Scientific Computing WS 2017/2018

Lecture 23

Jürgen Fuhrmann

juergen.fuhrmann@wias-berlin.de

The convection - diffusion equation

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

$$\begin{aligned} \partial_t u - \nabla \cdot (D\nabla u - u\mathbf{v}) &= f \quad \text{in } \Omega \times [0, T] \\ (D\nabla u - u\mathbf{v}) \cdot \mathbf{n} + \alpha(u - w) &= 0 \quad \text{on } \Gamma \times [0, T] \end{aligned}$$

- $u(x, t)$: species concentration, temperature
- $\mathbf{j} = D\nabla u - u\mathbf{v}$: species flux
- D: diffusion coefficient
- $\mathbf{v}(x, t)$: velocity of medium (e.g. fluid)
  - Given analytically
  - Solution of free flow problem (Navier-Stokes equation)
  - Flow in porous medium (Darcy equation): $\mathbf{v} = -\kappa\nabla p$ where

$$-\nabla \cdot (\kappa\nabla p) = 0$$

- For constant density, the divergence conditon $\nabla \cdot v = 0$ holds.

Finite volumes for convection diffusion

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

$$\begin{aligned} \partial_t u - \nabla \cdot \mathbf{j} &= 0 \quad \text{in } \Omega \times [0, T] \\ \mathbf{j}\mathbf{n} + \alpha(u - w) &= 0 \quad \text{on } \Gamma \times [0, T] \end{aligned}$$

- Integrate time discrete equation over control volume

$$\begin{aligned} 0 &= \int_{\omega_k} \left(\frac{1}{\tau}(u - v) - \nabla \cdot \mathbf{j} \right) d\omega = \frac{1}{\tau} \int_{\omega_k} (u - v) d\omega - \int_{\partial\omega_k} \mathbf{j} \cdot \mathbf{n}_k d\gamma \\ &= - \sum_{l \in \mathcal{N}_k} \int_{\sigma_{kl}} \mathbf{j} \cdot \mathbf{n}_{kl} d\gamma - \int_{\gamma_k} \mathbf{j} \cdot \mathbf{n} d\gamma - \frac{1}{\tau} \int_{\omega_k} (u - v) d\omega \\ &\approx \underbrace{\frac{|\omega_k|}{\tau}(u_k - v_k)}_{\to M} + \sum_{l \in \mathcal{N}_k} \underbrace{\frac{|\sigma_{kl}|}{h_{kl}} g_{kl}(u_k, u_l)}_{\to A_0} + \underbrace{|\gamma_k| \alpha(u_k - g_k)}_{\to D} \end{aligned}$$

- $\frac{1}{\tau} Mu + Au = \frac{1}{\tau} Mv$ where $A = A_0 + D$, $A_0 = (a_{kj})$

Central Difference Flux Approximation

- g_{kl} approximates normal convective-diffusive flux between control volumes ω_k, ω_l: $g_{kl}(u_k - u_l) \approx -(D\nabla u - u\mathbf{v}) \cdot n_{kl}$
- Let $v_{kl} = \frac{1}{|\sigma_{kl}|} \int \sigma_{kl} \mathbf{v} \cdot \mathbf{n}_{kl} d\gamma$ approximate the normal velocity $\mathbf{v} \cdot \mathbf{n}_{kl}$
- Central difference flux:

$$
\begin{aligned}
g_{kl}(u_k, u_l) &= D(u_k - u_l) + h_{kl}\frac{1}{2}(u_k + u_l)v_{kl} \\
&= (D + \frac{1}{2}h_{kl}v_{kl})u_k - (D - \frac{1}{2}h_{kl}v_{kl})u_l
\end{aligned}
$$

- if $\mathbf{v}_{kl}$ is large compared to h_{kl}, the corresponding matrix (off-diagonal) entry may become positive
- Non-positive off-diagonal entries only guaranteed for $h \to 0$!
- Otherwise, we can prove the discrete maximum principle

Simple upwind flux discretization

- Force correct sign of convective flux approximation by replacing central difference flux approximation $h_{kl}\frac{1}{2}(u_k+u_l)v_{kl}$ by

$$\left(\begin{cases} h_{kl}u_kv_{kl}, & v_{kl}<0 \\ h_{kl}u_lv_{kl}, & v_{kl}>0 \end{cases}\right) = h_{kl}\tfrac{1}{2}(u_k+u_l)v_{kl} + \underbrace{\frac{1}{2}h_{kl}|v_{kl}|}_{\text{Artificial Diffusion } \tilde{D}}$$

- Upwind flux:

$$\begin{aligned} g_{kl}(u_k,u_l) &= D(u_k-u_l) + \begin{cases} h_{kl}u_kv_{kl}, & v_{kl}>0 \\ h_{kl}u_lv_{kl}, & v_{kl}<0 \end{cases} \\ &= (D+\tilde{D})(u_k-u_l) + h_{kl}\frac{1}{2}(u_k+u_l)v_{kl} \end{aligned}$$

  - M-Property guaranteed unconditonally !
  - Artificial diffusion introduces error: second order approximation replaced by first order approximation

Exponential fitting flux I

- Project equation onto edge $x_K x_L$ of length $h = h_{kl}$, let $v = -v_{kl}$, integrate once

$$
\begin{aligned}
u' - uv &= j \\
u|_0 &= u_k \\
u|_h &= u_l
\end{aligned}
$$

- Linear ODE
- Solution of the homogeneus problem:

$$
\begin{aligned}
u' - uv &= 0 \\
u'/u &= v \\
\ln u &= u_0 + vx \\
u &= K \exp(vx)
\end{aligned}
$$

Exponential fitting II

- Solution of the inhomogeneous problem: set $K = K(x)$:

$$K' \exp(vx) + vK \exp(vx) - vK \exp(vx) = -j$$
$$K' = -j \exp(-vx)$$
$$K = K_0 + \frac{1}{v} j \exp(-vx)$$

- Therefore,

$$u = K_0 \exp(vx) + \frac{1}{v} j$$
$$u_k = K_0 + \frac{1}{v} j$$
$$u_l = K_0 \exp(vh) + \frac{1}{v} j$$

Exponential fitting III

- Use boundary conditions

$$
\begin{aligned}
K_0 &= \frac{u_k - u_l}{1 - \exp(vh)} \\
u_k &= \frac{u_k - u_l}{1 - \exp(vh)} + \frac{1}{v} j \\
j =& \frac{v}{\exp(vh) - 1}(u_k - u_l) + v u_k \\
=& v\left(\frac{1}{\exp(vh) - 1} + 1\right) u_k - \frac{v}{\exp(vh) - 1} u_l \\
=& v\left(\frac{\exp(vh)}{\exp(vh) - 1}\right) u_k - \frac{v}{\exp(vh) - 1} u_l \\
=& \frac{-v}{\exp(-vh) - 1} u_k - \frac{v}{\exp(vh) - 1} u_l \\
=& \frac{B(-vh) u_k - B(vh) u_l}{h}
\end{aligned}
$$

where $B(\xi) = \frac{\xi}{\exp(\xi) - 1}$: Bernoulli function

Exponential fitting IV

- General case: $Du' - uv = D(u' - u\frac{v}{D})$
- Upwind flux:

$$g_{kl}(u_k, u_l) = D(B(\frac{-v_{kl}h_{kl}}{D})u_k - B(\frac{v_{kl}h_{kl}}{D})u_l)$$

- Allen+Southwell 1955
- Scharfetter+Gummel 1969
- Ilin 1969
- Chang+Cooper 1970
- Guaranteed sign pattern, M property!

Exponential fitting: Artificial diffusion

- Difference of exponential fitting scheme and central scheme
- Use: $B(-x) = B(x) + x \Rightarrow$

$$B(x) + \frac{1}{2}x = B(-x) - \frac{1}{2}x = B(|x|) + \frac{1}{2}|x|$$

$$\begin{aligned}
D_{art}(u_k - u_l) =& D(B(\frac{-vh}{D})u_k - B(\frac{vh}{D})u_l) - D(u_k - u_l) + h\frac{1}{2}(u_k + u_l)v \\
=& D(\frac{-vh}{2D} + B(\frac{-vh}{D}))u_k - D(\frac{vh}{2D} + B(\frac{vh}{D})u_l) - D(u_k - u_l) \\
=& D\left(\frac{1}{2}|\frac{vh}{D}| + B(|\frac{vh}{D}|\right) - 1)(u_k - u_l)
\end{aligned}$$

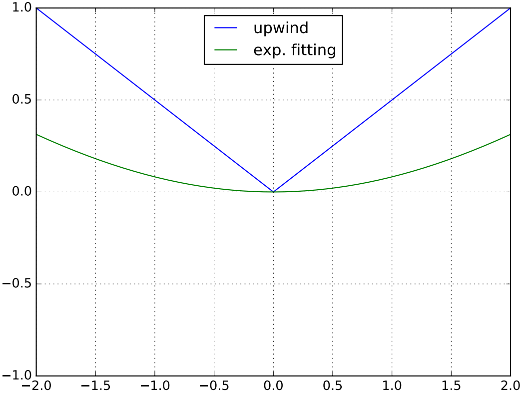
- Further, for $x > 0$:

$$\frac{1}{2}x \geq \frac{1}{2}x + B(x) - 1 \geq 0$$

- Therefore

$$\frac{|vh|}{2} \geq D_{art} \geq 0$$

Exponential fitting: Artificial diffusion II



Comparison of artificial diffusion functions $\frac{1}{2}|x|$ (upwind) and $\frac{1}{2}|x| + B(|x|) - 1$ (exp. fitting)

1D Convection-Diffusion implementation: central differences

```
F=0;
U=0;
for (int k=0, l=1;k<n-1;k++,l++)
{
  double g_kl=D - 0.5*(v*h);
  double g_lk=D + 0.5*(v*h);
  M(k,k)+=g_kl/h;
  M(k,l)-=g_kl/h;
  M(l,l)+=g_lk/h;
  M(l,k)-=g_lk/h;
}
M(0,0)+=1.0e30;
M(n-1,n-1)+=1.0e30;
F(n-1)=1.0e30;
```

1D Convection-Diffusion implementation: upwind scheme

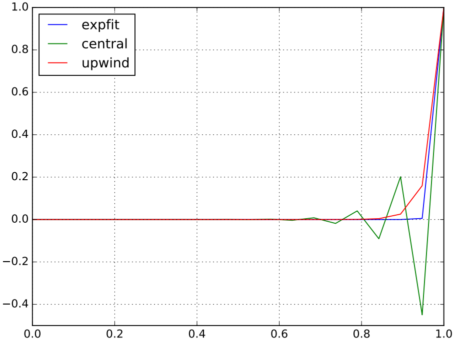
```
F=0;
U=0;
for (int k=0, l=1;k<n-1;k++,l++)
{
  double g_kl=D;
  double g_lk=D;
  if (v<0) g_kl-=v*h;
  else  g_lk+=v*h;

  M(k,k)+=g_kl/h;
  M(k,l)-=g_kl/h;
  M(l,l)+=g_lk/h;
  M(l,k)-=g_lk/h;
}

M(0,0)+=1.0e30;
M(n-1,n-1)+=1.0e30;
F(n-1)=1.0e30;
```

1D Convection-Diffusion implementation: exponential fitting scheme

```
  inline double B(double x)
  {
    if (std::fabs(x)<1.0e-10) return 1.0;
    return x/(std::exp(x)-1.0);
  }

...

    F=0;
    U=0;
    for (int k=0, l=1;k<n-1;k++,l++)
    {
      double g_kl=D* B(v*h/D);
      double g_lk=D* B(-v*h/D);
      M(k,k)+=g_kl/h;
      M(k,l)-=g_kl/h;
      M(l,l)+=g_lk/h;
      M(l,k)-=g_lk/h;
    }

    M(0,0)+=1.0e30;
    M(n-1,n-1)+=1.0e30;
    F(n-1)=1.0e30;
```

Convection-Diffusion test problem, N=20

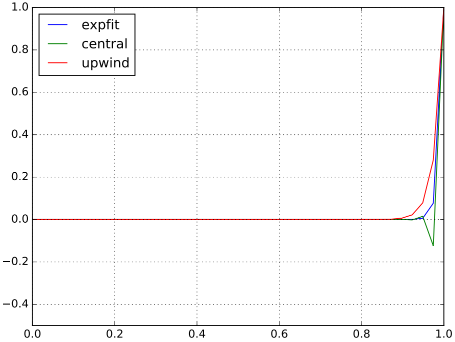
- $\Omega = (0,1)$, $-\nabla \cdot (D\nabla u + uv) = 0$, $u(0) = 0$, $u(1) = 1$
- $V = 1$, $D = 0.01$



- Exponential fitting: sharp boundary layer, for this problem it is exact
- Central differences: unphysical
- Upwind: larger boundary layer

Convection-Diffusion test problem, N=40

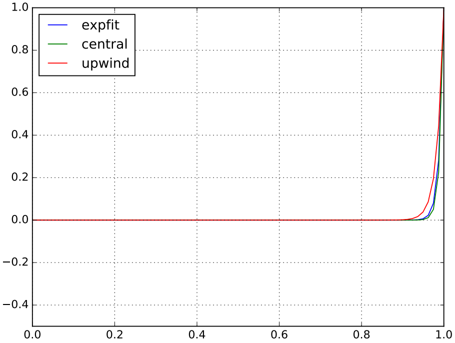
- $\Omega = (0, 1)$, $-\nabla \cdot (D\nabla u + uv) = 0$, $u(0) = 0$, $u(1) = 1$
- $V = 1$, $D = 0.01$



- Exponential fitting: sharp boundary layer, for this problem it is exact
- Central differences: unphysical, but less ''wiggles"
- Upwind: larger boundary layer

Convection-Diffusion test problem, N=80

- $\Omega = (0,1)$, $-\nabla \cdot (D\nabla u + uv) = 0$, $u(0) = 0$, $u(1) = 1$
- $V = 1$, $D = 0.01$



- Exponential fitting: sharp boundary layer, for this problem it is exact
- Central differences: grid is fine enough to yield M-Matrix property, good approximation of boundary layer due to higher convergence order
- Upwind: "smearing" of boundary layer

1D convection diffusion summary

- Upwinding and exponential fitting unconditionally yield the M-property of the discretization matrix
- Exponential fitting for this case (zero right hand side, 1D) yields exact solution. It is anyway "less diffusive" as artificial diffusion is optimized
- Central scheme has higher convergence order than upwind (and exponential fitting) but on coarse grid it may lead to unphysical oscillations
- For 2/3D problems, sufficiently fine grids to stabilize central scheme may be prohibitively expensive
- Local grid refinement may help to offset artificial diffusion

Convection-diffusion and finite elements

Search function $u : \Omega \to \mathbb{R}$ such that

$$-\nabla(\cdot D\nabla u - u\mathbf{v}) = f \quad \text{in } \Omega$$
$$u = u_D \quad \text{on } \partial\Omega$$

- Assume v is divergence-free, i.e. $\nabla \cdot v = 0$.
- Then the main part of the equation can be reformulated as

$$-\nabla(\cdot D\nabla u) + v \cdot \nabla u = 0 \quad \text{in } \Omega$$

yielding a weak formulation: find $u \in H^1(\Omega)$ such that $u - u_D \in H_0^1(\Omega)$ and $\forall w \in H_0^1(\Omega)$,

$$\int_\Omega D\nabla u \cdot \nabla w \; dx + \int_\Omega \mathbf{v} \cdot \nabla u \; w \; dx = \int_\Omega fw \; dx$$

- Galerkin formulation: find $u_h \in V_h$ with bc. such that $\forall w_h \in V_h$

$$\int_\Omega D\nabla u_h \cdot \nabla w_h \; dx + \int_\Omega \mathbf{v} \cdot \nabla u_h \; w_h \; dx = \int_\Omega fw_h \; dx$$

Convection-diffusion and finite elements II

- Galerkin ansatz has similar problems as central difference ansatz in the finite volume/finite difference case $\Rightarrow$ stabilization ?
- Most popular: streamline upwind Petrov-Galerkin

$$\int_\Omega D\nabla u_h \cdot \nabla w_h \ dx + \int_\Omega \mathbf{v} \cdot \nabla u_h \ w_h \ dx + S(u_h, w_h) = \int_\Omega f w_h \ dx$$

with

$$S(u_h, w_h) = \sum_K \int_K (-\nabla(\cdot D\nabla u_h - u_h\mathbf{v}) - f)\delta_K v \cdot w_h \ dx$$

where $\delta_K = \frac{h_K^v}{2|\mathbf{v}|}\xi(\frac{|\mathbf{v}|h_K^v}{D})$ with $\xi(\alpha) = coth(\alpha) - \frac{1}{\alpha}$ and h_K^v is the size of element K in the direction of $\mathbf{v}$.

Convection-diffusion and finite elements III

- Many methods to stabilize, *none* guarantees M-Property even on weakly acute meshes ! (V. John, P. Knobloch, Computer Methods in Applied Mechanics and Engineering, 2007)
- Comparison paper:

  M. Augustin, A. Caiazzo, A. Fiebach, J. Fuhrmann, V. John, A. Linke, and R. Umla, "An assessment of discretizations for convection-dominated convection-diffusion equations," Comp. Meth. Appl. Mech. Engrg., vol. 200, pp. 3395–3409, 2011:

  - if it is necessary to compute solutions without spurious oscillations: use FVM, taking care on the construction of an appropriate grid might be essential for reducing the smearing of the layers,
  - if sharpness and position of layers are important and spurious oscillations can be tolerated: often the SUPG method is a good choice.

- Topic of ongoing research

~

Nonlinear problems

Nonlinear problems: motivation

- Assume nonlinear dependency of some coefficients of the equation on the solution. E.g. nonlinear diffusion problem

$$-\nabla(\cdot D(u)\nabla u) = f \quad \text{in } \Omega$$
$$u = u_D \text{on} \partial\Omega$$

- FE+FV discretization methods lead to large nonlinear systems of equations

Nonlinear problems: caution!

This is a significantly more complex world:

- Possibly multiple solution branches
- Weak formulations in L^p spaces
- No direct solution methods
- Narrow domains of definition (e.g. only for positive solutions)

Finite element discretization for nonlinear diffusion

- Find $u_h \in V_h$ such that for all $w_h \in V_h$:

$$\int_\Omega D(u_h)\nabla u_h \cdot \nabla w_h \ dx = \int_\Omega f w_h \ dx$$

- Use appropriate quadrature rules for the nonlinear integrals
- Discrete system

$$A(u_h) = F(u_h)$$

Finite volume discretization for nonlinear diffusion

$$
\begin{aligned}
0 &= \int_{\omega_k} (-\nabla \cdot D(u)\nabla u - f)\, d\omega \\
&= -\int_{\partial\omega_k} D(u)\nabla u \cdot \mathbf{n}_k d\gamma - \int_{\omega_k} f d\omega && \text{(Gauss)} \\
&= -\sum_{L\in\mathcal{N}_k} \int_{\sigma_{kl}} D(u)\nabla u \cdot \mathbf{n}_{kl} d\gamma - \int_{\gamma_k} D(u)\nabla u \cdot \mathbf{n} d\gamma - \int_{\omega_k} f d\omega \\
&\approx \sum_{L\in\mathcal{N}_k} \frac{\sigma_{kl}}{h_{kl}} g_{kl}(u_k, u_l) + |\gamma_k|\alpha(u_k - w_k) - |\omega_k| f_k
\end{aligned}
$$

with

$$
g_{kl}(u_k, u_l) = \begin{cases} D(\frac{1}{2}(u_k + u_l))(u_k - u_l) \\ \text{or} \quad \mathcal{D}(u_k) - \mathcal{D}(u_l) \end{cases}
$$

where $\mathcal{D}(u) = \int_0^u D(\xi)\, d\xi$ (exact solution ansatz at discretization edge)

- Discrete system

$$
A(u_h) = F(u_h)
$$

Iterative solution methods: fixed point iteration

- Let $u \in \mathbb{R}^n$.
- Problem: $A(u) = f$:
- Assume $A(u) = M(u)u$, where for each u, $M(u) : \mathbb{R}^n \to \mathbb{R}^n$ is a linear operator.
- Iteration schem:

  Choose u_0, $i \leftarrow 0$;
  while *not converged* **do**
  | Solve $M(u_i)u_{i+1} = f$;
  | $i \leftarrow i + 1$;
  end

- Convergence criteria:
  - residual based: $||A(u) - f|| < \varepsilon$
  - update based $||u_{i+1} - u_i|| < \varepsilon$
- Large domain of convergence
- Convergence may be slow
- Smooth coefficients not necessary

Iterative solution methods: Newton method

- Solve

$$A(u) = \begin{pmatrix} A_1(u_1 \dots u_n) \\ A_2(u_1 \dots u_n) \\ \vdots \\ A_n(u_1 \dots u_n) \end{pmatrix} = \begin{pmatrix} f_1 \\ f_2 \\ \vdots \\ f_n \end{pmatrix} = f$$

- Jacobi matrix (Frechet derivative) for given u: $A'(u) = (a_{kl})$ with

$$a_{kl} = \frac{\partial}{\partial u_l} A_k(u_1 \dots u_n)$$

- Iteration scheme:

Choose u_0, $i \leftarrow 0$;
while *not converged* **do**
  Calculate residual $r_i = A(u_i) - f$;
  Calculate Jacobi matrix $A'(u_i)$;
  Solve update problem $A'(u_i)h_i = r_i$;
  Update solution: $u_{i+1} = u_i - h_i$;
  $i \leftarrow i + 1$;
end

Newton method II

- Convergence criteria: - residual based: $||r_i|| < \varepsilon$ - update based $||h_i|| < \varepsilon$
- Limited domain of convergence
- Slow initial convergence
- Fast (quadratic) convergence close to solution

Damped Newton method

- Remedy for small domain of convergence: damping

Choose u_0, $i \leftarrow 0$, damping parameter $d < 1$;
while *not converged* **do**
  Calculate residual $r_i = A(u_i) - f$;
  Calculate Jacobi matrix $A'(u_i)$;
  Solve update problem $A'(u_i)h_i = r_i$;
  Update solution: $u_{i+1} = u_i - d h_i$;
  $i \leftarrow i + 1$;
end

- Damping slows convergence down from quadratic to linear
- Better way: increase damping parameter during iteration:

Choose u_0, $i \leftarrow 0$,damping $d < 1$, growth factor $\delta > 1$;
while *not converged* **do**
  Calculate residual $r_i = A(u_i) - f$;
  Calculate Jacobi matrix $A'(u_i)$;
  Solve update problem $A'(u_i)h_i = r_i$;
  Update solution: $u_{i+1} = u_i - d h_i$;
  Update damping parameter: $d_{i+1} = \min(1, \delta d_i)$;
  $i \leftarrow i + 1$;
end

Newton method: further issues

- Even if it converges, in each iteration step we have to solve linear system of equations
  - Can be done iteratively, e.g. with the LU factorization of the Jacobi matrix from first solution step
  - Iterative solution accuracy my be relaxed, but this may diminuish quadratic convergence
- Quadratic convergence yields very accurate solution with no large additional effort: once we are in the quadratic convergence region, convergence is very fast
- Monotonicity test: check if residual grows, this is often an sign that the iteration will diverge anyway.

Newton method: embedding

- Embedding method for parameter dependent problems.
- Solve $A(u_\lambda, \lambda) = f$ for $\lambda = 1$.
- Assume $A(u_0, 0)$ can be easily solved.
- Parameter embedding method:

  Solve $A(u_0, 0) = f$;
  Choose initial step size δ;
  Set $\lambda = 0$;
  while $\lambda < 1$ **do**
  | Solve $A(u_{\lambda+\delta}, \lambda + \delta) = 0$ with initial value1 u_λ;
  | $\lambda \leftarrow \lambda + \delta$;
  end

- Possibly decrease stepsize if Newton's method does not converge, increase it later
- Parameter embedding + damping + update based convergence control go a long way to solve even strongly nonlinear problems!