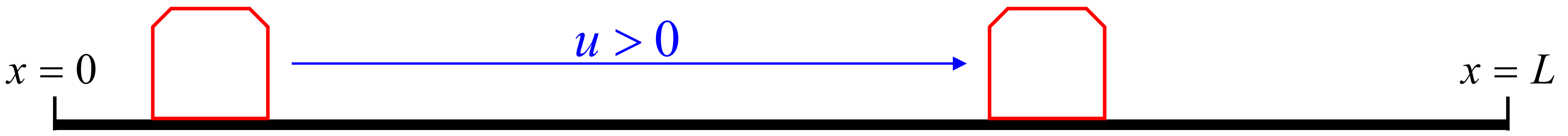
- it describes the propagation of a perturbation (e.g., wave, cloud of pollutant) along the domain $x \in [0,L]$ while the shape of the perturbation is preserved

the advection equation

- we consider the following equation

$$\frac{\partial c}{\partial t} + u\frac{\partial c}{\partial x} = 0$$

- it describes the propagation of a perturbation (e.g., wave, cloud of pollutant) along the domain $x \in [0,L]$ while the shape of the perturbation is preserved



- one initial condition (state of $c(x)$ at $t = 0$ along the entire domain) and one boundary condition at the boundary from which the perturbation is being propagated
  - in case $u > 0$, we impose the following boundary condition: $c(0,t) = f(t)$ (on west side)
  - in case $u < 0$, we impose the following boundary condition: $c(L,t) = f(t)$ (on east side)

discretization of advection equation

- we use <u>backward difference scheme</u> to capture the information upstream properly
  - central differences does not care about the direction of propagation, which leads to instabilities!

to be tested at WS 2.1

- we apply again <u>forward Euler scheme</u>
- the resulting scheme is the Forward in Time, Backward in Space (FTBS) scheme

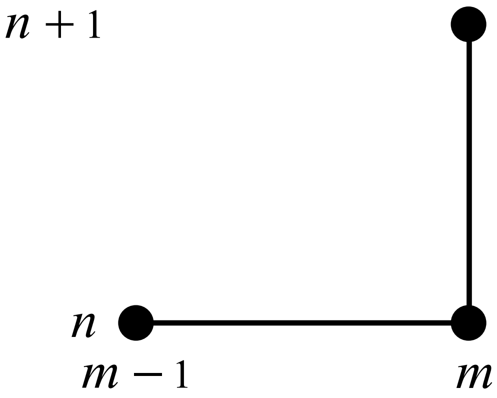
$$\frac{\partial c}{\partial t} + u\frac{\partial c}{\partial x} = 0 \quad \Rightarrow \quad \boxed{\frac{c_m^{n+1} - c_m^n}{\Delta t} + u\frac{c_m^n - c_{m-1}^n}{\Delta x} = 0}$$

here we assume $u > 0$

- the truncation error of the FTBS scheme is given by

$$\tau_{\Delta t,\Delta x} = O(\Delta t, \Delta x)$$

- the FTBS scheme is <u>explicit</u>, and its stencil looks like



discretization of advection equation

- we use <u>backward difference scheme</u> to capture the information upstream properly
  - central differences does not care about the direction of propagation, which leads to instabilities!

to be tested at WS 2.1

- we apply again <u>forward Euler scheme</u>
- the resulting scheme is the Forward in Time, Backward in Space (FTBS) scheme

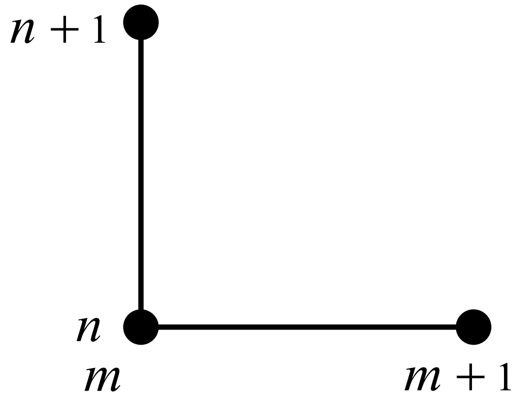
$$\frac{\partial c}{\partial t} + u\frac{\partial c}{\partial x} = 0 \quad \Rightarrow \quad \boxed{\frac{c_m^{n+1} - c_m^n}{\Delta t} + u\frac{c_{m+1}^n - c_m^n}{\Delta x} = 0}$$

here we assume $u < 0$ (!)

- the truncation error of the FTBS scheme is given by

$$\tau_{\Delta t, \Delta x} = O(\Delta t, \Delta x)$$

- the FTBS scheme is <u>explicit</u>, and its stencil looks like



the advection-diffusion equation

- we consider the following equation in domain $0 \leq x \leq 1$

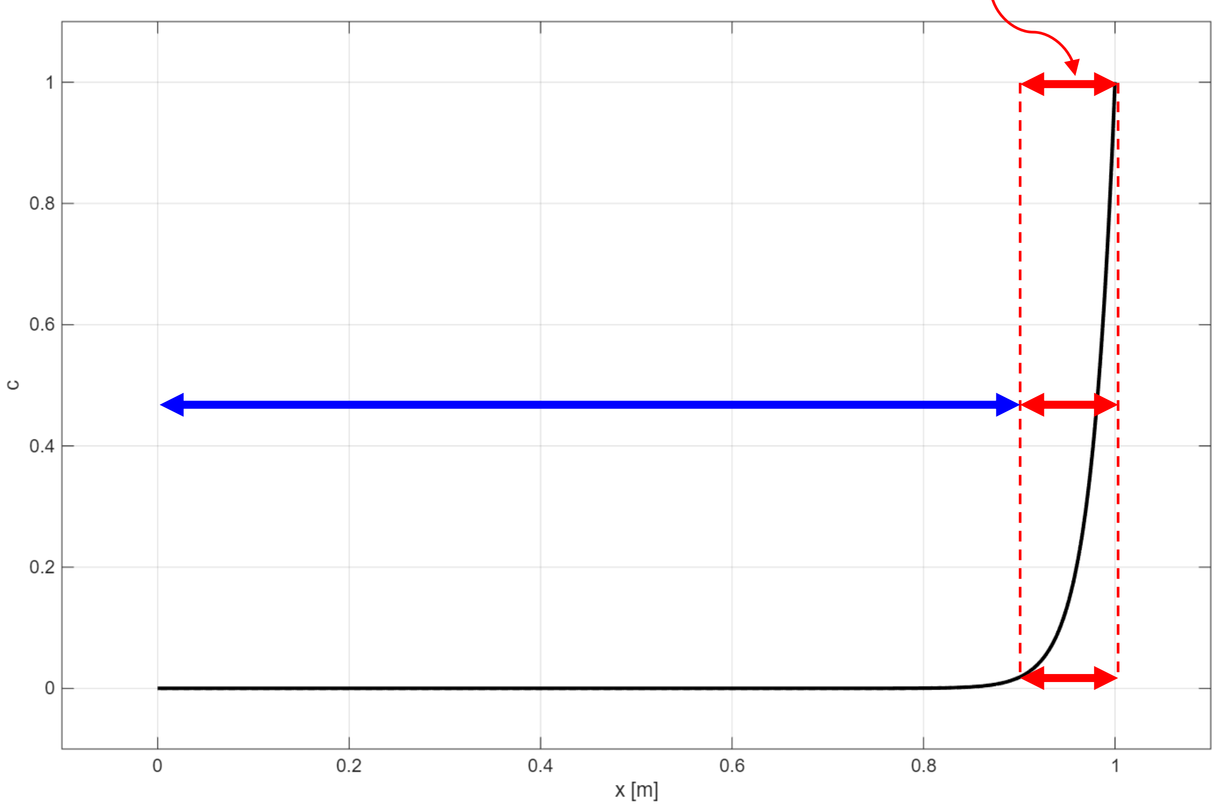
$$\frac{\partial c}{\partial t} + u\frac{\partial c}{\partial x} - \kappa\frac{\partial^2 c}{\partial x^2} = 0$$

- the following boundary conditions are provided
  - $c(0,t) = 0$ at west side, and
  - $c(1,t) = 1$ at east side

time-independent!

- the <u>steady-state solution</u> is of the **boundary layer** type
  - effects of diffusion are confined in a thin layer ("*inner region*") near the east boundary
  - the rest of the domain ("*outer region*") is treated as inviscid
- the larger the κ, the thicker is the boundary layer

$\delta \sim \sqrt{\kappa}$



TU Delft

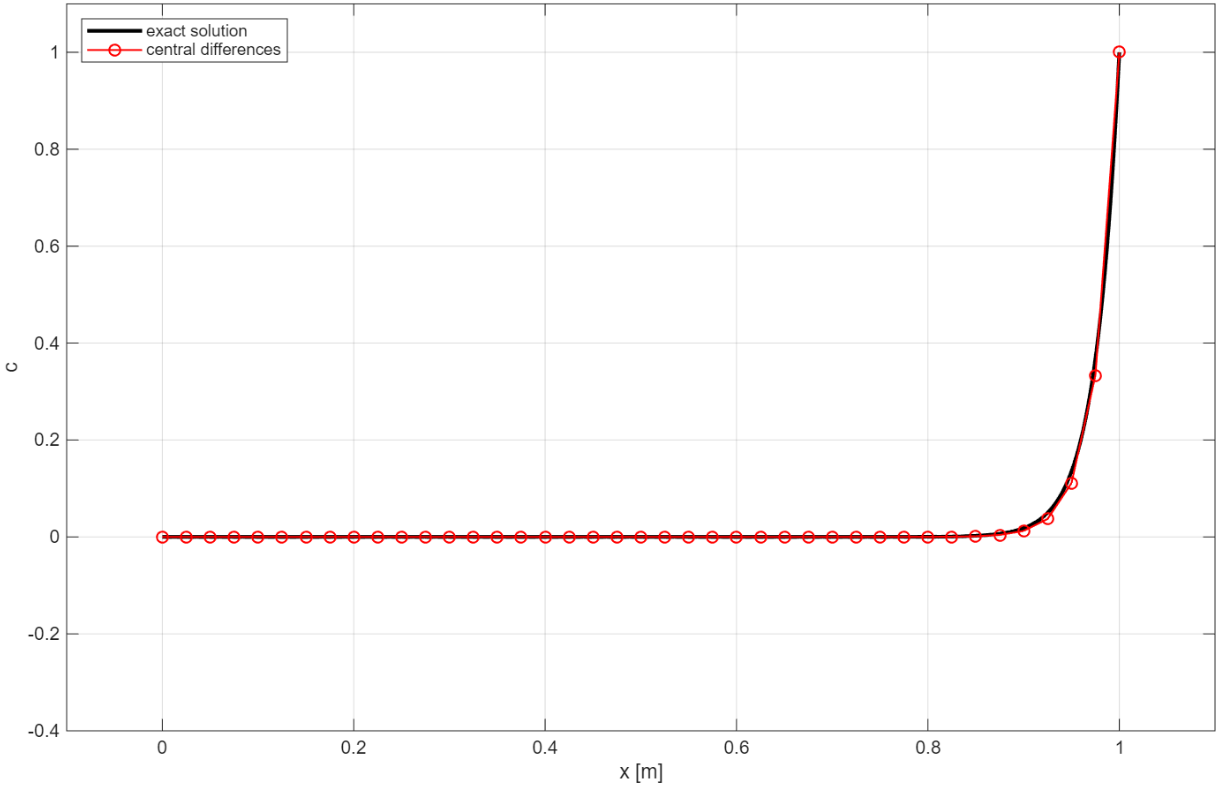
discretization of advection-diffusion equation (I)

- the FTCS scheme is given by

always central differences

$$\frac{c_m^{n+1}-c_m^n}{\Delta t}+u\,\frac{c_{m+1}^n-c_{m-1}^n}{2\Delta x}-\kappa\,\frac{c_{m+1}^n-2c_m^n+c_{m-1}^n}{\Delta x^2}=0$$

- its truncation error is given by $\tau_{\Delta t,\Delta x} = O\left(\Delta t, \Delta x^2\right)$
  - note: accuracy in time is irrelevant here!
- the *numerical* boundary layer solution generated by FTCS may look like



TU Delft

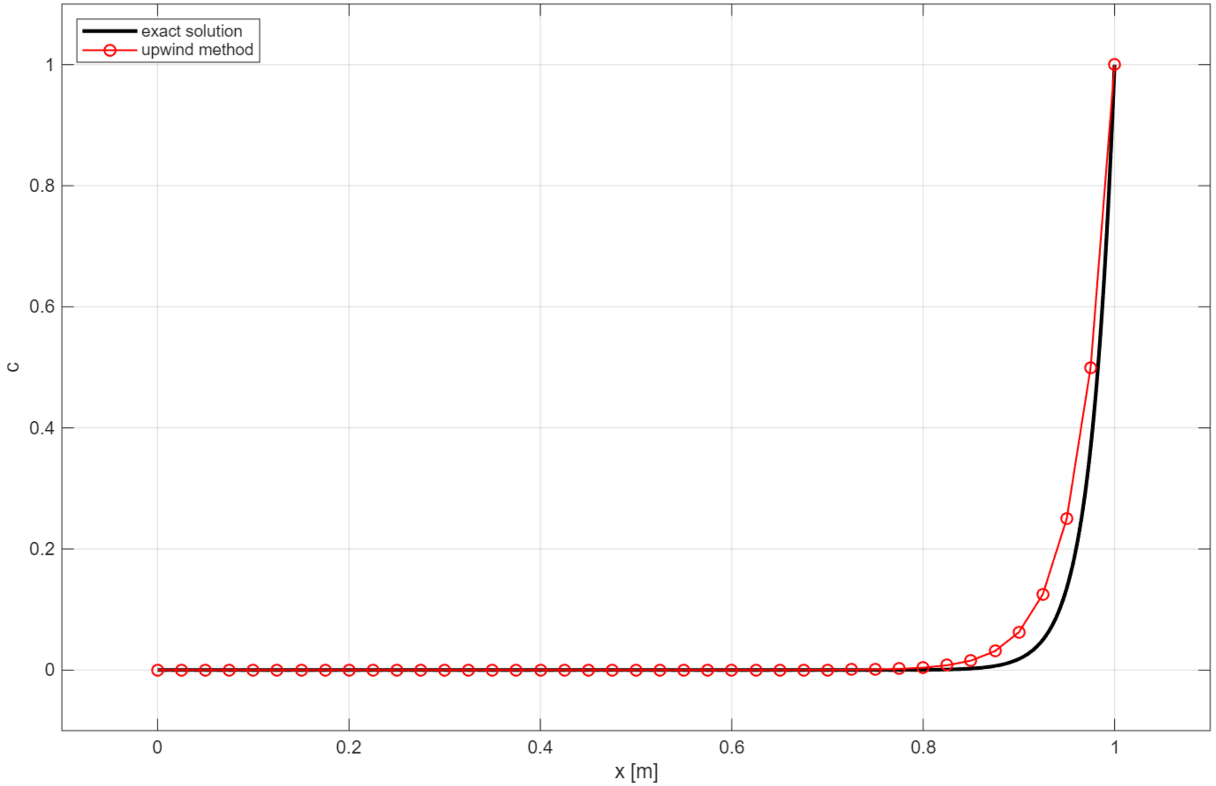
discretization of advection-diffusion equation (II)

- the FTBS scheme is given by (u > 0)

always central differences

$$\frac{c_m^{n+1}-c_m^n}{\Delta t}+u\frac{c_m^n-c_{m-1}^n}{\Delta x}-\kappa\frac{c_{m+1}^n-2c_m^n+c_{m-1}^n}{\Delta x^2}=0$$

- its truncation error is given by $\tau_{\Delta t,\Delta x} = O(\Delta t, \Delta x)$
  - note: time accuracy not relevant!
- the *numerical* boundary layer solution generated by FTBS may look like
  - the numerical boundary layer is thicker, because …
  - … the upwind scheme produces numerical diffusion!



numerical diffusion

- *Q*: what is the amount of numerical diffusion produces by the first order upwind scheme?
- *A*: consider the difference between FTBS and FTCS and reformulate it as a discrete diffusion term

$$u\frac{c_m^n - c_{m-1}^n}{\Delta x} - u\frac{c_{m+1}^n - c_{m-1}^n}{2\Delta x} = -\frac{1}{2}u\Delta x\frac{c_{m+1}^n - 2c_m^n + c_{m-1}^n}{\Delta x^2}$$

- the amount of numerical (or *artificial*) diffusion is given by

$$\boxed{\kappa_a = +\frac{1}{2}u\Delta x}$$

- numerical boundary layer solution produces by FTBS is rather inaccurate if $\kappa_a > \kappa$

TUDelft

stability of FTBS

- reconsider FTBS applied to the advection equation

$$\frac{\partial c}{\partial t} + u\frac{\partial c}{\partial x} = 0 \quad \Rightarrow \quad \frac{c_m^{n+1} - c_m^n}{\Delta t} + u\frac{c_m^n - c_{m-1}^n}{\Delta x} = 0$$

- since the upwind scheme produces numerical diffusion, the FTBS solution must be non-negative(!)
- we can derive stability condition(s) for FTBS based on the following requirement

$$c_m^n \ \geq \ 0 \, , \quad \forall m = 0, \cdots, M \, , \quad \forall n = 0, 1, 2, \cdots$$

- the (prototype) stability condition of the FTBS scheme is given by

$$0 \leq \sigma \equiv \frac{u\Delta t}{\Delta x} \leq 1$$

correct according to
von Neumann analysis

Courant number

TU Delft

stability of FTBS

- reconsider FTBS applied to the advection equation

$$\frac{\partial c}{\partial t} + u\frac{\partial c}{\partial x} = 0 \quad \Rightarrow \quad \frac{c_m^{n+1} - c_m^n}{\Delta t} + u\frac{c_m^n - c_{m-1}^n}{\Delta x} = 0$$

- since the upwind scheme produces numerical diffusion, the FTBS solution must be non-negative(!)
- we can derive stability condition(s) for FTBS based on the following requirement

$$c_m^n \geq 0, \quad \forall m = 0, \cdots, M, \quad \forall n = 0, 1, 2, \cdots$$

- the (prototype) stability condition of the FTBS scheme is given by

$$0 \leq \sigma \equiv \frac{u\Delta t}{\Delta x} \leq 1$$

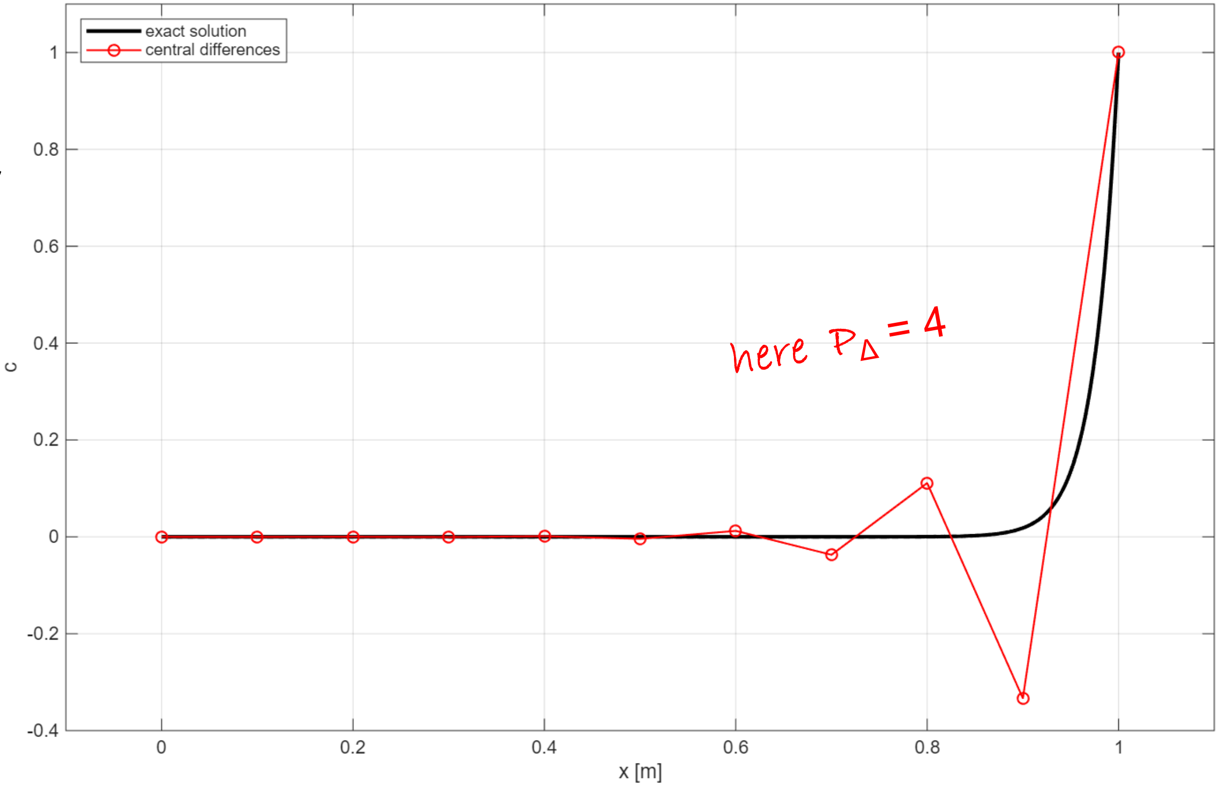
CFL condition

TU Delft

wiggles

- central differences applied to the <u>advection term</u> are prone to generate **spurious oscillations** near steep gradients, especially if the <u>mesh</u> is relatively <u>coarse</u>
  - this is a numerical artefact called **wiggles**
- let's go back to the <u>advection-diffusion equation</u> and reconsider its approximation by the FTCS scheme
- we reconsider the numerical boundary layer solution produces by FTCS *<u>but at a coarser grid</u>*
  - the solution now oscillates around the boundary layer
- the criterion for preventing wiggles is provided by the **mesh-Péclet number** and is given by

$$-2 \leq P_\Delta \equiv \frac{u\Delta x}{\kappa} \leq 2$$

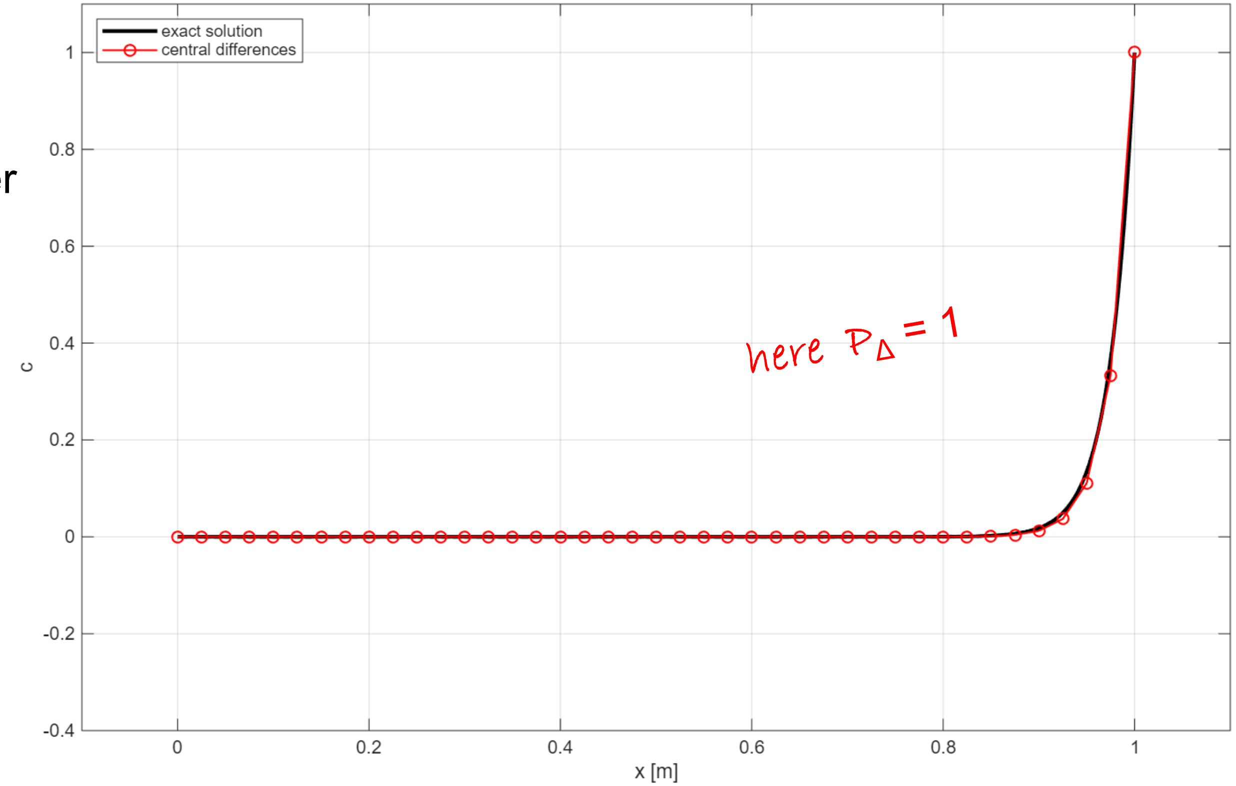


TU Delft

wiggles

- central differences applied to the <u>advection term</u> are prone to generate **spurious oscillations** near steep gradients, especially if the <u>mesh</u> is relatively <u>coarse</u>
  - this is a numerical artefact called **wiggles**
- let's go back to the <u>advection-diffusion equation</u> and reconsider its approximation by the FTCS scheme
- we reconsider the numerical boundary layer solution produces by FTCS <u>*but at a coarser grid*</u>
  - the solution now oscillates around the boundary layer
- the criterion for preventing wiggles is provided by the **mesh-Péclet number** and is given by

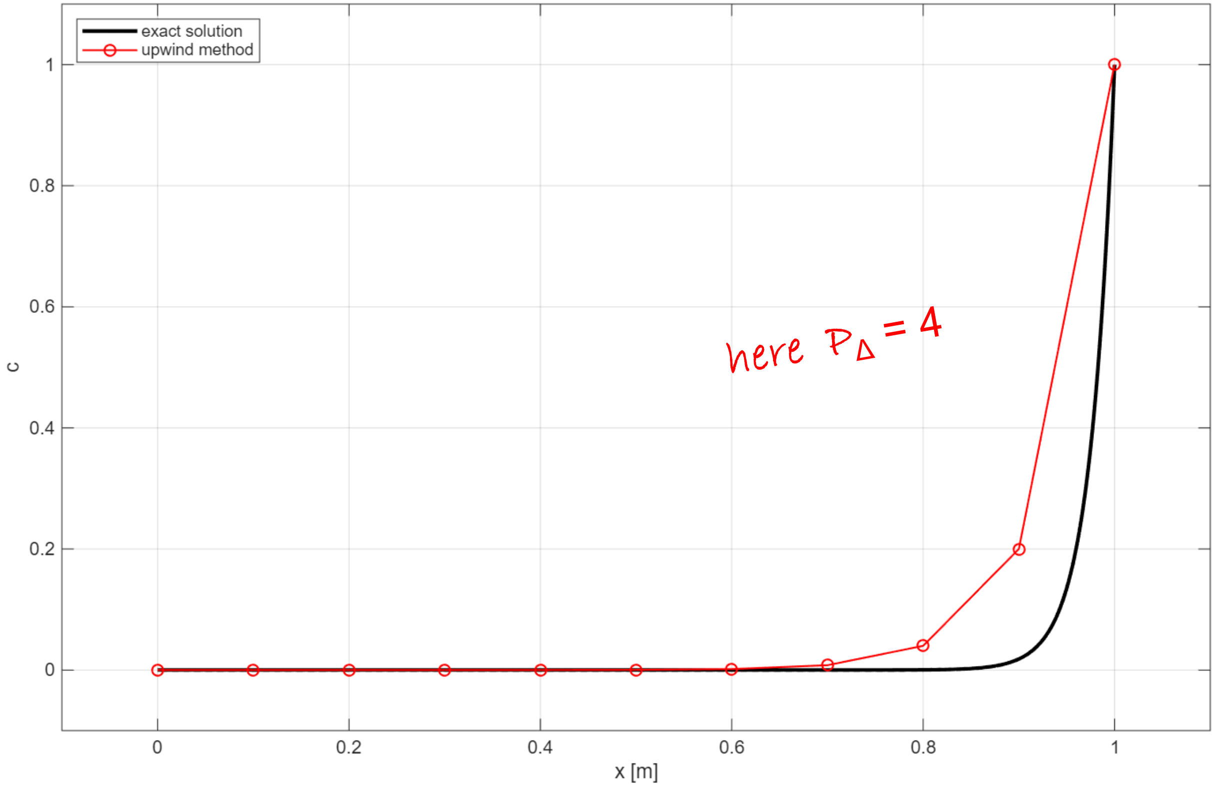
$$-2 \leq P_{\Delta} \equiv \frac{u\Delta x}{\kappa} \leq 2$$



TU Delft

wiggles

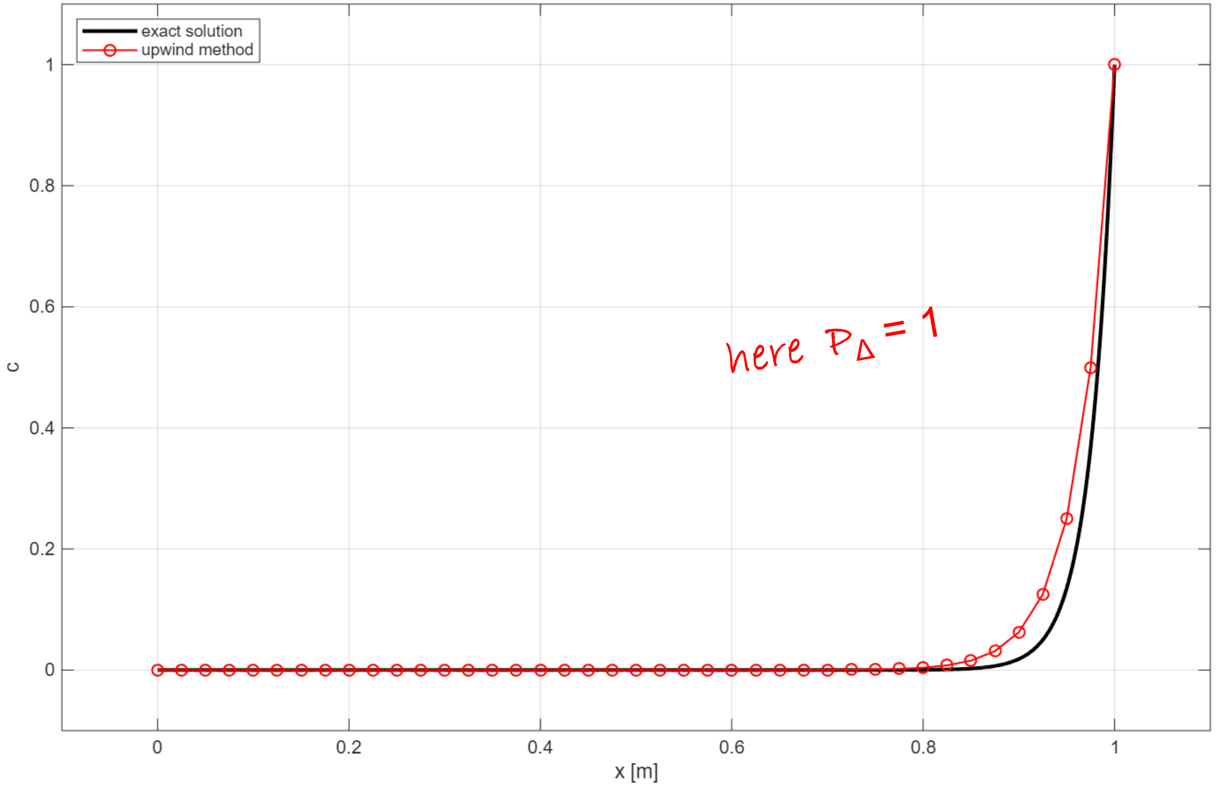
- the usual mechanism to counteract wiggles is numerical diffusion
  - wiggles can be suppressed by adding artificial diffusion
  - however, this must be done carefully: not too much, but also not too little
- one way to prevent wiggles is to apply the upwind scheme
  - it usually produces more than enough numerical diffusion(!)
- for instance, we consider the same coarse grid as in the previous example (P_Δ = 4)
  - the numerical boundary layer produces by FTBS is
  - adding $\kappa_a = \frac{1}{2} u \Delta x$ always reduces the mesh-Péclet number below 2



TU Delft

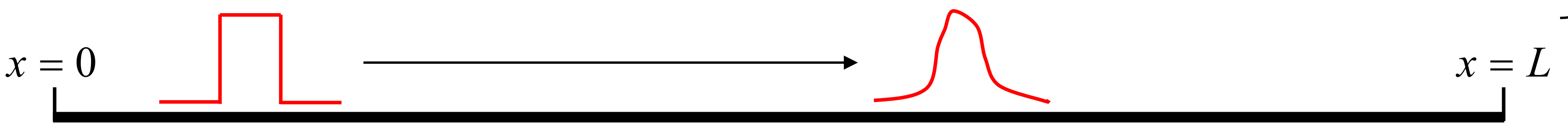
wiggles

- the usual mechanism to counteract wiggles is numerical diffusion
  - wiggles can be suppressed by adding artificial diffusion
  - however, this must be done carefully: not too much, but also not too little
- one way to prevent wiggles is to apply the upwind scheme
  - it usually produces more than enough numerical diffusion(!)
- for instance, we consider the same coarse grid as in the previous example (P_Δ = 4)
  - the numerical boundary layer produces by FTBS is
  - adding $\kappa_a = \frac{1}{2} u \Delta x$ always reduces the mesh-Péclet number below 2



instationary advection-diffusion equation

- another class of problems is described by an instationary advection-diffusion equation
  - the propagation of a disturbance (e.g., wave, cloud of pollutant) in which it spreads over a certain distance simultaneously



to be tested at GA 2.1

- here, discretization of time derivative becomes relevant, including its order of accuracy
- during the GA 2.1 another time integration scheme will be dealt with
  - the second order Crank-Nicolson scheme (or trapezoidal rule)
  - more accurate than forward and backward Euler schemes

key points

- advection dominated problems require different numerical treatment than diffusion dominated problems
- central differences is second order accurate but is prone to generate wiggles
- first order upwind scheme produces numerical diffusion and diminishes spurious oscillations
- explicit schemes are conditionally stable (or occasionally unconditionally unstable!)
  - prototype stability limit for diffusion problems: $\kappa \Delta t / \Delta x^2 \leq \frac{1}{2}$
  - prototype stability limit for advection problems: via Courant number, $|\sigma| \leq 1$

FTCS applied to advection equation

- wiggles in solution to advection-diffusion equation can be avoided if mesh-Péclet number is less than or equal 2