Scientific Computing TU Berlin Winter 2021/22 © Jürgen Fuhrmann
Notebook 27

```
begin
    using PlutoUI ,PlutoVista ,ExtendableGrids ,VoronoiFVM ,GridVisualize ,
HypertextLiteral
end;
```

Transient problems

Transient problems

Time dependent Robin boundary value problem

- Choose final time $T > 0$. Regard functions $(x, t) \to \mathbb{R}$.

$$\begin{aligned} \partial_t u - \nabla \cdot D \nabla u = f \quad & \text{in } \Omega \times [0, T] \\ D \nabla u \cdot \vec{n} + \alpha u = g \quad & \text{on } \partial\Omega \times [0, T] \\ u(x, 0) = u_0(x) \quad & \text{in } \Omega \end{aligned}$$

- This is an initial boundary value problem: besides of the boundary conditions, we need to specify an inital state of the system
- Discretization options:
  - *Rothe method:* first discretize in time, then in space
  - *Method of lines:* first discretize in space, get a huge ODE system, then apply methods for solution of systems of ordinary differential equations

  This difference is more or less formal

Choose time discretization points $0 = t^0 < t^1 \dots < t^N = T$ Set $\tau^n = t^n - t^{n-1}$. Approximate the time derivative by a finite difference in time. Evaluate the main part of the equation for a value interpolated between the old and the new timestep.

For $n = 1 \dots N$, solve

$$\begin{aligned} \frac{u^n - u^{n-1}}{\tau^n} - \nabla \cdot D \nabla u^\theta = f \quad & \text{in } \Omega \times [0,T] \\ D\nabla u^\theta \cdot \vec{n} + \alpha u^\theta = g \quad & \text{on } \partial\Omega \times [0,T] \end{aligned}$$

where $u^\theta = \theta u^n + (1-\theta)u^{n-1}$

- $\theta = 1$: backward (implicit) Euler method: Solve PDE problem in each timestep. First order accuracy in time.
- $\theta = \frac{1}{2}$: Crank-Nicolson scheme: Solve PDE problem in each timestep. Second order accuracy in time.
- $\theta = 0$: forward (explicit) Euler method: First order accurate in time. This does not involve the solution of a PDE problem $\Rightarrow$ Cheap? What do we have to pay for this ?

Time discretization for a homogeneous Neumann problem

Search function $u : \Omega \times [0,T] \to \mathbb{R}$ such that $u(x,0) = u_0(x)$ and

$$\begin{aligned} \partial_t u - \nabla \cdot D \nabla u = 0 \quad & \text{in} \Omega \times [0,T] \\ D\nabla u \cdot \vec{n} = 0 \quad & \text{on} \Gamma \times [0,T] \end{aligned}$$

- Given control volume ω_k, integrate equation over space-time control volume $\omega_k \times (t^{n-1}, t^n)$, divide by τ^n:

$$\begin{aligned} 0 &= \int_{\omega_k} \left(\frac{1}{\tau^n}(u^n - u^{n-1}) - \nabla \cdot D \nabla u^\theta \right) d\omega \\ &= \frac{1}{\tau^n} \int_{\omega_k} (u^n - u^{n-1}) d\omega - \int_{\partial\omega_k} D\nabla u^\theta \cdot \vec{n}_k d\gamma \\ &= -\sum_{l \in \mathcal{N}_k} \int_{\sigma_{kl}} D\nabla u^\theta \cdot \vec{n}_{kl} d\gamma - \int_{\gamma_k} D\nabla u^\theta \cdot \vec{n} d\gamma - \frac{1}{\tau^n} \int_{\omega_k} (u^n - u^{n-1}) d\omega \\ &\approx \underbrace{\frac{|\omega_k|}{\tau^n}(u_k^n - u_k^{n-1})}_{\to M} + \underbrace{\sum_{l \in \mathcal{N}_k} \frac{|\sigma_{kl}|}{h_{kl}}(u_k^\theta - u_l^\theta)}_{\to A} \end{aligned}$$

- Resulting matrix equation:

$$\frac{1}{\tau^n}\left(Mu^n - Mu^{n-1}\right) + Au^\theta = 0$$
$$\frac{1}{\tau^n}Mu^n + \theta Au^n = \frac{1}{\tau^n}Mu^{n-1} + (\theta-1)Au^{n-1}$$
$$u^n + \tau^n M^{-1}\theta Au^n = u^{n-1} + \tau^n M^{-1}(\theta-1)Au^{n-1}$$

- $M = (m_{kl})$, $A = (a_{kl})$ with

$$a_{kl} = \begin{cases} \sum_{l'\in\mathcal{N}_k} D\frac{|\sigma_{kl'}|}{h_{kl'}} & l = k \\ -D\frac{\sigma_{kl}}{h_{kl}}, & l \in \mathcal{N}_k \\ 0, & else \end{cases}$$
$$m_{kl} = \begin{cases} |\omega_k| & l = k \\ 0, & else \end{cases}$$

- $\Rightarrow \theta A + M$ is strictly diagonally dominant!
- $\sum_{l=1}^n a_{kl} = 0$

Lemma Assume A has positive main diagonal entries, nonpositive off-diagonal entries and row sum zero. Then, $||(I+A)^{-1}||_\infty \leq 1$.

Proof: Assume that $||(I+A)^{-1}||_\infty > 1$. $I+A$ is an irreducible M-matrix, thus $(I+A)^{-1}$ has positive entries.

Then for α_{ij} being the entries of $(I+A)^{-1}$,

$$\max_{i=1}^n \sum_{j=1}^n \alpha_{ij} > 1.$$

Let k be a row where the maximum is reached. Let $e = (1\dots 1)^T$. Then for $v = (I+A)^{-1}e$ we have that $v > 0$, $v_k > 1$ and $v_k \geq v_j$ for all $j \neq k$. The kth equation of $e = (I+A)v$ then looks like

$$\begin{aligned} 1 &= v_k + v_k \sum_{j\neq k} |a_{kj}| - \sum_{j\neq k} |a_{kj}| v_j \\ &\geq v_k + v_k \sum_{j\neq k} |a_{kj}| - \sum_{j\neq k} |a_{kj}| v_k \\ &= v_k \\ &> 1 \end{aligned}$$

This contradiction enforces $||(I+A)^{-1}||_\infty \leq 1$. $\square$

Stability condition

When can we have an estimate $||u^n||_\infty < ||u^{n-1}||_\infty$?

Regard the matrix equation again:

$$u^n + \tau^n M^{-1}\theta Au^n = u^{n-1} + \tau^n M^{-1}(\theta-1)Au^{n-1} =: B^n u^{n-1}$$
$$u^n = (I + \tau^n M^{-1}\theta A)^{-1} B^n u^{n-1}$$

with $B^n = I + \tau^n M^{-1}A$

From the lemma we have $||(I+\tau^n M^{-1}\theta A)^{-1}||_\infty \leq 1 \Rightarrow ||u^n||_\infty \leq ||B^n u^{n-1}||_\infty$ and we need to estimate $||B||_\infty$

- For the entries b^n_{kl} of B^n, we have

$$b^n_{kl} = \begin{cases} 1+\frac{\tau^n}{m_{kk}}(\theta-1)a_{kk}, & k=l \\ \frac{\tau^n}{m_{kk}}(\theta-1)a_{kl}, & else \end{cases}$$

- In any case, $b_{kl} \geq 0$ for $k \neq l$ because $a_{kl} \leq 0$
- Assume If in addition $b_{kk} \geq 0$, one can estimate $||B||_\infty = \max_{k=1}^N \sum_{l=1}^N b_{kl}$. Then

$$\sum_{l=1}^N b_{kl} = 1+(\theta-1)\frac{\tau^n}{m_{kk}}\left(a_{kk}+\sum_{l\in\mathcal{N}_k} a_{kl}\right) = 1 \qquad \text{and } ||B||_\infty = 1.$$

For a shape regular triangulation in $\mathbb{R}^d$, we can assume that $m_{kk} = |\omega_k| \sim h^d$, and $a_{kl} = \frac{|\sigma_{kl}|}{h_{kl}} \sim \frac{h^{d-1}}{h} = h^{d-2}$, thus $\frac{a_{kk}}{m_{kk}} \leq \frac{1}{Ch^2}$ for some constant C

- $b_{kk} \geq 0$ gives

$$(1-\theta)\frac{\tau^n}{m_{kk}} a_{kk} \leq 1$$

- A sufficient condition is that for some $C > 0$,

$$(1-\theta)\frac{\tau^n}{Ch^2} \leq 1$$
$$(1-\theta)\tau^n \leq Ch^2$$

- Method stability:
  - Implicit Euler: $\theta = 1 \Rightarrow$ unconditional stability !
  - Explicit Euler: $\theta = 0 \Rightarrow$ CFL condition $\tau \leq Ch^2$
  - Crank-Nicolson: $\theta = \frac{1}{2} \Rightarrow$ CFL condition $\tau \leq 2Ch^2$

We see a tradeoff between stability and accuracy.

- $\tau \leq Ch^2$ is called *Courant-Friedrichs-Levy (CFL)* condition
- Explicit (forward) Euler method can be applied on very fast systems (GPU), with small time step comes a high accuracy in time.
- Implicit Euler: unconditional stability – helpful when stability is of utmost importance, and accuracy in time is less important
- For hyperbolic systems (pure convection without diffusion), the CFL conditions is $\tau \leq Ch$ and therefore easier to fulfill, thus in this case explicit computations are mostly preferred

Discrete Maximum principle

Regard the implicit Euler method

$$\frac{1}{\tau^n}Mu^n + Au^n = \frac{1}{\tau}Mu^{n-1}$$
$$\frac{1}{\tau^n}m_{kk}u_k^n + a_{kk}u_k^n = \frac{1}{\tau^n}m_{kk}u_k^{n-1} + \sum_{k\neq l}(-a_{kl})u_l^n$$
$$u_k^n = \frac{1}{\frac{1}{\tau^n}m_{kk} + \sum_{l\neq k}(-a_{kl})}\left(\frac{1}{\tau^n}m_{kk}u_k^{n-1} + \sum_{l\neq k}(-a_{kl})u_l^n\right)$$
$$\leq \frac{\frac{1}{\tau^n}m_{kk} + \sum_{l\neq k}(-a_{kl})}{\frac{1}{\tau^n}m_{kk} + \sum_{l\neq k}(-a_{kl})}\max(\{u_k^{n-1}\}\cup\{u_l^n\}_{l\in\mathcal{N}_k})$$
$$\leq \max(\{u_k^{n-1}\}\cup\{u_l^n\}_{l\in\mathcal{N}_k})$$

- Provided, the right hand side is zero, the solution in a given node is bounded by the value from the old timestep, and by the solution in the neigboring points.
- No new local maxima can appear during time evolution
- There is a continuous counterpart which can be derived from weak solution theory
- Sign pattern is crucial for the proof.

Nonnegativity

$$u^n + \tau^n M^{-1}Au^n = u^{n-1}$$
$$u^n = (I + \tau^n M^{-1}A)^{-1}u^{n-1}$$

- $(I + \tau^n M^{-1}A)$ is an M-Matrix
- If $u_0 > 0$, then $u^n > 0\,\forall n > 0$

Mass conservation

- Continuous case: $\int_\Omega \nabla\cdot D\nabla u d\vec{x} = \int_{\partial\Omega} D\nabla u\cdot\vec{n}d\gamma = 0$
- Discrete equivalent:

$$\sum_{k=1}^N\left(a_{kk}u_k + \sum_{l\in\mathcal{N}_k}a_{kl}u_l\right) = \sum_{k=1}^N\sum_{l=1,l\neq k}^N a_{kl}(u_l - u_k)$$
$$= \sum_{k=1}^N\sum_{l=1,l<k}^N (a_{kl}(u_l-u_k) + a_{lk}(u_k-u_l))$$
$$= 0$$

- $\Rightarrow \int_\Omega u^n d\vec{x} = \int_\Omega u^{n-1}d\vec{x}$:
- Discrete equivalent: $\sum_{k=1}^N m_{kk}u_k^n = \sum_{k=1}^N m_{kk}u_k^{n-1}$

The amount of "species" in the domain remains constant.

Examples in VoronoiFVM.jl

General settings

Initial value problem with homgeneous Neumann boundary conditions

$$\Omega = (0,1)^d,\ d = 1,2$$

$$T = [0, t_{end}]$$

Define function for initial value u_0 with two methods - for 1D and 2D problems

```
fpeak (generic function with 2 methods)
```

```
begin
    fpeak(x)=exp(-100*(x-0.25)^2)
    fpeak(x,y)=exp(-100*((x-0.25)^2+(y-0.25)^2))
end
```

Create discretization grid in 1D or 2D

```
create_grid (generic function with 1 method)
```

```
function create_grid(nx,dim)
    X=collect(0:1.0/nx:1)
    if dim==1
      grid=simplexgrid(X)
    else
      grid=simplexgrid(X,X)
    end
end
```

Diffusion problem

$$\partial_t u - \nabla \cdot D \nabla u = 0 \text{ in } \Omega$$

$$D \nabla u \cdot \vec{n} = 0 \text{ on } \partial\Omega$$

$$u|_{t=0} = u_0$$

```
diffusion (generic function with 1 method)
```

```
function diffusion(;n=100,dim=1,tstep=1.0e-4,tend=1, D=1.0)
    grid=create_grid(n,dim)

    ## Diffusion flux between neigboring control volumes
    function flux!(f,u,edge)
        f[1]=D*(u[1,1]-u[1,2])
    end

    ## Storage term (under time derivative)
    function storage!(f,u,node)
        f[1]=u[1]
    end


    sys=VoronoiFVM.System(grid,flux=flux!,storage=storage!, species=[1])

    inival=unknowns(sys)

    ## Broadcast the initial value
    inival[1,:].=map(fpeak,grid)

    control=VoronoiFVM.SolverControl()
    control.Δt_min=0.01*tstep
    control.Δt=tstep
    control.Δt_max=0.1*tend
    control.Δu_opt=0.05

    tsol=solve(sys,inival=inival,times=[0,tend];control=control)
    return grid,tsol
end
```

```
dim = 1
```

```
dim=1
```

```
grid_diffusion,tsol_diffusion=diffusion(dim=dim,n=50);
```

```
TransientSolution{Float64, 3, Vector{Matrix{Float64}}, Vector{Float64}}
```

```
typeof(tsol_diffusion)
```

```
t: 44-element Vector{Float64}:
 0.0
 0.0001
 0.00022
 0.000364
 0.0005368
 0.00074416
 0.000992992
 ⋮
 0.5098373499892658
 0.6098373499892658
 0.7098373499892657
 0.8098373499892657
 0.9098373499892657
 1.0
u: 44-element Vector{Matrix{Float64}}:
 [0.0019304541362277093 0.005041760259690979 … 7.18533563590225e-24 3.7233631217505106e-25
 [0.003387008809660418 0.006300118156525835 … 2.0008275372882336e-19 6.669449946714916e-20
 [0.00514807562457328 0.008083186982761384 … 6.878526294212977e-18 2.621131422496832e-18]
 [0.007404041865364755 0.010537328310908471 … 1.4776955885734591e-16 6.338093825568941e-17
 [0.010400625445238012 0.013868893477498723 … 2.4970324296706681e-15 1.1914254063793755e-15
 [0.014449460531084734 0.01835458696419307 … 3.573404400316756e-14 1.8774785069159207e-14]
 [0.019935610876004123 0.02434513243667107 … 4.455948816039796e-13 2.5540340838690913e-13]
 ⋮
 [0.18090105364719605 0.18089376259079934 … 0.17351852237753165 0.1735112313151921]
 [0.17906602640228114 0.17906235634779133 … 0.17534992921643835 0.17534625916074723]
 [0.17814234043496177 0.17814049306302712 … 0.17627179262173392 0.1762699452495563]
 [0.17767739057964502 0.17767646067993437 … 0.17673582502921079 0.17673489512945098]
 [0.1774433517750711 0.17744288369746195 … 0.1769694020166165 0.17696893393899743]
 [0.17733167831956984 0.17733143060040125 … 0.17708085511104252 0.17708060739548298]
```

```
tsol_diffusion
```

- Documentation for **`SolverControl`**
- Documentation for **`solve`**
- Documentation for **`TransientSolution`**

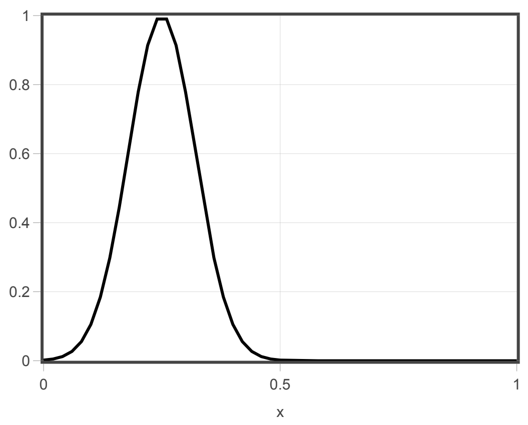
Timestep: 1

```
md"""
Timestep: $(@bind t_diffusion Slider(1:length(tsol_diffusion),default=1,show_value=true))
"""
```

Time: 0.0

sol_diffusion =
[0.00193045, 0.00504176, 0.0121552, 0.0270518, 0.0555762, 0.105399, 0.18452, 0.298197, 0.4

```
sol_diffusion=tsol_diffusion[1,:,t_diffusion]
```




```
visd=GridVisualizer(Plotter=PlutoVista,resolution=(400,300),dim=dim);visd
```

```
scalarplot!(visd,grid_diffusion,sol_diffusion,limits=(0,1),show=true,levels=20,
colormap=:summer,xlabel="x")
```

Reaction-diffusion problem

Diffusion + physical process which "eats" species

$$\partial_t u - \nabla \cdot D \nabla u + Ru = 0 \text{ in } \Omega$$

$$D \nabla u \cdot \vec{n} = 0 \text{ on } \partial\Omega$$

$$u|_{t=0} = u_0$$

reaction_diffusion (generic function with 1 method)

```
function reaction_diffusion(;
        n=100,
        dim=1,
        tstep=1.0e-4,
        tend=1,
        D=1.0,
        R=10.0)

    grid=create_grid(n,dim)
    ## Diffusion flux between neigboring control volumes
    function flux!(f,u,edge)
        f[1]=D*(u[1,1]-u[1,2])
    end

    ## Storage term (under time derivative)
    function storage!(f,u,node)
        f[1]=u[1]
    end

    ## Reaction term
    function reaction!(f,u,node)
        f[1]=R*u[1]
    end
 sys=VoronoiFVM.System(grid,flux=flux!,storage=storage!, reaction=reaction!, species=
[1])

    ## Create a solution array
    inival=unknowns(sys)

    ## Broadcast the initial value
    inival[1,:].=map(fpeak,grid)



    control=VoronoiFVM.SolverControl()
    control.Δt_min=0.01*tstep
    control.Δt=tstep
    control.Δt_max=0.1*tend
    control.Δu_opt=0.1

    tsol=solve(sys, inival=inival, times=[0,tend], control=control)
    return grid,tsol
end
```

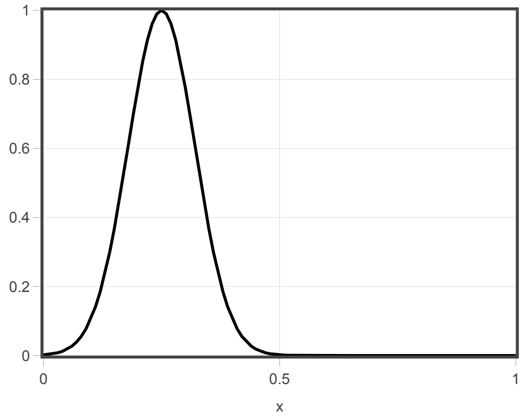
```
grid_rd,tsol_rd=reaction_diffusion(dim=dim,n=100,R=10);
```

Timestep: 1

Time: 0.0

sol_rd =

[0.00193045, 0.00315111, 0.00504176, 0.00790705, 0.0121552, 0.0183156, 0.0270518, 0.039163




```
visrd=GridVisualizer(Plotter=PlutoVista,resolution=(400,300),dim=dim,xlabel="x");visrd
```