Partial differential equations

= PDE


Examples for PDEs

Examples for scalar
boundary value problems
(elliptic eqs.)

*field* $\Phi$ depends on $\mathbf{x}$

**Poisson equation:**

$$\Delta \Phi = \rho(\mathbf{x}), \quad \Phi(\Gamma) = \Phi_0$$

Dirichlet boundary condition

**Laplace equation:**

$$\Delta \Phi = 0, \quad \nabla_n \Phi(\Gamma) = \Psi_0$$

von Neumann boundary condition
Examples for PDEs

example: vectorial boundary value problem

\[ \tilde{\mathbf{u}}(\tilde{x}) \] is a vector field defined on space

\[ \nabla (\nabla \tilde{\mathbf{u}}(\tilde{x})) + (1 - \nu) \Delta \tilde{\mathbf{u}}(\tilde{x}) = 0 \]

Lamé equation of elasticity
(elliptic eq.)

Examples for PDEs

wave equation

\[ \frac{\partial^2 \Phi}{\partial t^2} = c^2 \Delta \Phi , \quad \Phi(\tilde{x}, t_0) = \Phi_0(\tilde{x}) \]

diffusion equation

\[ \frac{\partial \Phi}{\partial t} = \kappa \Delta \Phi \]

initial boundary problem

\[ \Phi(\Gamma, t) = \Phi_0(t) \]
Examples for PDEs

\[ \mathbf{\vec{v}}(\mathbf{x}, t) \] vector field in space and time

\[ \frac{\partial \mathbf{\vec{v}}}{\partial t} + (\mathbf{\vec{v}} \nabla) \mathbf{\vec{v}} = -\frac{1}{\rho} \nabla p + \mu \nabla^2 \mathbf{\vec{v}}, \quad \nabla \cdot \mathbf{\vec{v}} = 0 \]

\[ \mathbf{\vec{v}}(\mathbf{x}, t_0) = \mathbf{\vec{V}}_0(\mathbf{x}), \quad p(\mathbf{x}, t_0) = P_0(\mathbf{x}) \]

\[ \mathbf{\vec{v}}(\Gamma, t) = \mathbf{\vec{v}}_0(t), \quad p(\Gamma, t) = p_0(t) \]

Navier – Stokes eq. for fluid motion

---

Discretization of space

\[ \Phi_{i,j} = \Phi(x_i, y_j) \]

\[ x_{i+1} = x_i + \Delta x \]

\[ y_{j+1} = y_j + \Delta y \]

Finite Difference Method
Discretization of derivatives

First derivative in 1d

\[ \Delta x \text{ small , } \, x_n = n \cdot \Delta x \]

\[ \frac{\partial \Phi}{\partial x} = \frac{\Phi(x_{n+1}) - \Phi(x_n)}{\Delta x} + O(\Delta x) \]

\[ = \frac{\Phi(x_n) - \Phi(x_{n-1})}{\Delta x} + O(\Delta x) \]

\[ = \frac{\Phi(x_{n+1}) - \Phi(x_{n-1})}{2 \Delta x} + O(\Delta x^2) \]

Second derivative in one dimension

\[ \frac{\partial^2 \Phi}{\partial x^2} = \frac{\Phi(x_{n+1}) + \Phi(x_{n-1}) - 2\Phi(x_n)}{\Delta x^2} + O(\Delta x^2) \]

or better

\[ \frac{\partial^2 \Phi}{\partial x^2} = \frac{-\Phi(x_{n-2}) + 16 \Phi(x_{n-1}) - 30 \Phi(x_n) + 16 \Phi(x_{n+1}) - \Phi(x_{n+2})}{12 \Delta x^2} + O(\Delta x^4) \]
Discretization of derivatives

\[ \frac{\partial^i \Phi}{\partial x^i} = \frac{1}{\Delta x^i} \sum_{k=-l}^{l} a_k \Phi(x_{n+k}) \]

Taylor expansion:

\[ \Phi(x_{n+k}) = \Phi(x_n) + k \Delta x \frac{\partial \Phi}{\partial x} (x_n) + \frac{k^2}{2} \Delta x^2 \frac{\partial^2 \Phi}{\partial x^2} (x_n) + \frac{k^3}{6} \Delta x^3 \frac{\partial^3 \Phi}{\partial x^3} (x_n) + O(\Delta x^4) \]

\[ i = 2 \implies \]

\[ \frac{\partial^2 \Phi}{\partial x^2} = \frac{-\Phi(x_{n-2}) + 16 \Phi(x_{n-1}) - 30 \Phi(x_n) + 16 \Phi(x_{n+1}) - \Phi(x_{n+2})}{12 \Delta x^2} + O(\Delta x^4) \]

third derivative

\[ \frac{\partial^3 \Phi}{\partial x^3} = \frac{-\Phi(x_{n-2}) + 2 \Phi(x_{n-1}) - 2 \Phi(x_{n+1}) + \Phi(x_{n+2})}{\Delta x^3} + O(\Delta x^2) \]

Derivatives in higher dimension

Be \( \Delta x = \Delta y = \Delta z \).

2 d

\[ \Delta \Phi \Delta x^2 = \Phi(x_{n+1}, y_n) + \Phi(x_{n-1}, y_n) + \Phi(x_n, y_{n+1}) + \Phi(x_n, y_{n-1}) - 4 \Phi(x_n, y_n) \]

3 d

\[ \Delta \Phi \Delta x^2 = \Phi(x_{n+1}, y_n, z_n) + \Phi(x_{n-1}, y_n, z_n) + \Phi(x_n, y_{n+1}, z_n) + \Phi(x_n, y_{n-1}, z_n) + \Phi(x_n, y_{n}, z_{n+1}) + \Phi(x_n, y_{n}, z_{n-1}) - 6 \Phi(x_n, y_n, z_n) \]
Poisson equation

\[ \Delta \Phi(\vec{x}) = \rho(\vec{x}) \]

discretize one-dimensional space by \( x_n \), \( n = 1, \ldots, N \) be \( \Phi_n \equiv \Phi(x_n) \)

discretization of the Poisson equation:

\[ \Phi_{n+1} + \Phi_{n-1} - 2 \Phi_n = \Delta x^2 \cdot \rho(x_n) \]

Dirichlet boundary conditions: \( \Phi_0 = c_0 \) and \( \Phi_N = c_1 \)

\[ \Rightarrow \text{System of } N-1 \text{ coupled linear equations} \]

---

Poisson equation in 1d

example: chain of \( N = 5 \) with \( \rho = 0 \)

and Dirichlet boundary conditions

\[
\begin{pmatrix}
-2 & 1 & 0 & 0 \\
1 & -2 & 1 & 0 \\
0 & 1 & -2 & 1 \\
0 & 0 & 1 & -2
\end{pmatrix}
\begin{pmatrix}
\Phi_1 \\
\Phi_2 \\
\Phi_3 \\
\Phi_4
\end{pmatrix}
= -
\begin{pmatrix}
c_0 \\
0 \\
0 \\
c_1
\end{pmatrix}
\]
Poisson equation in 2d

two-dimensional discretized equation on grid $L \times L$: 

$$ (\Delta x = \Delta y) $$

$$ \Phi_{i+1,j} + \Phi_{i-1,j} + \Phi_{i,j+1} + \Phi_{i,j-1} - 4 \Phi_{i,j} = \Delta x^2 \rho_{i,j} $$

replace indices $i$ and $j$ by $k = i + (j - 1) (L-2)$

$$ \Phi_{k+1} + \Phi_{k-1} + \Phi_{k+L-2} + \Phi_{k-L+2} - 4 \Phi_{k} = \Delta x^2 \rho_{k} $$

$\Rightarrow$ System of $N = (L-2)^2$ coupled linear equations:

$$ \vec{A} \cdot \vec{\Phi} = \vec{b} $$

Laplace equation in 2d

Example $5 \times 5$ lattice with $\rho = 0$ and $\Phi_m = \Phi_0$ for all $m \in \Gamma$,
i.e. Dirichlet boundary condition with fixed $\Phi_0$ on $\Gamma$.

$$ \begin{pmatrix} -4 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & -4 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & -4 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & -4 & 1 & 0 & 1 & 0 & 0 \\ 0 & : & (L-2)^2 \times (L-2)^2 \text{ matrix} & 0 \\ 0 & 0 & 1 & 0 & 1 & -4 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & -4 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & -4 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & -4 \end{pmatrix} \begin{pmatrix} \Phi_1 \\ \Phi_2 \\ \Phi_3 \\ \Phi_4 \\ \Phi_5 \\ \Phi_6 \\ \Phi_7 \\ \Phi_8 \\ \Phi_9 \end{pmatrix} = \begin{pmatrix} 2 \\ 1 \\ 2 \\ 1 \\ 0 \\ 1 \\ 2 \\ 1 \\ 2 \end{pmatrix} $$
Exact solution

\[
\begin{pmatrix}
  a_{11} \Phi_1 & \cdots & a_{1N} \Phi_N \\
  \vdots & \ddots & \vdots \\
  a_{N1} \Phi_1 & \cdots & a_{NN} \Phi_N
\end{pmatrix}
\begin{pmatrix}
  \Phi_1 \\
  \vdots \\
  \Phi_N
\end{pmatrix}
= 
\begin{pmatrix}
  b_1 \\
  \vdots \\
  b_N
\end{pmatrix}
\]

solution \( \Phi^* = A^{-1} \vec{b} \)

Gauss elimination procedure \( \Rightarrow \) matrix \( \vec{A} \) triangular

\[
q_{ik} = -\frac{a_{ik}}{a_{kk}} \quad \text{for } k = 1, \ldots, N
\]

\[
a'_{jl} = a_{jl} + q_{jk} a_{kl}, \quad \forall j, l > k
\]

\[
b'_{i} = b_{i} + q_{ik} b_{k}
\]

\( \Rightarrow O(N^3) \sim O(L^{3d}) \)

Poisson equation in 2d

Independently of the size of the system each row or column has only maximally five non-zero matrix elements \( \Rightarrow \) sparse matrix

Invert with LU decomposition

Use sparse matrix solvers!
Sparse matrices

Store non-zero elements in a vector and also their coordinates $i$ and $j$ in vectors.

⇒ Yale Sparse Matrix Format

example:

Hanwell Subroutine Library

For more details see:

www.cise.ufl.edu/research/sparse/codes

Iain Duff

Sparse matrix solvers

Table 1: Package features

<table>
<thead>
<tr>
<th>package</th>
<th>LIC</th>
<th>CHOLMOD</th>
<th>CSE</th>
<th>DSDPACK</th>
<th>GPLU</th>
<th>KLU</th>
<th>LD L</th>
<th>MA27</th>
<th>MA37</th>
<th>MA47</th>
<th>MA57</th>
<th>MA67</th>
<th>MA67MP2</th>
<th>MA67MF42</th>
<th>MA67MG</th>
<th>MA85</th>
<th>MA97</th>
<th>MA97C</th>
<th>MA97N2</th>
<th>MA97R2</th>
<th>MA97R3</th>
<th>MA97S2</th>
</tr>
</thead>
<tbody>
<tr>
<td>method</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 2: Package authors, references, and availability

<table>
<thead>
<tr>
<th>package</th>
<th>Authors, references</th>
<th>URL and/or context</th>
</tr>
</thead>
</table>
| CHOLMOD | Davis, Lovett, and Pier [6, 8, 9, 40] | www.cise.ufl.edu/research/sparse/CHOLMOD |}

www.cise.ufl.edu/research/sparse/codes
Quadtrees

Tree data structure where each node has up to four children corresponding to the four quadrants. That means that each node can contain several pointers indexed by two binary variables representing coordinates $i$ and $j$.

Computational considerations

Computational effort for Gauss elimination $\sim N^3$. For a lattice $100 \times 100 = 10^4$ one needs 2 days. \[\Rightarrow\] Abandon exact solution and use approximation. But for that $A$ must be well-conditioned:

example for ill-conditioned situation:

\[
\begin{pmatrix}
2.0 & 6.0 \\
2.0 & 6.00001
\end{pmatrix}
\begin{pmatrix}
x \\
y
\end{pmatrix}
= 
\begin{pmatrix}
8.0 \\
8.00001
\end{pmatrix}
\Rightarrow 
\begin{cases}
x = 1.0 \\
y = 1.0
\end{cases}
\]

\[
\begin{pmatrix}
2.0 & 6.0 \\
2.0 & 5.99999
\end{pmatrix}
\begin{pmatrix}
x \\
y
\end{pmatrix}
= 
\begin{pmatrix}
8.0 \\
8.00002
\end{pmatrix}
\Rightarrow 
\begin{cases}
x = 10.0 \\
y = -2.0
\end{cases}
\]
**Jacobi relaxation method**

example 2d Poisson equation: start with any \( \Phi_{ij}(0) \)

\[
\Phi_{ij}(t + 1) = \frac{1}{4} \left( \Phi_{i+1,j}(t) + \Phi_{i-1,j}(t) + \Phi_{i,j+1}(t) + \Phi_{i,j-1}(t) - b_{ij} \right)
\]

fixed point is the exact solution:

\[
\Delta \Phi(x, y) = b(x, y)
\]

\[
\Phi_{ij}^* = \frac{1}{4} \left( \Phi_{i+1,j}^* + \Phi_{i-1,j}^* + \Phi_{i,j+1}^* + \Phi_{i,j-1}^* - b_{ij} \right)
\]

<table>
<thead>
<tr>
<th>general:</th>
<th>decompose:</th>
<th>( \bar{A} \cdot \bar{\Phi} = \bar{b} )</th>
<th>( \bar{A} = \bar{D} + \bar{O} + \bar{U} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \Rightarrow ) ( \bar{D} \bar{\Phi} = \bar{b} - (\bar{O} + \bar{U}) \bar{\Phi} )</td>
<td>( \bar{\Phi}(t + 1) = \bar{D}^{-1} \left( \bar{b} - (\bar{O} + \bar{U}) \bar{\Phi}(t) \right) )</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Error of Jacobi relaxation**

Exact solution is only reached for \( t \rightarrow \infty \). Define required precision \( \varepsilon \) and stop when:

\[
\delta'(t + 1) \equiv \frac{\| \bar{\Phi}(t + 1) - \bar{\Phi}(t) \|}{\| \bar{\Phi}(t) \|} \leq \varepsilon
\]

real error:

\[
\bar{\delta}(t + 1) \equiv \frac{\bar{A}^{-1} \bar{b}}{\text{exact solution}} - \bar{\Phi}(t + 1) = \bar{A}^{-1} \bar{b} - \bar{D}^{-1} \left( \bar{b} - (\bar{O} + \bar{U}) \bar{\Phi}(t) \right)
\]

\[
= -\bar{D}^{-1} (\bar{O} + \bar{U}) \left( \bar{A}^{-1} \bar{b} - \bar{\Phi}(t) \right) = -\bar{D}^{-1} (\bar{O} + \bar{U}) \bar{\delta}(t)
\]
Error of Jacobi relaxation

\[ \tilde{\delta}(t + 1) = -\tilde{\Lambda} \cdot \tilde{\delta}(t) \quad \text{with} \quad \tilde{\Lambda} = \tilde{D}^{-1}(\tilde{O} + \tilde{U}) \]

be \( \lambda \) the largest eigenvalue of \( \tilde{\Lambda} \) \( \quad 0 < |\lambda| < 1 \)

for large \( t \):

\[ \tilde{\Phi}(t) \approx \tilde{\Phi}^* + \tilde{c} \lambda^t \]

\[ \frac{\|\tilde{\Phi}(t + 1) - \tilde{\Phi}(t)\|}{\|\tilde{\Phi}(t) - \tilde{\Phi}(t - 1)\|} \approx \frac{\lambda^{t+1} - \lambda^t}{\lambda^t - \lambda^{t-1}} = \lambda \]

---

Error of Jacobi relaxation

real error:

\[ \delta(t) = \frac{\|\tilde{\Phi}^* - \tilde{\Phi}(t)\|}{\|\tilde{\Phi}(t)\|} \approx \frac{\|\tilde{c}\|}{\|\tilde{\Phi}(t)\|} \lambda^t \]

\[ \delta'(t + 1) = \frac{\|\tilde{\Phi}(t + 1) - \tilde{\Phi}(t)\|}{\|\tilde{\Phi}(t)\|} \approx \frac{\|\tilde{c} (\lambda^{t+1} - \lambda^t)\|}{\|\tilde{\Phi}(t)\|} = \frac{\|\tilde{c}\|}{\|\tilde{\Phi}(t)\|} \lambda^t |\lambda - 1| \]

\[ \Rightarrow \quad \delta'(t + 1) = (1 - \lambda) \delta(t) \]

\[ \delta(t) = \frac{\delta'(t + 1)}{1 - \lambda} \approx \frac{\|\tilde{\Phi}(t) - \tilde{\Phi}(t - 1)\|^2}{\|\tilde{\Phi}(t)\|(\|\tilde{\Phi}(t) - \tilde{\Phi}(t - 1)\| - \|\tilde{\Phi}(t + 1) - \tilde{\Phi}(t)\|)} \]
Gauss-Seidel relaxation

\[
\Phi_i(t+1) = -\frac{1}{a_{ii}} \left( \sum_{j=i+1}^{N} a_{ij} \Phi_j(t) + \sum_{j=1}^{i-1} a_{ij} \Phi_j(t+1) - b_j \right)
\]

\[\bar{A} \cdot \Phi = \bar{b} \quad \text{and} \quad \bar{A} = \bar{D} + \bar{O} + \bar{U}\]

\[\Rightarrow (\bar{D} + \bar{O}) \Phi = \bar{b} - \bar{U} \cdot \Phi\]

\[\Phi(t+1) = (\bar{D} + \bar{O})^{-1} \left( \bar{b} - \bar{U} \cdot \Phi(t) \right)\]

fixed point is the exact solution

Error in Gauss-Seidel

\[
\delta(t+1) = \underbrace{\bar{A}^{-1} \bar{b}}_{\text{exact solution}} - \underbrace{(\bar{D} + \bar{O})^{-1} \left( \bar{b} - \bar{U} \cdot \Phi(t) \right)}_{\text{approximate solution}}
\]

\[= -(\bar{D} + \bar{O})^{-1} \bar{U} \left( \bar{A}^{-1} \bar{b} - \Phi(t) \right) = -(\bar{D} + \bar{O})^{-1} \bar{U} \cdot \delta(t)\]

\[\tilde{\delta}(t+1) = -\tilde{\Lambda} \cdot \tilde{\delta}(t) \quad \text{with} \quad \tilde{\Lambda} = (\bar{D} + \bar{O})^{-1} \bar{U}\]

\[\delta(t) = \frac{\|\Phi(t+1) - \Phi(t)\|}{(1 - \lambda) \|\Phi(t)\|} \leq \varepsilon \quad \lambda \text{ largest EV of } \tilde{\Lambda} \]
\[ \vec{A} \cdot \vec{\Phi} = \vec{b} \]

Jacobi relaxation

Gauss-Seidel relaxation

\[ \Phi_i(t+1) = -\frac{1}{a_{ii}} \left( \sum_{j=i+1}^{N} a_{ij} \Phi_j(t) + \sum_{j=1}^{i-1} a_{ij} \Phi_j(t+1) - b_j \right) \]

\[ \vec{A} \cdot \vec{\Phi} = \vec{b} \quad \text{and} \quad \vec{A} = \vec{D} + \vec{O} + \vec{U} \]

\[ \Rightarrow \quad (\vec{D} + \vec{O}) \vec{\Phi} = \vec{b} - \vec{U} \vec{\Phi} \]

\[ \vec{\Phi}(t+1) = (\vec{D} + \vec{O})^{-1} \left( \vec{b} - \vec{U} \vec{\Phi}(t) \right) \]

fixed point is the exact solution
Overrelaxation

Successive overrelaxation = SOR

$$\Phi(t+1) = (D + \omega O)^{-1} \left( \omega \vec{b} + [(1-\omega)D - \omega U] \Phi(t) \right)$$

Fixed point is the exact solution.
$$\omega$$ is the overrelaxation parameter.
$$1 \leq \omega < 2$$
$$\omega = 1$$ Gauss-Seidel relaxation

Non-linear problem

Consider a network of resistors with a non-linear I-U relation $$f$$.
Then Kirchhoff’s law takes the form:

$$f(U_{i+1,j} - U_{ij}) + f(U_{ij} - U_{i-1,j}) + f(U_{ij+1} - U_{ij}) + f(U_{ij} - U_{ij-1}) = 0$$

Solve with relaxation:

$$f(U_{i+1,j}(t) - U_{ij}(t+1)) + f(U_{ij}(t+1) - U_{i-1,j}(t)) + f(U_{ij+1}(t) - U_{ij}(t+1)) + f(U_{ij}(t+1) - U_{ij-1}(t)) = 0$$
Gradient methods

Be matrix $\hat{A}$ positive and symmetric.

The residuum $\mathbf{r} = \hat{A} \mathbf{\delta} = \hat{A} (\hat{A}^{-1} \mathbf{b} - \Phi) = \mathbf{b} - \hat{A} \Phi$ is a measure for the error.

Minimize the functional:

$$\mathcal{J} = \mathbf{r}^t \hat{A}^{-1} \mathbf{r} = \begin{cases} 0 & \text{if } \Phi = \Phi^* \\ > 0 & \text{otherwise} \end{cases}$$

Gradient methods

exact solution

$\mathcal{J}(\Phi)$

$\Phi^{(1)}$ $\Phi^{(2)}$
Gradient methods

\[ \mathcal{J} = (\mathbf{b} - \mathbf{A} \Phi)' \mathbf{A}^{-1} (\mathbf{b} - \mathbf{A} \Phi) = \mathbf{b}' \mathbf{A}^{-1} \mathbf{b} + \Phi' \mathbf{A} \Phi - 2 \mathbf{b} \Phi \]

Be \( \Phi_i \) the \( i \)th approximation.

Minimize along lines:

\[ \Phi = \Phi_i + \alpha_i \mathbf{d}_i \]

minimization condition with respect to \( \alpha_i \):

\[ \frac{\partial \mathcal{J}}{\partial \alpha_i} = 2 \mathbf{d}_i' \left( \mathbf{A} \mathbf{d}_i - \mathbf{r}_i \right) = 0 \quad \Rightarrow \quad \alpha_i = \frac{\mathbf{d}_i' \mathbf{r}_i}{\mathbf{d}_i' \mathbf{A} \mathbf{d}_i} \]

Method of steepest descent

Start with \( \Phi_1 \) and choose \( \mathbf{d}_i = \mathbf{r}_i \)

\( \mathbf{r}_1 = \mathbf{b} - \mathbf{A} \Phi_1 \)

iterate:

\[ \mathbf{u}_i = \mathbf{A} \mathbf{r}_i \]

\( \alpha_i = \frac{\mathbf{r}_i^2}{\mathbf{r}_i' \mathbf{u}_i} \)

\[ \Phi_{i+1} = \Phi_i + \alpha_i \mathbf{r}_i \]

\[ \mathbf{r}_{i+1} = \mathbf{r}_i + \alpha_i \mathbf{u}_i \]

each step \( \sim N^2 \), but when matrix \( \mathbf{A} \) sparse \( \sim N \)
Gradient methods

Conjugate gradient

Hestenes and Stiefel (1957)

Choose $\tilde{d}_i$ conjugate to each other:

$$\tilde{d}_i^T \tilde{A} \tilde{d}_j = 0 \quad \text{if} \quad i \neq j$$

as before:

$$\tilde{r}_i = \tilde{b} - \tilde{A} \Phi_i \quad , \quad \alpha_i = \frac{\tilde{r}_i \tilde{d}_i}{\tilde{d}_i^T \tilde{A} \tilde{d}_i} \quad , \quad \Phi_{i+1} = \Phi_i + \alpha_i \tilde{d}_i$$

$$\Rightarrow \quad \tilde{r}_i = \tilde{b} - \tilde{A} \left( \Phi_1 + \sum_{j=1}^{i-1} \alpha_j \tilde{d}_j \right)$$
**Conjugate gradient**

Construct conjugate basis using an orthogonalization procedure: (Gram – Schmidt)

\[ \vec{d}_1 = \vec{r}_1, \quad \vec{d}_i = \vec{r}_i - \sum_{j=1}^{i-1} \frac{\vec{d}_j^t \vec{A} \vec{r}_i}{\vec{d}_j^t \vec{A} \vec{d}_j} \vec{d}_j \]

one can also show:

\[ \vec{r}_i^t \vec{A} \vec{d}_j = 0 \text{ if } i \neq j \]

---

**Conjugate gradient**

1. initialize: \( \vec{r}_i = \vec{b} - \vec{A} \vec{\Phi}_1, \quad \vec{d}_1 = \vec{r}_1 \)

2. iterate:

\[ c = \left( \vec{d}_i^t \vec{A} \vec{d}_i \right)^{-1}, \quad \alpha_i = c \vec{d}_i \vec{r}_i, \quad \vec{\Phi}_{i+1} = \vec{\Phi}_i + \alpha_i \vec{d}_i \]

\[ \vec{r}_{i+1} = \vec{b} - \vec{A} \vec{\Phi}_{i+1}, \quad \vec{d}_{i+1} = \vec{r}_{i+1} - \left( c \vec{r}_{i+1} \vec{A} \vec{d}_i \right) \vec{d}_i \]

3. stop when: \( \vec{r}_i^t \vec{r}_i < \varepsilon \)
Conjugate gradient

If matrix not symmetric then use biconjugate gradient method.

Consider two residuals:

\[ \vec{r} = \vec{b} - \vec{A}\Phi \quad \text{and} \quad \tilde{\vec{r}} = \vec{b} - \vec{A}^T\Phi \]

This method does not always converge and can be unstable.

---

Biconjugate gradient

1. initialize:

\[ \vec{r}_1 = \vec{b} - \vec{A}\Phi_1, \quad \vec{d}_1 = \vec{r}_1 \]

\[ \tilde{\vec{r}}_1 = \vec{b} - \vec{A}^T\Phi_1, \quad \tilde{\vec{d}}_1 = \tilde{\vec{r}}_1 \]

2. iterate:

\[ \vec{r}_{i+1} = \vec{r}_i - \alpha_i \vec{A} \vec{d}_i, \quad \tilde{\vec{r}}_{i+1} = \tilde{\vec{r}}_i - \alpha_i \vec{A}^T\tilde{\vec{d}}_i, \quad \alpha_i = c \vec{r}_i^T\tilde{\vec{r}}_i \]

\[ \vec{d}_{i+1} = \vec{r}_i + \tilde{\alpha}_i \vec{d}_i, \quad \tilde{\vec{d}}_{i+1} = \tilde{\vec{r}}_i + \tilde{\alpha}_i \tilde{\vec{d}}_i, \quad \tilde{\alpha}_i = \tilde{c} \vec{r}_i^T\tilde{\vec{r}}_i \]

with \( c = (\vec{d}_i^T \vec{A} \vec{d}_i)^{-1} \) and \( \tilde{c} = (\tilde{\vec{d}}_i^T \vec{A}^T \tilde{\vec{d}}_i)^{-1} \)

3. stop when: \( \vec{r}_i^T \vec{r}_i < \varepsilon \) \( \Rightarrow \)

\[ \Phi_n = \Phi_1 + \sum_i^n \alpha_i \vec{d}_i \]
Choose a preconditioning matrix \( \tilde{P} \) such that
\[
\tilde{P}^{-1} \tilde{A} \approx \mathbb{I}
\]
and solve equation:
\[
(\tilde{P}^{-1} \tilde{A}) \tilde{\Phi} = \tilde{P}^{-1} \tilde{b}
\]
example: Jacobi preconditioner:
\[
P_{ij} = A_{ii} \delta_{ij} = \begin{cases} A_{ii} & \text{if } i = j \\ 0 & \text{otherwise} \end{cases} \Rightarrow P_{ij}^{-1} = \frac{1}{A_{ii} \delta_{ij}}
\]

example: SOR preconditioner:
\[
\tilde{P} = \left( \frac{\tilde{D}}{\omega} + \tilde{U} \right)^{-1} \frac{\omega}{2 - \omega} \tilde{D}^{-1} \left( \frac{\tilde{D}}{\omega} + \tilde{O} \right)
\]

\( \Rightarrow \) Preconditioned Conjugate Gradient
Multigrid procedure

Achi Brandt (1970)

Consider coarser lattices on which the long-wave errors are damped out.

\( h = 2 \)


**Strategy:** solve the equation for the error on the coarser lattice.

**Two-level procedure:**

1. Determine residuum \( \vec{r} \) on the original lattice.

\[
\vec{r}_n = \vec{b} - \vec{A} \vec{\Phi}_n , \quad \vec{\delta}_n = \vec{A}^{-1} \vec{r}_n
\]
Multigrid procedure

2. Define the residuum on the coarser lattice through a restriction operator \( \mathcal{R} \):
\[
\hat{r}_n = \mathcal{R} \mathbf{r}_n
\]

3. Then obtain the error on the coarser lattice solving equation:
\[
\hat{A} \hat{\delta}_{n+1} = \hat{r}_n
\]

4. Then get the error on the original lattice through an extension operator \( \mathcal{P} \):
\[
\hat{\delta}_{n+1} = \mathcal{P} \hat{\delta}_{n+1}
\]

5. Get new approximate solution through:
\[
\overrightarrow{\Phi}_{n+1} = \overrightarrow{\Phi}_n + \hat{\delta}_{n+1}
\]

In an \( m \)-level procedure one solves the equation only on the last (coarsest) level. On each level one can also smoothen the error using several Gauss-Seidel relaxation steps.
Multigrid procedure

Example for extension operator on square lattice:

bilinear interpolation

\[
P\begin{pmatrix}
\hat{r}_{2i,2j} = \hat{r}_{i,j} \\
\hat{r}_{2i+1,2j} = \frac{1}{2}(\hat{r}_{i,j} + \hat{r}_{i+1,j}) \\
\hat{r}_{2i,2j+1} = \frac{1}{2}(\hat{r}_{i,j} + \hat{r}_{i,j+1}) \\
\hat{r}_{2i+1,2j+1} = \frac{1}{4}(\hat{r}_{i,j} + \hat{r}_{i+1,j} + \hat{r}_{i,j+1} + \hat{r}_{i+1,j+1})
\end{pmatrix}
\]

Multigrid procedure

Corresponding restriction operator:

\[
R\begin{pmatrix}
\hat{r}_{i,j} = \frac{1}{4} r_{i,j} + \frac{1}{8} \left(r_{i+1,j} + r_{i-1,j} + r_{i,j+1} + r_{i,j-1}\right) \\
+ \frac{1}{16} \left(r_{i+1,j+1} + r_{i-1,j+1} + r_{i-1,j+1} + r_{i-1,j-1}\right)
\end{pmatrix}
\]

They are adjunct to each other, i.e.

\[
\sum_{x,y} P \hat{v}(x, y) \cdot u(x, y) = h^2 \sum_{\hat{x},\hat{y}} \hat{v}(\hat{x}, \hat{y}) \cdot R u(x, y)
\]
Multigrid procedure

One can also vary the protocol  
⇒ V-cycles, W-cycles, ...

Solving PDEs

discretize ⇒ system of coupled linear equations  
\[ \hat{A} \cdot \Phi = \hat{b} \]

- Finite difference methods:
  Field \( \Phi \) is discretized on sites: \( \Phi_i \).

- Finite element methods = FEM:
  Field \( \Phi \) is patched together from a discrete set of continuous functions.
The Fathers of FEM

J. Argyris  R.W. Clough  O.C. Zienkiewicz

Finite Elements at ETH

• Gerald Kress: Strukturanalyse mit FEM
• Christoph Schwab: Numerik der Dgln.
• Peter Arbenz: Introduction to FEM
• Pavel Hora: Grundlagen der nichtlinearen FEM
• Andrei Gusev: FEM in Solids and Structures
• Falk Wittel: Eine kurze Einführung in FEM
• Eleni Chatzi: Method of Finite Elements
Literature for FEM


Finite Elements

Strukturmechanik/Anwendung:

Properties of FEM

Clough (1960)

Advantage of finite elements over finite differences

• Irregular geometries
• Strongly inhomogeneous fields
• Moving boundaries
• Non-linear equations

adaptive meshing, e.g. triangulation

Adaptive meshing in 2d

triangulations with different resolution
Adaptive meshing in 3d

triangulation of a wheel-rim

One dimensional example

Poisson equation:

$$\frac{d^2 \Phi}{dx^2}(x) = -4\pi \rho(x) \quad \text{with} \quad \Phi(0) = \Phi(L) = 0$$

Expand in terms of localized basis functions $u_i$:

$$\Phi(x) = \sum_{i=1}^{\infty} a_i u_i(x) \approx \Phi_N(x) = \sum_{i=1}^{N} a_i u_i(x)$$
Define weight functions $w_j(x)$ and get $a_i$ from:

$$-\sum_{i=1}^{N} a_i \int_{0}^{L} \frac{\partial^2 u_i(x)}{\partial x^2} w_j(x) \, dx = 4\pi \int_{0}^{L} \rho(x) w_j(x) \, dx, \quad j = 1, \ldots, N$$

$w_j(x) = u_j(x)$ is called the Galerkin method.

$\Rightarrow$ system of linear equations

$$\vec{A} \cdot \vec{a} = \vec{b}$$

with

$$A_{ij} = -\int_{0}^{L} u_i''(x) w_j(x) \, dx = \int_{0}^{L} u_i'(x) w_j'(x) \, dx$$

and

$$b_j = 4\pi \int_{0}^{L} \rho(x) w_j(x) \, dx$$
One dimensional example

Example for basis functions $u_i(x)$ are hat functions centered around $x_i$:

$$\Delta x \equiv x_i - x_{i-1}$$

= „element“

$$u_i(x) = \begin{cases} 
    \frac{x - x_{i-1}}{\Delta x} & \text{for } x \in [x_{i-1}, x_i] \\
    \frac{x_{i+1} - x}{\Delta x} & \text{for } x \in [x_i, x_{i+1}] \\
    0 & \text{otherwise}
\end{cases}$$

$\Delta x = x_i - x_{i-1}$

$\Rightarrow$

$$A_{ij} = \int_0^L u_i(x) \, dx$$

Boundary conditions are automatically fulfilled because basis functions were zero at both ends.

If $\Phi(0) = \Phi_0$, $\Phi(L) = \Phi_1$

then use following decomposition:

$$\Phi_N(x) = \frac{1}{L} \left( \Phi_0 (L - x) + \Phi_1 x + \sum_{i=1}^{N} a_i u_i(x) \right)$$
Non-linear PDEs

1d example:

\[ \Phi(x) \frac{d^2 \Phi}{dx^2}(x) = -4\pi \rho(x) \]

Then solve:

\[ \int_0^L \left[ \Phi(x) \frac{d^2 \Phi}{dx^2}(x) + 4\pi \rho(x) \right] w_k(x) \, dx = 0 \]

i.e. the coupled non-linear system of equations:

\[ \sum_{i,j} A_{ijk} a_i a_j = b_k \]

with

\[ A_{ijk} = -\int_0^L u_i(x) u_j''(x) w_k(x) \, dx \]

Picard iteration

Start with a guess \( \Phi_0 \).

Solve linear equation for \( \Phi_1 \):

\[ \Phi_0(x) \frac{d^2 \Phi_1}{dx^2}(x) = -4\pi \rho(x) \]

Then iterate:

\[ \Phi_n(x) \frac{d^2 \Phi_{n+1}}{dx^2}(x) = -4\pi \rho(x) \]
Finite Elements

\[ \Delta \Phi(x, y) + a \Phi + b = 0 \]

Decompose in basis functions \( N_i \)

\[ \Phi(x, y) = \sum_{i=1}^{n} \Phi_i N_i(x, y) \]

Variational Approach

Minimize the functional: Argyris (1954)

\[
E = \iint_{G} \left( \frac{1}{2} (\nabla \Phi)^2 + \frac{1}{2} a \Phi^2 + b \Phi \right) dxdy + \int_{\Gamma} \left( \frac{\alpha}{2} \Phi^2 + \beta \Phi \right) ds
\]

\[
\delta E = \iint_{G} (\nabla \Phi \nabla \Phi + a \Phi \delta \Phi + b \delta \Phi) dxdy + \int_{\Gamma} (\alpha \Phi \delta \Phi + \beta \delta \Phi) ds
\]

first Green‘s theorem:

\[
\iint_{G} \nabla \Phi \nabla \Psi \ dxdy = - \iint_{G} \Psi \Delta \Phi \ dxdy + \int_{\Gamma} \frac{\partial \Phi}{\partial n} \Psi \ ds \quad \Rightarrow
\]

\[
\delta E = \iint_{G} \left( - \Delta \Phi + a \Phi + b \right) = 0 \ dxdy + \int_{\Gamma} \left( \alpha \Phi + \beta + \frac{\partial \Phi}{\partial n} \right) \delta \Phi \ ds = 0
\]
Variational Approach

\[ \Delta \Phi = a \Phi + b \]

First term of total energy

\[ E = \sum_{\text{elements } j} \iint_{G_j} \left( \frac{1}{2} (\nabla \Phi)^2 + \frac{1}{2} a \Phi^2 + b \Phi \right) dx dy \]

can be brought into the form:

\[ E = \frac{1}{2} \ddot{\Phi} A \ddot{\Phi} + \dddot{b} \dddot{\Phi} \]

Minimizing then gives:

\[ \frac{\partial E}{\partial \Phi} = 0 \quad \Rightarrow \quad \dddot{A} \dddot{\Phi} + \dddot{b} = 0 \]

Function on Element

Higher dimensions

In 2d define function over one element = triangle of the triangulation

e.g. linearly:

\[ \Phi(\vec{r}) \approx a_1 + a_2 x + a_3 y \]

or by a paraboloid:

\[ \Phi(\vec{r}) \approx a_1 + a_2 x + a_3 y + a_4 x^2 + a_5 xy + a_6 y^2 \]
Linear case

Standard Form

Transform any element $j$ into the standard form.

\[
\begin{align*}
x &= x_1 + (x_2 - x_1) \xi + (x_3 - x_1) \eta \\
y &= y_1 + (y_2 - y_1) \xi + (y_3 - y_1) \eta
\end{align*}
\]

\[
\begin{align*}
\eta &= \left( (y - y_1)(x_2 - x_1) - (x - x_1)(y_2 - y_1) \right) / D \\
\xi &= \left( (x - x_1)(y_3 - y_1) - (y - y_1)(x_3 - x_1) \right) / D \\
D &= (y_3 - y_1)(x_2 - x_1) - (x_3 - x_1)(y_2 - y_1)
\end{align*}
\]
## Coordinate transformation

\[ \nabla \Phi = \left( \frac{\partial \Phi}{\partial x}, \frac{\partial \Phi}{\partial y} \right) \rightarrow \nabla \Phi = \left( \frac{\partial \Phi}{\partial \xi} + \frac{\partial \Phi}{\partial \eta} \right) \left( \frac{\partial \Phi}{\partial x} \frac{\partial \xi}{\partial x} + \frac{\partial \Phi}{\partial y} \frac{\partial \xi}{\partial y} \right) \left( \frac{\partial \Phi}{\partial \eta} + \frac{\partial \Phi}{\partial \eta} \right) \]

\[ \begin{align*}
\frac{\partial \xi}{\partial x} &= \frac{y_3 - y_1}{D} \\
\frac{\partial \xi}{\partial y} &= -\frac{x_3 - x_1}{D} \\
\frac{\partial \eta}{\partial x} &= -\frac{y_2 - y_1}{D} \\
\frac{\partial \eta}{\partial y} &= \frac{x_2 - x_1}{D} 
\end{align*} \]

\[ \begin{align*}
\left( \frac{\partial \Phi}{\partial x} \right)^2 &= \left( \frac{\partial \Phi}{\partial \xi} \right)^2 + \left( \frac{\partial \Phi}{\partial \eta} \right)^2 \\
\left( \frac{\partial \Phi}{\partial y} \right)^2 &= \left( \frac{x_3 - x_1}{D} \right)^2 \Phi_\xi^2 - 2 \left( \frac{(y_3 - y_1)(y_2 - y_1)}{D^2} \right) \Phi_\xi \Phi_\eta + \left( \frac{(y_2 - y_1)}{D} \right)^2 \Phi_\eta^2 
\end{align*} \]

\[ \iint_{G_j} ...dxdy = \iint_T ...\det( \bar{J} ) \ d\xi d\eta \]

### Jacobi matrix

\[ \bar{J} = \begin{pmatrix}
\frac{\partial x}{\partial \xi} & \frac{\partial x}{\partial \eta} \\
\frac{\partial y}{\partial \xi} & \frac{\partial y}{\partial \eta}
\end{pmatrix} \]

\[ \det( \bar{J} ) = \frac{\partial x}{\partial \xi} \frac{\partial y}{\partial \eta} - \frac{\partial x}{\partial \eta} \frac{\partial y}{\partial \xi} \]

\[ = (x_2 - x_1)(y_3 - y_1) - (x_3 - x_1)(y_2 - y_1) = D \]
Coordinate transformation

Inserting gives for each element

\[ \iint_{G_j} \left( \Phi_x^2 + \Phi_y^2 \right) \, dx \, dy = \int_{\Gamma} \left( c_1 \Phi_\xi^2 + 2c_2 \Phi_\xi \Phi_\eta + c_3 \Phi_\eta^2 \right) \, d\xi \, d\eta \]

\[ \iint_{G_j} \left( \Phi_x^2 + \Phi_y^2 \right) \, dx \, dy = c_1 \iint_{\Gamma} \Phi_\xi^2 \, d\xi \, d\eta + 2c_2 \iint_{\Gamma} \Phi_\xi \Phi_\eta \, d\xi \, d\eta + c_3 \iint_{\Gamma} \Phi_\eta^2 \, d\xi \, d\eta \]

coefficients are only calculated once for each element.

\[
\begin{align*}
c_1 &= \frac{D}{D} + \frac{D}{D} \\
c_2 &= \frac{-\left(y_3 - y_1\right)\left(y_2 - y_1\right) - \left(x_3 - x_1\right)\left(x_2 - x_1\right)}{D} \\
c_3 &= \frac{\left(y_2 - y_1\right)^2 + \left(x_2 - x_1\right)^2}{D}
\end{align*}
\]

Basis functions

In 2d define function over one element = triangle of the triangulation

e.g. linearly: \( \Phi(\vec{r}) \approx a_1 + a_2 x + a_3 y \)

or by a paraboloid:

\( \Phi(\vec{r}) \approx a_1 + a_2 x + a_3 y + a_4 x^2 + a_5 xy + a_6 y^2 \)
Shape of basis functions

Basis functions

Decompose on standard element in basis functions $N_i$

$$\Phi(\xi, \eta) = \sum_{i=1}^{6} \Phi_i N_i(\xi, \eta) = \vec{\phi} \cdot \vec{N}(\xi, \eta)$$

$N_1 = (1 - \xi - \eta)(1 - 2\xi - 2\eta)$, $N_2 = \xi(2\xi - 1)$

$N_3 = \eta(2\eta - 1)$, $N_4 = 4\xi(1 - \xi - \eta)$

$N_5 = 4\xi\eta$, $N_6 = 4\eta(1 - \xi - \eta)$

$$\vec{\phi} = (\Phi_1, ..., \Phi_6)$$, $$\vec{N} = (N_1, ..., N_6)$$
Shape functions on square lattice

\[ N_1^e = \frac{1}{4} (\xi - 1)(\eta - 1) \xi \eta \]

\[ N_2^e = \frac{1}{4} (\xi^2 - 1)(\eta - 1) \]

\[ N_3^e = \frac{1}{4} (1 - \xi^2)\eta(\eta - 1) \quad \text{(back view)} \]

\[ N_4^e = (1 - \xi^2)(1 - \eta^2) \]

\[ \Phi(\vec{r}) \approx c_1 + c_2 x + c_3 y + c_4 x^2 + c_5 xy + c_6 y^2 + c_7 xy^2 + c_8 x^2 y + c_9 x^2 y^2 \]

Energy Integrals

Calculate the energy integrals on standard element

\[
I_1 = \iint_T \Phi_\xi^2 \xi d\xi d\eta = \iint_T \left( \vec{\Phi} \ \vec{N}_\xi (\xi, \eta) \right)^2 d\xi d\eta \\
= \iiint_T \vec{\Phi}' \ \vec{N}_\xi \vec{N}_\xi' \vec{\Phi} \ d\xi d\eta = \vec{\Phi}' \ \left[ \iiint_T \vec{N}_\xi \vec{N}_\xi' d\xi d\eta \right] \vec{\Phi}
\]

and analogously

\[
I_2 = \iint_T \Phi_\xi \Phi_\eta d\xi d\eta = \vec{\Phi}' \ \vec{S}_2 \vec{\Phi} \]

\[
I_3 = \iint_T \Phi_\eta^2 d\xi d\eta = \vec{\Phi}' \ \vec{S}_3 \vec{\Phi}
\]

defining matrices \( \vec{S}_1, \vec{S}_2 \) and \( \vec{S}_3 \) on standard triangle.
Rigidity Matrix

\[ \int \int_{G_j} (\nabla \Phi)^2 \, dxdy = \int \int_T \left( c_1 \Phi_\xi^2 + 2c_2 \Phi_\xi \Phi_\eta + c_3 \Phi_\eta^2 \right) \, d\xi d\eta = \bar{\phi}^t \bar{S} \bar{\phi} \]

defines the rigidity matrix \( \bar{S} \) for any element:

\[ \bar{S} = c_1 \bar{S}_1 + 2c_2 \bar{S}_2 + c_3 \bar{S}_3 \]

Mass Matrix

Analogously one defines the mass matrix \( \bar{M} \):

\[ \int \int_{G_j} a \, \Phi^2 \, dxdy = \int \int_T a \left( \bar{\phi}_j \bar{N}(\xi, \eta) \right)^2 \, D_j \, d\xi d\eta \]
\[ = \bar{\phi}_j^t a \int \int_T \bar{N} \bar{N}^t D_j \, d\xi d\eta \bar{\phi}_j \equiv \bar{\phi}_j^t \bar{M}_j \bar{\phi}_j \]

\[ E = \sum_{\text{elements } j} \int \int_{G_j} \left( (\nabla \Phi)^2 + a \, \Phi^2 \right) \, dxdy = \sum_{\text{elements } j} \bar{\phi}_j^t \left( \bar{S}_j + \bar{M}_j \right) \bar{\phi}_j \]

\( \Rightarrow \ E = \bar{\Phi}^t \bar{A} \bar{\Phi} \quad \text{with} \quad \bar{\Phi} = (\bar{\phi}_j) \quad \text{and} \quad \bar{A} = \sum_j \left( \bar{S}_j + \bar{M}_j \right) \)
Assembly of the Matrix

The elements must be joined such that the field is continuous.

This is done by identifying the values of the coefficients at each vertex for all elements that share this vertex.

Field term

\[
\iint_{G_j} b\Phi \, dx \, dy = \iiint_T b \, \tilde{\phi}_j \, \tilde{N}(\xi, \eta) \, D_j \, d\xi \, d\eta \\
= \tilde{\phi}_j \, b \iint_T \tilde{N}(\xi, \eta) \, D_j \, d\xi \, d\eta = \tilde{b}_j \tilde{\phi}_j
\]

\[\Rightarrow \quad \tilde{b}_j\]

\[E = \tilde{\Phi} \, \tilde{A} \, \tilde{\Phi} + \tilde{b} \, \tilde{\Phi} \quad \text{with} \quad \tilde{b} = \left(\tilde{b}_j\right)\]
Numerical task of FEM

\[ \vec{A} \Phi + \vec{b} = 0 \]

⇒ Solve system of \( N \) linear equations where \( N \) is the number of vertices. Matrix \( \vec{A} \) and vector \( \vec{b} \) only depend on the triangulation and on the basis functions and the unknowns are the coefficients \( \Phi = (\phi_i) \).

The connection between the elements gives off-diagonal terms in the matrix \( \vec{A} \).
Finally one must also include the boundary terms, which appear as before on the right side of the equation.

Applet
http://www.lnm.mw.tum.de/teaching/tmapplets/
Stresses in a hinge

Stresses in a clip
Network of trusses

Time dependent PDE’s

Simple example is heat equation:

\[
\frac{\partial T}{\partial t}(\bar{x}, t) = \frac{\kappa}{C\rho} \nabla^2 T(\bar{x}, t) + \frac{1}{C\rho} W(\bar{x}, t)
\]

*T* is temperature, *C* is specific heat

\( \rho \) is density, \( \kappa \) is thermal conductivity

and \( W \) are external sources or sinks.
Time dependent PDE’s

„line method“ in two dimensions:

\[
T(x_{ij}, t + \Delta t) = T(x_{ij}, t) + \frac{\kappa \Delta t}{C \rho \Delta x^2} \left( T(x_{i+1,j}, t) + T(x_{i-1,j}, t) \right) \\
+ T(x_{ij+1}, t) + T(x_{ij-1}, t) - 4T(x_{ij}, t) \right) + \frac{\Delta t}{C \rho} W(x_{ij}, t)
\]

clearly unstable if

\[
\frac{\kappa \Delta t}{C \rho \Delta x^2} \geq \frac{1}{4}
\]

Courant-Friedrichs-Lewy (CFL) condition (1928)

Unstable 1d parabolic PDE
Crank - Nicolson method

(1947)

**implicit algorithm**

\[
T(\bar{x}, t + \Delta t) = T(\bar{x}, t) + \frac{\kappa \Delta t}{2C \rho} \left[ \nabla^2 T(\bar{x}, t) + \nabla^2 T(\bar{x}, t + \Delta t) \right] \\
+ \frac{\Delta t}{2C \rho} \left( W(\bar{x}, t) + W(\bar{x}, t + \Delta t) \right)
\]

**define**

\[
\tilde{T}(t) = \left( T(x_n, t) \right), \quad \tilde{W}(t) = \left( W(x_n, t) \right), \quad n = 1, \ldots, L^2
\]

---

**Crank - Nicolson method**

Define operator \( O \)

\[
OT(x_n, t) = \frac{\kappa \Delta t}{C \rho \Delta x^2} \left[ T(x_{n+1}, t) + T(x_{n-1}, t) \right] \\
+ T(x_{n+L}, t) + T(x_{n-L}, t) - 4T(x_n, t)
\]

Then Crank – Nicolson becomes:

\[
T(\bar{x}, t + \Delta t) = T(\bar{x}, t) + \frac{1}{2} \left( OT(\bar{x}, t) + OT(\bar{x}, t + \Delta t) \right) \\
+ \frac{\Delta t}{2C \rho} \left( W(\bar{x}, t) + W(\bar{x}, t + \Delta t) \right)
\]
Crank - Nicolson method

\[ \begin{align*}
2 \ T(\bar{x}, t + \Delta t) &= 2 \ T(\bar{x}, t) + \left( OT(\bar{x}, t) + OT(\bar{x}, t + \Delta t) \right) \\
&\quad + \frac{\Delta t}{C \rho} \left( W(\bar{x}, t) + W(\bar{x}, t + \Delta t) \right)
\end{align*} \]

Then Crank – Nicolson becomes:

\[ \begin{align*}
(2 \cdot 1 - O) \ \widetilde{T}(t + \Delta t) &= (2 \cdot 1 + O) \ \widetilde{T}(t) + \frac{\Delta t}{C \rho} \left( \widetilde{W}(t) + \tilde{W}(t + \Delta t) \right)
\end{align*} \]

where 1 is the unity operator.

Crank - Nicolson method

Calculate the inverted operator \( B \) before:

\[ \begin{align*}
B &= (2 \cdot 1 - O)^{-1} \\
\widetilde{T}(t + \Delta t) &= B \left[ (2 \cdot 1 + O) \ \widetilde{T}(t) + \frac{\Delta t}{C \rho} \left( \widetilde{W}(t) + \tilde{W}(t + \Delta t) \right) \right]
\]
Crank - Nicolson method

Example: 1d diffusion equation:

Courant-Friedrichs-Lewy (CFL) number

\[ \frac{\partial u}{\partial t} = D \frac{\partial^2 u}{\partial x^2} \]

\[ \mu = \frac{D \Delta t}{2 (\Delta x)^2} \]

Crank-Nicholson discretization:

\[ u_i(t + \Delta t) - u_i(t) = \frac{D}{\Delta t} \left[ (u_{i+1}(t + \Delta t) - 2u_i(t + \Delta t) + u_{i-1}(t + \Delta t)) + (u_{i+1}(t) - 2u_i(t) + u_{i-1}(t)) \right] \]

\[ + \mu u_{i+1}(t + \Delta t) + (1 + 2\mu) u_i(t + \Delta t) - \mu u_{i-1}(t + \Delta t) \]

\[ = \mu u_{i+1}(t) + (1 - 2\mu) u_i(t) + \mu u_{i-1}(t) \]

Tridiagonal matrix problem

equation:

\[ a_i x_{i-1} + b_i x_i + c_i x_{i+1} = d_i \]

modify coefficients:

\[ c'_i = \begin{cases} \frac{c_i}{b_i} & ; i = 1 \\ \frac{c_i}{b_i - c_{i-1} a_i} & ; i = 2, 3, \ldots, n - 1 \end{cases} \]

\[ d'_i = \begin{cases} \frac{d_i}{b_i} & ; i = 1 \\ \frac{d_i - d'_{i-1} a_i}{b_i - c'_{i-1} a_i} & ; i = 2, 3, \ldots, n. \end{cases} \]

solution:

\[ x_n = d'_n \]

\[ x_i = d'_i - c'_i x_{i+1} ; i = n - 1, n - 2, \ldots, 1. \]

Algorithm goes like O(N) (instead of O(N^3) in Gauss elimination).
Wave equation

\[ \frac{\partial^2 y}{\partial t^2} = c^2 \nabla^2 y \quad \text{with} \quad c = \sqrt{\frac{k}{\rho}} \]

\[ \implies \frac{y(x_n, t_{k+1}) + y(x_n, t_{k-1}) - 2y(x_n, t_k)}{\Delta t^2} \approx c^2 \nabla^2 y(x_n, t_k) \]

\[ \implies y(x_n, t_{k+1}) = 2(1 - 2\lambda^2)y(x_n, t_k) - y(x_n, t_{k-1}) + \lambda^2 \left( y(x_{n+1}, t_k) + y(x_{n-1}, t_k) + y(x_{n+L}, t_k) + y(x_{n-L}, t_k) \right) \]

with \( \lambda = \frac{c \Delta t}{\Delta x} < 1/\sqrt{2} \)

which corresponds to cut off modes for wave lengths smaller than \( \lambda \).

Initialization

To start the iterations one needs to know the field at two times \( t \) and \( t-\Delta t \).

That means, one needs to know \( y(x_n,0) \) and \( \frac{\partial y}{\partial t}(x_n,0) \)

Set

\[ y(x_n, \Delta t) = y(x_n, 0) + \Delta t \frac{\partial y}{\partial t}(x_n, 0) \]

better

\[ y(x_n, \Delta t) = \left(1 - \lambda^2\right)y(x_n, 0) + \Delta t \frac{\partial y}{\partial t}(x_n, 0) \]

\[ + \frac{\lambda^2}{4} \left( y(x_{n+1}, 0) + y(x_{n-1}, 0) + y(x_{n+L}, 0) + y(x_{n-L}, 0) \right) \]

error \( O(\Delta t^2) \)
Solution of the wave equation

\[ \frac{\partial \tilde{v}}{\partial t} + (\tilde{v} \nabla) \tilde{v} = -\frac{1}{\rho} \nabla p + \mu \nabla^2 \tilde{v}, \quad \tilde{\nabla} \tilde{v} = 0 \]

Navier – Stokes equation

\[ \tilde{v}(\tilde{x}, t) \text{ velocity field, } p(\tilde{x}, t) \text{ pressure field} \]

Euler:

\[ \frac{\partial \tilde{v}}{\partial t} + (\tilde{v} \nabla) \tilde{v} = -\frac{1}{\rho} \nabla p \]

Stokes:

\[ \frac{\partial \tilde{v}}{\partial t} = -\frac{1}{\rho} \tilde{\nabla} p + \mu \nabla^2 \tilde{v} \]

equation of motion for incompressible fluid
Solvers for NS equation

- Penalty method with MAC
- Finite Volume Method (FLUENT, OpenFOAM)
- Spectral method
- Lattice Boltzmann (Ladd)
- Discrete methods: DPD, SPH, SRD, LGA,…
- k-ε model for turbulence

CFD = Computational Fluid Dynamics

Navier – Stokes equation

\[
\frac{\vec{v}_{k+1} - \vec{v}_k}{\Delta t} = -\nabla p_{k+1} - \mu \nabla^2 \vec{v}_k - (\vec{v}_k \cdot \nabla)\vec{v}_k
\]

Apply on both sides \(\nabla\):

\[
\frac{\nabla \vec{v}_{k+1} - \nabla \vec{v}_k}{\Delta t} = -\nabla^2 p_{k+1} - \mu \nabla^2 (\nabla \vec{v}_k) - \nabla (\vec{v}_k \cdot \nabla) \vec{v}_k
\]

Insert incompressibility condition:

\[
\nabla \cdot \vec{v}_{k+1} = \nabla \cdot \vec{v}_k = 0
\]
Navier – Stokes equation

\[ \nabla^2 p_{k+1} = -\nabla \left( (\vec{v}_k \cdot \nabla) \vec{v}_k \right) \]

Poisson equation \( \Rightarrow \) determine pressure \( p_{k+1} \)

To solve it, one needs boundary conditions for the pressure which one obtains projecting the NS equation on the boundary. This must be done numerically.

Operator splitting

Introduce auxiliary variable field \( \vec{v}^* \)

\[ \frac{\vec{v}_{k+1} - \vec{v}^* + \vec{v}^* - \vec{v}_k}{\Delta t} = -\nabla p_{k+1} - \mu \nabla^2 \vec{v}_k - (\vec{v}_k \cdot \nabla) \vec{v}_k \]

and split in two equations:

\[ \frac{\vec{v}^* - \vec{v}_k}{\Delta t} = -\mu \nabla^2 \vec{v}_k - (\vec{v}_k \cdot \nabla) \vec{v}_k \]

\[ \frac{\vec{v}_{k+1} - \vec{v}^*}{\Delta t} = -\nabla p_{k+1} \]
Operator splitting

Applying $\vec{\nabla}$ on

$$\frac{\vec{v}_{k+1} - \vec{v}^*}{\Delta t} = -\vec{\nabla}p_{k+1}$$

one obtains

$$\nabla^2 p_{k+1} = \frac{\vec{\nabla}\vec{v}^*}{\Delta t}$$

Projecting on the normal $\vec{n}$ to the boundary one obtains:

$$\frac{\partial p_{k+1}}{\partial n} \equiv (\vec{n} \cdot \vec{\nabla}) p_{k+1} = \frac{1}{\Delta t} \vec{n} \left( \vec{v}^* - \vec{v}_{k+1} \right)$$

Spatial discretization

MAC = Marker and Cell is a staggered lattice:

Place components of velocity on middle of edges and pressures in the centers of the cells.

h is the lattice spacing
Spatial discretization

\[(\nabla p)_{x,i+1/2,j} = \frac{1}{h}(p_{i+1,j} - p_{i,j})\]

\[\nabla^2 p_{i,j} = \frac{1}{h^2}(p_{i+1,j} + p_{i-1,j} + p_{i,j+1} + p_{i,j-1} - 4p_{i,j})\]

Spatial discretization

\[\nabla^2 p_{k+1} = \frac{\nabla \vec{v}^*}{\Delta t}\]

Poisson equation for the pressure \(p_{k+1}\) is solved on the centers of the cells (●) .
Spatial discretization

\[ \mathbf{v}_{k+1} = \mathbf{v}_k + \Delta t \left( -\nabla p_{k+1} - \mu \nabla^2 \mathbf{v}_k - (\nabla \cdot \mathbf{v}_k) \mathbf{v}_k \right) \]

The equations for the velocity components are solved on the edges.

\[
\left( \mathbf{v} \cdot \nabla \right) \mathbf{v}^x = \mathbf{v}^x \left( \partial / \partial x \right) \mathbf{v}^x + \mathbf{v}^y \left( \partial / \partial y \right) \mathbf{v}^x
\]
Flow around a vocal chord

$t = 8.300e-03 : \text{velocity}$

Sedimentation

Glass beads descending in silicon oil

comparing experiment and simulation
Finite Volume Method


Solve conservation law
\[ \frac{\partial v}{\partial t} (x, t) + \nabla f (v (x, t)) = g (v (x, t)) \]

in integral form
\[ \int_{G_i} \left( \frac{\partial v}{\partial t} + \nabla f (v) \right) dV = \int_{G_i} g (v) dV \]

using Green’s theorem:
\[ \int_{G_i} \frac{\partial v}{\partial t} dV + \int_{\partial G_i} f (v) \mathbf{n} dS = \int_{G_i} g (v) dV \]

Finite Volume Method

\[ \int_{G_i} \frac{\partial v}{\partial t} dV + \int_{\partial G_i} f (v) \mathbf{n} dS = 0 \]

change of value in volume \(i\)
\[ \frac{\partial v_i}{\partial t} = - \frac{1}{|G_i|} \int_{\partial G_i} f (v) \mathbf{n} dS \]
Forward-Time Central-Space (FTCS)

\[ \frac{\partial v}{\partial t} = -\nabla f \left( v, x, t, \frac{\partial^2 v}{\partial x^2} \right) \]

\[ \frac{v_i(t + \Delta t) - v_i(t)}{\Delta t} = -\nabla f \left( v, i, t, \frac{\partial^2 v}{\partial x^2} \right) \]

\( f \) is spatially discretized in a central difference scheme

\[ v_i(t + \Delta t) = v_i(t) - \frac{\Delta t}{2\Delta x} \left( f(v_{i+1}(t)) - f(v_{i-1}(t)) \right) \]

FTCS

Time evolution of the inviscid Euler equation using a forward time central space scheme
Lax-Friedrichs Scheme

Peter Lax

Kurt Friedrichs

\[ \nu_i(t + \Delta t) = \frac{1}{2} (\nu_{i+1}(t) + \nu_{i-1}(t)) - \frac{\Delta t}{2\Delta x} (f(\nu_{i+1}(t)) - f(\nu_{i-1}(t))) \]

Lax-Friedrichs Scheme

2d Euler equation with reflecting boundaries
Turbine with FLUENT

Airfoil with FLUENT
Shock waves

Solutions of parabolic equations which move with constant velocity and develop a sharp front.

typical initial condition: Riemann problem

example: tsunami
Godunov Scheme

Example

1d inviscid Burgers equation:

$$\frac{\partial \rho}{\partial t} + \rho \frac{\partial \rho}{\partial x} = 0$$

Sergei K. Godunov (1959)

$${\rho_i(t + \Delta t) = \rho_i(t) + \frac{\Delta t}{\Delta x} \left[ F(\rho_i(t - \Delta t), \rho_i(t)) - F(\rho_i(t), \rho_i(t + \Delta t)) \right]}$$

with

$$F(\rho_L, \rho_R) = \frac{g^2}{2}, \quad g = \begin{cases} \rho_L & \text{if } \rho_L > 0 \\ \rho_R & \text{if } \rho_R < 0 \\ 0 & \text{if } \rho_L \leq 0 < \rho_R \end{cases}$$

$$\bar{\rho} = \frac{(\rho_L + \rho_R)}{2}$$

1d Burgers equation

formation of shock wave
Spectral Methods

PDE solver for smooth solutions without adaptive meshing. Has excellent convergence properties.

Finite elements:
- basis functions: local smooth functions

Spectral methods:
- basis functions: global smooth functions

PDE:
\[ Lu(x, t) = f(u(x, t)) \]
with \( u(0, t) = u_B \) and \( u(x, 0) = u_I(x) \)

\( L \) differential operator
- e.g. \( Lu(x, t) = \left( \frac{\partial}{\partial t} - \frac{\partial}{\partial x} \right) u(x, t) \)

Expand in terms of basis functions \( \phi_i \):
\[
 u(x, t) = \sum_{i=1}^{\infty} a_i(t) \phi_i(x) \approx u_N(x, t) = \sum_{i=1}^{N} a_i(t) \phi_i(x)
\]
Define \( N \) (orthogonal) test functions \( w_j(x) \):

\[
\int_0^L \left[ Lu(x, t) + f(u(x, t)) \right] w_j(x) \, dx \, dt = 0, \quad j = 1, \ldots, N
\]

\( w_j(x) = \phi_j(x) \) is called the Galerkin method and \( w_j(x) = \delta(x-x_j) \) is called a collocation.

---

Example 1: 1d advection equation

\[
\frac{\partial u}{\partial t} - \frac{\partial u}{\partial x} = 0 \quad \text{on} \quad (0, 2\pi)
\]

truncated expansion:

\[
u^{(N)}(x, t) = \sum_{l=-N/2}^{N/2} a_l(t) \phi_l(x)
\]
Spectral Methods

trigonometric basis and test functions:

\[ \phi_l(x) = e^{ilx} \quad \text{and} \quad w_k(x) = \frac{1}{2\pi} e^{-ikx} \]

\[ \int_0^{2\pi} e^{i(l-k)x} \, dx = 2\pi \delta_{lk} \]

\[ \frac{1}{2\pi} \int_0^{2\pi} \left[ \left( \frac{\partial}{\partial t} - \frac{\partial}{\partial x} \right) \sum_{l=-N/2}^{N/2} a_l(t) e^{ilx} \right] e^{-ikx} \, dx = 0 \]

\[ \Rightarrow \quad \frac{da_k}{dt} - ika_k = 0 \quad \forall k = -\frac{N}{2}, \ldots, \frac{N}{2} \]

solve initial condition

\[ \frac{da_k}{dt} - ika_k = 0 \quad \text{with} \quad a_k(0) = \int_0^{2\pi} u_I(x) e^{-kx} \, dx \]

choose for instance

\[ u_I(x) = \sin(\pi \cos(x)) \]

\[ \Rightarrow \quad a_k(t) = \sin\left(\frac{k\pi}{2}\right) J_k(\pi) e^{ikt} \]
Spectral Methods

\[ a_k(t) = \sin\left(\frac{k\pi}{2}\right) J_k(\pi) e^{ikt} \]

From asymptotic behaviour of Bessel functions:

\[ \forall p: k^p a_k(t) \to 0 \text{ for } k \to \infty \]

\[ \Rightarrow \quad u^{(N)}(x,t) = \sum_{k=-N/2}^{N/2} a_k(t) e^{ikx} \]

converges faster than any power of \(1/N\).

Spectral Methods

Example 2: 1d (full) Burgers equation

\[ \partial_t u + u \partial_x u = \mu \partial_{xx} u \]

integral or «weak» form, \(\forall w, \forall t:\)

\[ \langle \partial_t u, w \rangle + \langle u \partial_x u, w \rangle = \mu \langle \partial_{xx} u, w \rangle \quad \text{with} \quad \langle f, w \rangle = \int_0^{2\pi} f(x)\bar{w}(x)dx \]

Fourier-Galerkian expansion

\[ u^{(N)}(x,t) = \sum_{k=-N/2}^{N/2} a_k(t) e^{ikx} \]

\[ w(x) = e^{ikx}, \quad k = -\frac{N}{2}, \ldots, \frac{N}{2} \]
\[ \langle \partial_t u, w \rangle + \langle u \partial_x u, w \rangle = \mu \langle \partial_{xx} u, w \rangle \]

\[ \langle \partial_t u, e^{ikx} \rangle = \langle \partial_x \left( -\frac{1}{2} u^2 + \mu \partial_x u \right), e^{ikx} \rangle \]

integrating by parts:

\[ \langle \partial_t u, e^{ikx} \rangle = \left\langle \frac{1}{2} u^2 - \mu \partial_x u, \partial_x e^{ikx} \right\rangle = \left\langle \frac{1}{2} u^2 - \mu \partial_x u, ike^{ikx} \right\rangle \]

Spectral Methods

Integrating by parts gives:

\[ \left\langle \partial_t u, e^{ikx} \right\rangle = \left\langle \frac{1}{2} u^2 - \mu \partial_x u, ike^{ikx} \right\rangle = \left\langle \frac{1}{2} u^2 - \mu \partial_x u, ike^{ikx} \right\rangle \]

The integrals can be evaluated as:

\[ \left\langle e^{i\ell x}, e^{ikx} \right\rangle = \int_0^{2\pi} e^{i(l-k)x} \, dx = 2\pi \delta_{l,k} \]

and

\[ \left\langle \partial_x a_i(t), e^{ikx} \right\rangle = 2\pi \partial_x a_k \]

The final expression for the integral is:

\[ \left\langle \frac{1}{2} u^2 - \mu \partial_x u, ike^{ikx} \right\rangle = \left\langle \frac{1}{2} \sum_{l=-N/2}^{N/2} a_k a_l e^{i(l+m)x} - i\mu \sum_{l=-N/2}^{N/2} l a_l e^{ikx}, ike^{ikx} \right\rangle \]

\[ = -\frac{ik}{2} \left\langle \sum_{l,m} a_k a_l e^{i(l+m)x}, e^{ikx} \right\rangle - \mu k \left\langle \sum_l l a_l e^{ikx}, e^{ikx} \right\rangle = -i\pi k \sum_{l+m=k} a_m a_l - 2\pi \mu k^2 a_k \]
This system of coupled ODE can be solved e.g. with Runge Kutta using the Fourier transformed initial condition:

\[
a_k(0) = \frac{1}{2\pi} \left\langle u(x,0), e^{ikx} \right\rangle = \frac{1}{2\pi} \int_0^{2\pi} u(x,0)e^{-ikx} \, dx
\]

Spectral Methods with other basis functions

Fourier decomposition is good when functions are periodic. Families of orthogonal polynomials on [-1,1] are Legendre and Chebychev polynomials.

Legendre polynomials:
\[
\int_{-1}^{1} P_m(x)P_n(x) \, dx = \frac{2}{2n+1} \delta_{mn}
\]
- \(P_0(x) = 1\), \(P_1(x) = x\), \(P_2(x) = \frac{3}{2}x^2 - \frac{1}{2}\)

Chebyshev polynomials:
\[
\int_{-1}^{1} T_m(x)T_n(x) \frac{dx}{\sqrt{1-x^2}} = \frac{\pi}{2}(1 + \delta_{0m}) \delta_{mn}
\]
- \(T_0(x) = 1\), \(T_1(x) = x\), \(T_2(x) = 2x^2 - 1\)

Laguerre polynomials on [0,∞)
Hermite polynomials on (-∞, ∞)
Discrete fluid solvers

- Lattice Gas Automata (LGA)
- Lattice Boltzmann Method (LBM)
- Dissipative Particle Dynamics (DPD)
- Smooth Particle Hydrodynamics (SPH)
- Stochastic Rotation Dynamics (SRD)
- Direct Simulation Monte Carlo (DSMC)

Lattice gas Automata

- J.-P. Rivet and J.P. Boon, „Lattice Gas Hydrodynamics“ (Cambridge Univ. Press, 2001)
Lattice gas Automata

Particles move on a triangular lattice and follow the following collision rules:

Momentum is conserved at each collision. It can be proven (Chapman-Enskog) that its continuum limit is the Navier Stokes eq.
Each vector is an average over time of the velocities inside a square cell of 25 triangles.

Lattice gas Automata

Problem in three dimensions, because there exists no translationally invariant lattice which is locally isotropic. One must study the model in 4d and then project down to 3d. Start with 4d face centered hypercube that has 24 directions giving $2^{24} = 1677216$ possible states. Projecting onto a 3d hyperplane that already contains 12 directions adds another six new directions giving 18 in 3d.
Discrete fluid solvers

- Lattice Gas Automata (LGA)
- Lattice Boltzmann Method (LBM)
- Dissipative Particle Dynamics (DPD)
- Smooth Particle Hydrodynamics (SPH)
- Stochastic Rotation Dynamics (SRD)
- Direct Simulation Monte Carlo (DSMC)

Lattice Boltzmann

From LGCA to Lattice Boltzmann Models (LBM)

- (Boolean) molecules to (discrete) distributions
  \[ n_i \rightarrow f_i = \langle n_i \rangle \]
  \( n_i \) is the number of particles in a cell going in direction \( i \)
- (Lattice) Boltzmann equations (LBE)
  \[
  f_i(x + c_i, t + 1) - f_i(x, t) = C_i(f)
  \]

Boltzmann equation

**Distribution Function**

\[ f(\mathbf{x}, \mathbf{v}, t) \Delta \mathbf{x} \Delta \mathbf{v} \] is the number of particles having at time \( t \) velocities between \( \mathbf{v} \) and \( \mathbf{v} + \Delta \mathbf{v} \) in the elementary volume between \( \mathbf{x} \) and \( \mathbf{x} + \Delta \mathbf{x} \).

Taylor expansion:

\[
f(\mathbf{x} + \Delta \mathbf{x}, \mathbf{v} + \Delta \mathbf{v}, t + \Delta t) = f(\mathbf{x}, \mathbf{v}, t) + \Delta t \partial_t f + \Delta \mathbf{x} \partial_x f + \Delta \mathbf{v} \partial_{\mathbf{v}} f
\]

\[
\lim_{\Delta t \to 0} \frac{f(\mathbf{x} + \Delta \mathbf{x}, \mathbf{v} + \Delta \mathbf{v}, t + \Delta t) - f(\mathbf{x}, \mathbf{v}, t)}{\Delta t} = \partial_t f + \mathbf{v} \cdot \partial_{\mathbf{v}} f + \mathbf{a} \cdot \partial_{\mathbf{v}} f
\]

\[
\mathbf{a} = \lim_{\Delta t \to 0} \frac{\Delta \mathbf{v}}{\Delta t}
\]

Boltzmann equation

Due to collisions between particles in the volume \( \Delta \mathbf{x} \) during the time interval \( \Delta t \)

some additional \( \Delta f_{\text{coll}}^+ (\mathbf{x}, \mathbf{v}, t) \) particles acquire velocities between \( \mathbf{v} \) and \( \mathbf{v} + \Delta \mathbf{v} \) and

some \( \Delta f_{\text{coll}}^- (\mathbf{x}, \mathbf{v}, t) \) particles do not anymore have velocities between \( \mathbf{v} \) and \( \mathbf{v} + \Delta \mathbf{v} \), giving the collision term:

\[
\Omega_{\text{coll}} = \Delta f_{\text{coll}}^+ (\mathbf{x}, \mathbf{v}, t) - \Delta f_{\text{coll}}^- (\mathbf{x}, \mathbf{v}, t)
\]
Boltzmann equation

This gives the Boltzmann equation:

$$\partial_t f + \mathbf{v} \cdot \nabla_x f + \mathbf{a} \cdot \nabla_v f = \Omega_{\text{coll}}$$

In thermal equilibrium one expects the Maxwell-Boltzmann distribution:

$$f^{eq} = \frac{\rho_n}{\sqrt{2\pi kT}} e^{-\frac{(\mathbf{v} - \mathbf{u})(\mathbf{v} - \mathbf{u})}{2kT/m}}$$

BGK collision term


BGK model:

$$\Omega_{\text{coll}} = \frac{f - f^{eq}}{\tau}$$

where $\tau$ is a relaxation time

$$\tau = \frac{\mu m}{kT} = \frac{\mu}{c_s^2}$$

$c_s$ is «sound speed»

$\mu$ is viscosity

$c_s^2 \equiv \frac{kT}{m}$
Averaged quantities

Moments of the velocity distribution:

mass density:
\[ \rho(\vec{x}, t) = \int m f(\vec{x}, \vec{v}, t) d\vec{v} \]
momentum density:
\[ \rho(\vec{x}, t) \vec{u}(\vec{x}, t) = \int m \vec{v} f(\vec{x}, \vec{v}, t) d\vec{v} \]
energy density:
\[ \rho(\vec{x}, t) e(\vec{x}, t) = \int m \left( \frac{(\vec{v} - \vec{u})^2}{2} \right) f(\vec{x}, \vec{v}, t) d\vec{v} \]

Knudsen number

Validity of the continuum description:
characteristic length of system \( L \) must be much larger than the mean free path \( l \) of the molecules (distance between two subsequent collisions).

\[ K = \frac{l}{L} \]

Navier-Stokes equation: \( 0.01 > K \)
Boltzmann equation: \( 0.005 > K \)
Chapman-Enskog expansion

\[ f = \sum_{n=0}^{\infty} K^n f^{(n)} \]

where the small parameter \( K \) is the Knudsen number

\[ f^{(0)} = f^{eq} \]

\[ \nabla_x = \sum_{n=1}^{\infty} K^n \nabla_x^{(n)} \]

\[ \frac{\partial}{\partial t} = \sum_{n=1}^{\infty} K^n \frac{\partial}{\partial t^{(n)}} \]

Chapman-Enskog

momentum conservation

\[ \frac{\partial}{\partial t^{(1)}} (\rho \tilde{u}) + \nabla \cdot (\rho \tilde{u} \otimes \tilde{u}) = -\nabla \cdot (\rho e) + \rho \tilde{a} \]

\[ \begin{align*}
\frac{\partial f^{(0)}}{\partial t^{(2)}} + \frac{\partial f^{(1)}}{\partial t^{(1)}} + \frac{\partial f^{(0)}}{\partial t^{(2)}} + \frac{\partial f^{(1)}}{\partial t^{(1)}} + \tilde{v} \nabla_x f^{(1)} + \tilde{a} \nabla_v f^{(1)} &= -\frac{1}{\tau} f^{(2)} \\
\end{align*} \]

Navier Stokes equation:

\[ \frac{\partial \rho \tilde{u}}{\partial t} + \nabla \Pi = 0 \quad , \quad \Pi_{xy} = \int \tilde{v} \otimes \tilde{v} \left( f^{eq} + \left(1 - \frac{1}{2\tau}\right) f^{(1)}\right) dv \]
**Gaussian quadrature theorem**

Be \( g(x) \) a polynomial of at most degree \( 2n+1 \)

\[
\int_{a}^{b} g(x) \ w(x) \ dx = \sum_{i=0}^{n} w_{i} \ g(x_{i})
\]

with \( w_{i} = \int_{a}^{b} w(x) \prod_{k \neq i}^{n} \frac{x - x_{k}}{x_{i} - x_{k}} \ dx \), \( i = 0, \ldots, n \)

if for the positive weight function \( w(x) \) there exists a polynomial \( p(x) \) of degree \( n+1 \) such that

\[
\int_{a}^{b} x^{k} \ p(x) \ w(x) \ dx = 0 \quad , \quad \forall \ k = 0, \ldots, n
\]

and \( x_{i}, \ i = 0, \ldots, n \) are the zeros of \( p(x) \).

---

**Lattice Boltzmann**

\[
f^{\text{eq}} = \frac{\rho}{m (2 \pi kT/m)^{d/2}} e^{-\left(\frac{\bar{v} - \bar{u}}{2kT/m}\right)^{2}}\]

small parameter:

\[
\left\|\bar{u}\right\| c_{s}^{2} \equiv \frac{kT}{m}
\]

\[
f^{\text{eq}} \approx \frac{\rho}{m (2 \pi c_{s}^{2})^{d/2}} e^{-\frac{\bar{v}^{2}}{2c_{s}^{2}}} \left[ 1 + \frac{\bar{v}u}{c_{s}^{2}} + \frac{(\bar{v}u)^{2}}{2c_{s}^{4}} - \frac{\bar{u}^{2}}{2c_{s}^{2}} \right]
\]

\[
w(x) = \sum_{i=0}^{2} a_{i} H_{i}(\bar{v})
\]

\[
p(x) = \sum_{i=0}^{2} a_{i} H_{i}(\bar{v})
\]
Hermite Polynomials

\[ H_0(x) = 1, \quad H_1(x) = x, \quad H_2(x) = x^2 - 1, \quad H_3(x) = x^3 - 3x \]

\[ \int_{-\infty}^{\infty} H_i(x) H_j(x) e^{-x^2} dx = \sqrt{2\pi} i! \delta_{ij} \]

Lattice Boltzmann

one dimensional case:

\[ w(v) = \frac{1}{\sqrt{2\pi c_s^2}} e^{-\frac{v^2}{2c_s^2}} \]

\[ n + 1 = 3 \quad v_i = -\frac{1}{\sqrt{3}}, 0, \frac{1}{\sqrt{3}} \]

\[ w_i = \frac{(n+1)!}{(n+1)^2 [H_n(v_i)]^2} = \begin{pmatrix} 1 & 2 & 1 \\ 6 & 3 & 6 \end{pmatrix}_{i=0,1,2} \]
Lattice Boltzmann

three dimensional case:

\[
\begin{align*}
\mathbf{e} & = -\frac{\mathbf{v}^2}{2c_s^2} \mathbf{e} - \frac{\mathbf{v}_x^2}{2c_s^2} \mathbf{e} - \frac{\mathbf{v}_y^2}{2c_s^2} \mathbf{e} - \frac{\mathbf{v}_z^2}{2c_s^2} \mathbf{e}
\end{align*}
\]

27 discrete velocity vectors

\[
\begin{align*}
\mathbf{w}_{(0,0,0)} &= \mathbf{w}_0 \mathbf{w}_0 \mathbf{w}_0 = \frac{8}{27} \\
\mathbf{w}_{(\pm 1/\sqrt{3}, 0, 0)} &= \mathbf{w}_{(0, \pm 1/\sqrt{5}, 0)} = \mathbf{w}_{(0,0, \pm 1/\sqrt{5})} = \mathbf{w}_{1/\sqrt{3}} \mathbf{w}_0 \mathbf{w}_0 = \frac{2}{27} \\
\mathbf{w}_{(\pm 1/\sqrt{3}, \pm 1/\sqrt{3}, 0)} &= \mathbf{w}_{(0, \pm 1/\sqrt{5}, \pm 1/\sqrt{5})} = \mathbf{w}_{(\pm 1/\sqrt{3}, 0, \pm 1/\sqrt{3})} = \mathbf{w}_{1/\sqrt{3}} \mathbf{w}_{1/\sqrt{3}} \mathbf{w}_0 = \frac{1}{54} \\
\mathbf{w}_{(\pm 1/\sqrt{3}, \pm 1/\sqrt{3}, \pm 1/\sqrt{3})} &= \mathbf{w}_{1/\sqrt{3}} \mathbf{w}_{1/\sqrt{3}} \mathbf{w}_{1/\sqrt{3}} = \frac{1}{216}
\end{align*}
\]

Lattice Boltzmann

D2Q9

\[
\begin{align*}
w_i = \begin{cases} 
4/9 & i = 0 \\
1/9 & i = 1, 2, 3, 4 \\
1/36 & i = 5, 6, 7, 8 
\end{cases}
\]

D3Q15

\[
\begin{align*}
w_i = \begin{cases} 
2/9 & i = 0 \\
1/9 & i = 1 - 6 \\
1/72 & i = 7 - 14 
\end{cases}
\]

D3Q19

\[
\begin{align*}
w_i = \begin{cases} 
1/3 & i = 0 \\
1/18 & i = 1 - 6 \\
1/36 & i = 7 - 18 
\end{cases}
\]

Lattice Boltzmann

Define on each site $x$ of a lattice on each outgoing bond $i$ a velocity distribution function $f(x, v_i, t)$ which is updated as:

$$f_i(x + v_i, v_i, t + 1) - f_i(x, v_i, t) + F_i(v_i) = \frac{1}{\tau} \left[ f_i^0 (\rho_n, u, T) - f_i(x, v_i, t) \right]$$

where the equilibrium distribution is defined as:

$$f_i^0 = \rho_n w_i \left[ 1 + \frac{3\vec{v}\vec{u}}{c_s^2} + \frac{9(\vec{v}\vec{u})^2}{2c_s^4} - \frac{3\vec{u}^2}{2c_s^2} \right]$$

Lattice Boltzmann discretization

CFL number

$$\frac{v|\Delta t}{|\Delta x|} = 1$$

$$\tau = \frac{\mu}{c_s^2} + \frac{\Delta t}{2}$$

$$f (\bar{x} + \Delta \bar{x}, \bar{v} + \Delta \bar{v}, t + \Delta t) - f (\bar{x}, \bar{v}, t) = \Delta t \left( \partial_t + \bar{v} \vec{\nabla} \right) f + \frac{\Delta t^2}{2} \left( \partial_t + \bar{v} \vec{\nabla} \right)^2 f$$
Multi-Relaxation-Time (MRT) LBM

\[ |f(\bar{x} + \vec{c}_\alpha \delta t, t + \delta t)|_\alpha - |f(\bar{x}, t)|_\alpha = -\sum_{j=0}^{N} S_j \langle \phi_j | \phi_j \rangle (m_j - m_j^{eq}) |\phi_j\rangle_\alpha \]

Orthogonal polynomials \( S_j \) is the inverse of a relaxation time.

Projections of the distribution \( m_j = \langle \phi_j | f \rangle \)

Chapman-Enskog expansion:

Shear viscosity

\[
\mu = c_s^2 \left( \frac{1}{s_{j,...,13}} - \frac{1}{2} \right)
\]

Bulk viscosity

\[
\xi = \frac{5 - 9 c_s^2}{9} \left( \frac{1}{s_2} - \frac{1}{2} \right)
\]

Lattice Boltzmann

Car design

Powerflow, EXA
Raising of a bubble

3d Rayleigh Benard

Rayleigh-Benard convection cell
$Ra = 1e7$
$Pr = 0.7$
Flow through porous medium in 2d using a NVidia GTX680

Surface Flow with Moving and Deforming Objects
Interfaces and free surfaces

Discrete fluid solvers

- Lattice Gas Automata (LGA)
- Lattice Boltzmann Method (LBM)
- Dissipative Particle Dynamics (DPD)
- Smooth Particle Hydrodynamics (SPH)
- Stochastic Rotation Dynamics (SRD)
- Direct Simulation Monte Carlo (DSMC)
Smooth Particle Hydrodynamics

- **SPH** describes a fluid by replacing its continuum properties with locally (smoothed) quantities at discrete Lagrangian locations ⇒ meshless
- **SPH** is based on integral interpolants (Lucy 1977, Gingold & Monaghan 1977, Liu 2003)

\[
A(\mathbf{r}) = \int_{\Omega} A(\mathbf{r}') W(\mathbf{r} - \mathbf{r}', h) d\mathbf{r}'
\]

\((W) is the smoothing kernel\)

- These can be approximated discretely by a summation interpolant

\[
A(\mathbf{r}) \approx \sum_{j=1}^{N} A(\mathbf{r}_j) W(\mathbf{r} - \mathbf{r}_j, h) \frac{m_j}{\rho_j}
\]

---

Smooth Particle Hydrodynamics

The kernel (or weighting Function)

- **Example:** quadratic kernel

\[
W(r, h) = \frac{3}{2\pi h^2} \left( \frac{1}{4} q^2 - q + 1 \right)
\]

\[q = \frac{r}{h}, \quad r = |\mathbf{r}_a - \mathbf{r}_b|\]
Smooth Particle Hydrodynamics

• Spatial gradients are approximated using a summation containing the gradient of the chosen kernel function

\[ \nabla A_i = \sum_j \frac{m_j}{\rho_j} A_j \nabla W_{ij} \]

\[ \rho_i (\nabla \cdot \mathbf{u})_i = \sum_j m_j \left( \mathbf{u}_i - \mathbf{u}_j \right) \cdot \nabla W_{ij} \]

• Advantages are:
  – spatial gradients of the data are calculated analytically
  – the characteristics of the method can be changed by using a different kernel

Smooth Particle Hydrodynamics

Equations of Motion

• Navier-Stokes equations:

\[ \frac{d \rho}{dt} = -\rho \nabla \cdot \mathbf{v} \]

\[ \frac{d \mathbf{v}}{dt} = -\frac{1}{\rho} \nabla p + \mu \nabla^2 \mathbf{u} + \mathbf{F}_i \]

• Recast in particle form as:

\[ \frac{d \mathbf{r}_i}{dt} = \mathbf{v}_i + \varepsilon \sum_j m_j \left( \frac{\mathbf{v}_{ji}}{\rho_{ji}} \right) W_{ij} \]

\[ \left( \frac{d m_i}{dt} = 0 \right) \]

\[ \frac{d \rho_i}{dt} = \sum_j m_j (\mathbf{v}_i - \mathbf{v}_j) \cdot \nabla W_{ij} \]

\[ \frac{d \mathbf{v}_i}{dt} = -\sum_j m_j \left( \frac{p_i}{\rho_i^2} + \frac{p_j}{\rho_j^2} + \Pi_{ij} \right) \nabla W_{ij} + \mathbf{F}_i \]
Simulation of free surface

Dwarf Galaxy Formation

THE FORMATION OF A BULGELESS GALAXY WITH A SHALLOW DARK MATTER CORE

Fabio Governato (University of Washington)
Chris Brook (University of Central Lancashire)
Lucio Mayer (ETH and University of Zurich) and the N-Body Shop

KEY: Blue: gas density map. The brighter regions represent gas that is actively forming stars. The clock shows the time from the Big Bang. The frame is 50,000 light years across.

Simulations were run on Columbia (NASA Advanced Supercomputing Center) and at ARSC
Discrete fluid solvers

- Lattice Gas Automata (LGA)
- Lattice Boltzmann Method (LBM)
- Dissipative Particle Dynamics (DPD)
- Smooth Particle Hydrodynamics (SPH)
- Stochastic Rotation Dynamics (SRD)
- Direct Simulation Monte Carlo (DSMC)

Stochastic Rotation Dynamics

Stochastic Rotation Dynamics (SRD)
- introduction of representative fluid particles
- collective interaction by rotation of local particle velocities
- very simple dynamics, but recovers hydrodynamics correctly
- Brownian motion is intrinsic

\[
\begin{align*}
\vec{x}'_n &= \vec{x}_n + \vec{v}_n \Delta t \\
\vec{v}'_n &= \vec{u} + \Omega (\vec{v}_n - \vec{u}) + \vec{g} \\
\Omega^\pm &= \begin{pmatrix} \cos \alpha & \pm \sin \alpha & 0 \\ \mp \sin \alpha & \cos \alpha & 0 \\ 0 & 0 & 1 \end{pmatrix} \\
\vec{u} &= \langle \vec{v}_n \rangle
\end{align*}
\]

Stochastic Rotation Dynamics

Example of two particles in cell:

Shift grid to impose Galilean invariance.

Shear flow
One particle in fluid

e.g. pull sphere through fluid

no-slip condition: \( \vec{v}_\Gamma = \vec{v}_{\text{particle}} \)

\[ \vec{v} \]

\[ \text{fluid} \]

\[ \Gamma \]

\[ \text{moving boundary condition} \]

create shear in fluid: exchange momentum

Drag force

\[ \vec{F}_D = \int_{\Gamma} \Theta d\vec{A} \]

drag force

(filloullis principle)

\[ \Theta_{ij} = -p\delta_{ij} + \eta \left( \frac{\partial v_i}{\partial x_j} + \frac{\partial v_j}{\partial x_i} \right) \]

stress tensor

\( \eta = \rho \mu \) is static viscosity
Homogeneous flow

Re << 1  
**Stokes law:**  

\[ F_D = 6\pi \eta R v \]  
(exact for Re = 0)

R is particle radius, v is relative velocity

Re >> 1  
**Newton’s law:**  

\[ F_D = 0.22\pi \rho R^2 v^2 \]

general drag law:  

\[ F_D = \frac{\pi \eta^2}{8 \rho} C_D \text{Re}^2 \]

\( C_D \) is the drag coefficient

Drag coefficient \( C_D \)
In velocity or pressure gradients: **Lift forces** are perpendicular to the direction of the external flow, important for wings of airplanes.

\[
L = C_L \times \rho \times \frac{V^2}{2} \times A
\]

\(C_L\) is „lift coefficient“

when particle rotates: **Magnus effect**
important for soccer

---

Many particles in fluids

- The fluid velocity field follows the incompressible Navier Stokes equations.
- Many industrial processes involve the transport of solid particles suspended in a fluid. The particles can be sand, colloids, polymers, etc.
- The particles are dragged by the fluid with a force:

\[
F_D = \frac{\pi}{8} \frac{\eta^2}{\rho} C_D \text{Re}^2
\]

simulating particles moving in a sheared fluid
Stokes limit

hydrodynamic interaction between the particles

\[ \vec{v}_i = \sum_{j \neq i} M_{ij} (\vec{r}_i - \vec{r}_j) \vec{v}_j \]

for Re = 0 mobility matrix exact

**Stokesian Dynamics** (Brady and Bossis)

invert a full matrix \(\Rightarrow\) only a few thousand particles

---

**Numerical techniques**

1. Calculate stress tensor directly by evaluating the gradients of the velocity field through interpolation on the numerical grid, e.g. using Chebychev polynomials.

2. Method of Fogelson and Peskin: Advect markers that were placed in the particle and then put springs between their new and their old position. These springs then pull the particle.
Method of A.L. Fogelson and C.S. Peskin:
Advect markers that were placed in the
particle and then put springs between
their new and their old position.
These springs then pull the particle.

Sedimentation

Glass beads descending in silicon oil

Comparing experiment and simulation
Sedimentation of platelets

Oblate ellipsoids descend in a fluid under the action of gravity.

This has applications in biology (blood), industry (paint) and geology (clay).

**θ = 0.15 in 3d**
Oral exams

Jan.22-Feb.02
2017

15 relevant questions

• Congruential and lagged-Fibonacci RN
• Definition of percolation
• Fractal dimension and sand-box method
• Hoshen-Kopelman algorithm
• Finite size scaling
• Integration with Monte Carlo
• Detailed balance and $MR^2T^2$
• Ising model
15 relevant questions

- Simulate random walk
- Euler method
- 2nd order Runge-Kutta
- 2nd order predictor-corrector
- Jacobi and Gauss-Seidel relaxation
- Gradient methods
- Strategy of finite elements, finite volumes and spectral methods

Next semester

402-0810 Computational Quantum Physics
Giuseppe Carleo and Philippe de Forcrand
Tuesday afternoon: V Di 14-16, U Di 16-18

402-0812 Computational Statistical Physics
Mirko Lukovic and Miller Mendoza
Friday morning: V Fr 11-13, U Fr 9-11

327-5102 Molecular Materials Modelling
Daniele Passerone
Friday afternoon: V Fr 14-16, U Fr 16-18
One particle quantum mechanics:
- scattering problem, time evolution
- shooting technique
- Numerov algorithm

Many particle systems:
- Fock space, etc (≈ 2 weeks theory)
- Hartree-Fock approximation
- density functional theory and electron structure (He & H₂)
- strongly correlated electrons
- Hubbard and T-J models
Computational Quantum Physics

- Lanczos method
- Path integral Monte Carlo
- Bosonic world lines
- QCD, lattice gauge theory
- Fermions, QFT

Molecular Materials Modelling

- Daniele Passerone

  - Friday afternoon; V Fr 14-16, U Fr 16-18

- Empirical potentials and transition rates
- Bio-force fields, charges, peptides
- Embedded atom models, Wilff's theorem
- Pair-correlation function with MD for neutron scattering
Molecular Materials Modelling

Melting temperature from phase coexistence
MO-theory, basic SCF, chemical reactions
Density functional theory, pseudopotentials
DFT on realistic systems, hybrids
Linear scaling, GPW
Electronic spectroscopies, STM
Bandstructure, graphene, free energies

Computational Statistical Physics

Mirko Lukovic and Miller Mendoza

Friday morning: V Fr 11-13, U Fr 9-11

Advanced Monte Carlo techniques:
continuous variables (XY, Heisenberg)
multi-spin coding, bit-manipulation
vectorization, parallelization
histogram methods, multi canonical
Computational Statistical Physics

Kawasaki dynamics, heat bath microcanonical, Creutz algorithm, Q2R critical slowing down, dynamical scaling cluster algorithms (Swendsen-Wang, Wolff) Monte Carlo Renormalization Group

Molecular Dynamics Simulations:
Verlet and leap frog methods linked cell method, Verlet tables

parallelization, realistic potentials Ewald sums, reaction field method Nose-Hoover thermostat, rescaling constant pressure MD, melting Discrete Elements, friction, inelasticity rotation and quaternions ab-initio calculations, Car Parinello