Simulation und wissenschaftliches Rechnen II (SiwiR II) Sommersemester 2017

Christoph Pflaum

Contents

1 Multigrid

1.1 Damped Jacobi Iteration

Let us consider the finite difference discretization of Poisson's equation $-\triangle u = f$ on $\Omega =]0,1[^2$ with Dirichlet boundary conditions.

This leads to a matrix equation

$$L_h x_h = f_h,$$

where the diagonal is

$$D = E\frac{4}{h^2}$$

and L_h has eigenvalues

$$\lambda_{\nu,\mu} = \frac{4}{h^2}\left(\sin^2\left(\frac{\pi\nu h}{2}\right) + \sin^2\left(\frac{\pi\mu h}{2}\right)\right)$$

and eigenvectors $e_{\nu,\mu}$, $\nu, \mu = 1, ..., m-1$.

1.1.1 Jacobi Method with Damping Parameter

Let us consider the iteration

$$x_h^{k+1} = (E - \frac{h^2}{8}L_h)x_h^k + \frac{h^2}{8}f_h.$$

The algebraic error satisfies

$$x_h^{k+1} - x_h \ = \ \left(E - \frac{h^2}{8}L_h\right)\ \left(x_h^k - x_h\right).$$

If the algebraic error is an eigenvector like

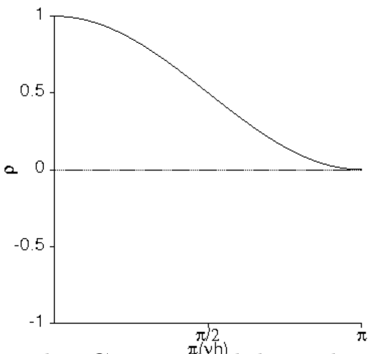
$$x_h^k - x_h = e_{\nu,\mu},$$

then we get for $\nu = \mu$

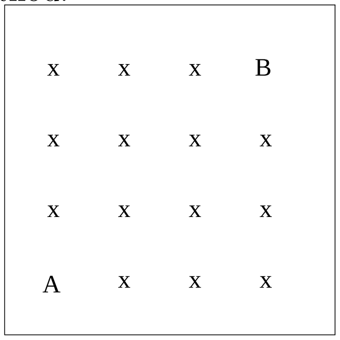
$$x_h^{k+1} - x_h \;=\; (1 - \frac{h^2}{8}\lambda_{\nu,\nu})\; e_{\nu,\nu} \;=\; \left(1 - \sin^2\left(\frac{\pi\nu h}{2}\right)\right)\;\left(x_h^k - x_h\right).$$

This means that the Jacobi Method with Damping Parameter
has the following properties

- Bad convergence for low frequencies.
- Good convergence for high frequencies.



The Gauss–Seidel method has similar properties as the damped Jacobi method.



Jacobi and Gauss-Seidel iteration need $O(\sqrt{n}) = O(h^{-1})$ operations for a correction in B due to a change of A.

The idea is to achieve a better correction by using coarser grids.

1.2 Multigrid algorithm on a Simple Structured Grid

Multigrid:

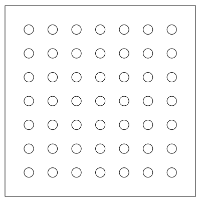


Figure 1: l=3

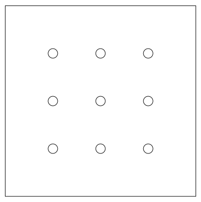


Figure 2: l=2

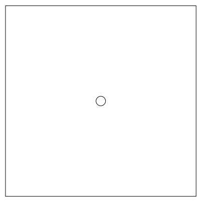


Figure 3: l=1

Let $l_{\max}$ be the number of levels such that $l_{\max} \in \mathbb{N}$ and

$$m_l = 2^l$$
$$n_l = (m_l - 1)^2$$
$$h_l = 2^{-l}$$

for $l = 1 \ldots l_{\max}$.

Let us assume that a PDE (e.g. Poisson's equation) is given. Discretize this equation by the grids $\Omega_l := \Omega_{h_l}$ where $l = 1, \ldots, l_{max}$. This leads to the discrete matrix equations

$$A_l x_l = b_l \tag{1}$$

where $b_l, x_l \in S_l$ and $S_l = \mathbb{R}^{n_l}$. The matrix A_l is an invertible matrix of order $n_l \times n_l$.

Let an iterative solver for (1) be given as

$$x_l{}^{k+1} \quad = \quad C_l{}^{relax} x_l{}^k + N_l b_l \quad = \quad \mathcal{S}_{l,b_l}(x_l^k) \tag{2}$$

Idea of Multigrid Algorithm:

Let $\widetilde{x}_l$ be an approximate solution for (1). The algebraic error $\widetilde{e}_l$ is defined as

$$\widetilde{e}_l = x_l - \widetilde{x}_l. \tag{3}$$

Now $\widetilde{e}_l$ has to be calculated in order to find x_l. The following residual equation is valid for $\widetilde{e}_l$,

$$A_l \widetilde{e}_l = r_l, \tag{4}$$

where r_l is called the residual and is given by

$$r_l = b_l - A_l \widetilde{x}_l. \tag{5}$$

The aim is to find an approximate solution of the residual equation by solving the equation approximately on a coarse grid Ω_{l-1}. To this end, we need the following matrix operators

- Restriction operator
$$I_l^{l-1} : S_l \mapsto S_{l-1}$$
- Prolongation operator
$$I_{l-1}^{l} : S_{l-1} \mapsto S_l$$

Two–grid Algorithm:

Two–grid Multigrid algorithm with parameters v_1 and v_2
Let x_l^k be an approximate solution of (1) and v_1 and v_2 the parameters of pre–smoothing and post–smoothing.

1. Step 1 (Pre–smoothing)
$$x_l^{k,1} = \mathcal{S}_{l,b_l}^{v_1} x_l^k \tag{6}$$
2. Step 2 (Coarse grid correction)
Residual calculation :
$$r_l = b_l - A_l x_l^{k,1} \tag{7}$$
Restriction :
$$r_{l-1} = I_l^{l-1} r_l \tag{8}$$
Solve on coarse grid:
$$e_{l-1} = {A_{l-1}}^{-1} r_{l-1} \tag{9}$$
Prolongation :
$$e_l = I_{l-1}^{l} e_{l-1} \tag{10}$$
Correction :
$$x_l^{k,2} = x_l^{k,1} + e_l \tag{11}$$

3. Step 3 (Post–smoothing)

$$x_l^{k+1} = \mathcal{S}_{l,b_l}^{\,v_2}(x_l^{k,2}) \tag{12}$$

Restriction and Prolongation Operators:

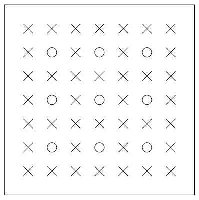


Figure 4: O–Coarse grid point and X+O–Fine grid point

Let us abbreviate $x_{i,j} = x_{(ih_{l-1}, jh_{l-1})}$ and set $x_{i,j} = 0$ for $i = 0$ or $j = 0$ or $i = m_{l-1}$ or $j = m_{l-1}$.

Prolongation or Interpolation:
The interpolation or prolongation of $x_{i,j}$ given by $w_{i,j} = \{I_{l-1}^{l}(x)\}_{(ih_l, jh_l)}$ is defined by the following equations

$$w_{2i,2j} = \frac{1}{2} x_{i,j} \tag{13}$$

$$w_{2i+1,2j} = \frac{1}{4}(x_{i,j} + x_{i+1,j}) \tag{14}$$

$$w_{2i,2j+1} = \frac{1}{4}(x_{i,j} + x_{i,j+1}) \tag{15}$$

$$w_{2i+1,2j+1} = \frac{1}{8}(x_{i,j} + x_{i+1,j} + x_{i,j+1} + x_{i+1,j+1}) \tag{16}$$

Pointwise Restriction:
Piecewise restriction is rarely applied and defined by

$$\{\dot{I}_l^{\,l-1}(x)\}_{(ih_{l-1}, jh_{l-1})} = x_{2i,2j} \tag{17}$$

The quality of this restriction operator is not very good.

Weighted Restriction:

Weighted restriction or full weighting is defined by

$$\begin{aligned}\{I_l^{\,l-1}(x)\}_{(ih_{l-1},jh_{l-1})} &= \frac{1}{8}(x_{2i+1,2j+1} + x_{2i-1,2j+1} + x_{2i+1,2j-1} + x_{2i-1,2j-1}) + \\ &\quad \frac{1}{4}(x_{2i+1,2j} + x_{2i-1,2j} + x_{2i,2j+1} + x_{2i,2j-1}) + \\ &\quad \frac{1}{2}x_{2i,2j}\end{aligned}$$

Remark

$$(I_l^{l-1})^T = I_{l-1}^l \qquad (18)$$

1.3 Multigrid Algorithm

Multigrid algorithm $MGM(x_l^k, b_l, l)$ with parameters (v_1,v_2,μ)

Let $x_{l_{max}}^k$ be an approximate solution of (1). Then,

$$x_{l_{max}}^{k+1} = MGM(x_{l_{max}}^k, x_{l_{max}}, l_{max})$$

is the approximate solution of (1) calculated by the multigrid algorithm with an initial vector $x_{l_{max}}^k$. The multigrid algorithm can then be described as

If $l = 1$ then $MGM(x_l^k, b_l, l) = A_l^{-1} b_l$.

If $l > 1$ then

Step 1 (v_1-pre–smoothing)

$x_l^{k,1} = \mathcal{S}_{l,b_l}^{\,v_1}(x_l^k)$

Step 2 (Coarse grid correction)

Residual : $r_l = b_l - A_l x_l^{k,1}$

Restriction : $r_{l-1} = I_l^{l-1} r_l$

Recursive call:

```
        $e_{l-1}^0 = 0$
        for $i = 1 \dots \mu$
            $e_{l-1}^i = MGM(e_{l-1}^{i-1}, r_{l-1}, l-1)$
        $e_{l-1} = e_{l-1}^{\mu}$
    Prolongation : $e_l = I_{l-1}^l e_{l-1}$
    Correction : $x_l^{k,2} = x_l^{k,1} + e_l$
```

Step 3 (v_2-post–smoothing)

$$MGM(x_l^k, b_l, l) = \mathcal{S}_{l,b_l}^{v_2}(x_l^{k,2})$$

V-cycle and W-cycle:
The algorithm $\mu = 1$ is called V-cycle.
The algorithm $\mu = 2$ is called W-cycle.
Homework: Describe the multigrid algorithm as a finite state machine, where every state is a smoothing step and an operation is a restriction or prolongation. Then, the finite state machine of a V-cycle looks like a "V" and the finite state machine of a W-cycle looks like a "W".

Convergence of Multigrid:
Let N be the number of unknowns. The computational amount of the V-cycle and W-cycle is $O(N)$.
The theory of multigrid algorithms shows that there is a constant ρ such that the convergence rate of the multigrid algorithm satisfies

$$\rho(C_{MGM,l}) \leq \rho < 1$$

independent of l.

1.4 Debugging of MG

- command out parts of the code (recursive coarse grid call, correction step, ...)
- often the coarse grid matrix is defined by

$$A_H := I_h^H A_h I_H^h, \quad I_h^H = (I_H^h)^T.$$

Then, the following equation must hold for all coarse grid vectors v, w:

$$v^T A_H w = (I_H^h v)^T A_h I_H^h w.$$

Test this equation for $w = \underline{1}$ and other simple test functions.

- In case of Neumann boundary conditions and Poisson's equation:

$$A_H \underline{1} = \underline{0}.$$

2 Finite Elements

2.1 Linear Elements in 1D

Definition 1. *q is a linear function on the interval $]a, b[$, if there exist $c, d \in \mathbb{R}$ such that*

$$q(x) = cx + d \quad \forall x \in]a, b[.$$

Let $h = \frac{1}{m}$, $m \in \mathbb{N}$.
Then, the space of functions

$$V_h := \{u_h \in \mathcal{C}([0,1]) \mid u_h|_{]ih,(i+1)h[} \text{ is linear } \forall i = 0, ..., m-1 \}$$

is called the finite element space of linear functions.
Define

$$\overset{\circ}{V}_h := \{u_h \in V_h \mid u_h(0) = u_h(1) = 0\}.$$

Let us consider Poisson's equation in 1D:

$$\begin{aligned} -u'' &= f \quad \text{on }]0,1[, \\ u(0) = u_0, &\qquad u(1) = u_1. \end{aligned}$$

Then, we get

$$\int_0^1 u' v_h' \, dx = \int_0^1 f v_h \, dx \quad \forall v_h \in \overset{\circ}{V}_h .$$

Geht Ablei- tung u'?? Wie sieht sie aus?

Definition 2. *Let $u_h \in V_h$ such that*

$$\begin{aligned} \int_0^1 u_h' v_h' \, dx &= \int_0^1 f v_h \, dx \quad \forall v_h \in \overset{\circ}{V}_h, \\ u_h(0) = u_0, &\qquad u_h(1) = u_1. \end{aligned}$$

u_h is called the finite element discretization with linear finite elements.

Let

$$\begin{aligned} \Omega_h &= \{ih \mid i = 0, ..., m\}, \\ \overset{\circ}{\Omega}_h &= \{ih \mid i = 1, ..., m-1\}. \end{aligned}$$

Definition 3. *The nodal basis of* $\overset{\circ}{V}_h$ *is*

$$v_{h1}, ..., v_{h(m-1)} \in \overset{\circ}{V}_h$$

where

$$v_p(q) = \delta_{pq} \quad \forall p, q \in \Omega_h.$$

Stiffness Matrix:

For reasons of simplicity let us assume $u_0 = u_1 = 0$. Then, let us write

$$u_h = \sum_{q \in \overset{\circ}{\Omega}_h} x_q v_q,$$

where $x_h = (x_q)_{q \in \overset{\circ}{\Omega}_h} \in \mathbb{R}^{m-1}$.
Define the 1D local stiffness matrix and load vector as follows

$$\begin{aligned} A_h &= \left(\int_0^1 v_q' v_p' \, dx \right)_{p,q \in \overset{\circ}{\Omega}_h} \\ f_h &= \left(\int_0^1 f v_p \, dx \right)_{p \in \overset{\circ}{\Omega}_h}. \end{aligned}$$

Then, we get

$$A_h x_h = f_h.$$

The local stiffness matrix and the load vector can be calculated exactly or numerically.
Numerical integration leads to a matrix equation

$$\tilde{A}_h x_h = \tilde{f}_h.$$

Example of Stiffness Matrices:

Let us consider linear finite elements in 1D.
The stiffness matrix corresponding to

$$\int_0^1 u'v'\ dx$$

is

$$A_h = \frac{1}{h}\begin{pmatrix} 2 & -1 & & & \\ -1 & 2 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 2 & -1 \\ & & & -1 & 2 \end{pmatrix}.$$

The stiffness matrix corresponding to

$$\int_0^1 u'v\ dx$$

is

$$A_h = \frac{1}{2}\begin{pmatrix} 0 & 1 & & & \\ -1 & 0 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 0 & 1 \\ & & & -1 & 0 \end{pmatrix}.$$

The corresponding operator is

$$u \mapsto u'.$$

The stiffness matrix corresponding to

$$\int_0^1 uv'\ dx$$

is

$$A_h = \frac{1}{2}\begin{pmatrix} 0 & -1 & & & \\ 1 & 0 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & 0 & -1 \\ & & & 1 & 0 \end{pmatrix}.$$

The corresponding operator is

$$u \mapsto -u'.$$

The stiffness matrix corresponding to

$$\int_0^1 uv \; dx$$

is

$$A_h \;=\; \frac{h}{6}\begin{pmatrix} 4 & 1 & & & \\ 1 & 4 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & 4 & 1 \\ & & & 1 & 4 \end{pmatrix}.$$

The corresponding operator is

$$u \mapsto u.$$

Example 1: Poisson's Equation in 1D:

$$\begin{aligned} -u'' \;&=\; f \quad \text{on }]0,1[, \\ u(0)=0, \qquad & u(1)=0. \end{aligned}$$

Discretize this equation by $A_h x_h = \tilde{f}_h$ where

$$A_h = \left(\int_0^1 v_q' v_p' \; dx\right)_{p,q\in\,\overset{\circ}{\Omega}_h}, \quad \tilde{f}_h = \left(\int_0^1 I_h(f) v_p \; dx\right)_{p\in\,\overset{\circ}{\Omega}_h}.$$

This means

$$\frac{1}{h}\begin{pmatrix} 2 & -1 & & & \\ -1 & 2 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 2 & -1 \\ & & & -1 & 2 \end{pmatrix}\begin{pmatrix} x_1 \\ \cdot \\ \cdot \\ \cdot \\ x_{m-1} \end{pmatrix} = \frac{h}{6}\begin{pmatrix} 4 & 1 & & & \\ 1 & 4 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & 4 & 1 \\ & & & 1 & 4 \end{pmatrix}\begin{pmatrix} f(h) \\ \cdot \\ \cdot \\ \cdot \\ f(1-h) \end{pmatrix}.$$

2.2 Streamline Diffusion in 1D

The discretization of

$$u \mapsto u'$$

by finite elements leads to a discretization similar to the central difference discretization

$$A_h = \frac{1}{2}\begin{pmatrix} 0 & 1 & & & \\ -1 & 0 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 0 & 1 \\ & & & -1 & 0 \end{pmatrix}.$$

How do we get something similar to FD upwind?
Consider the convection diffusion equation in 1D:

$$-u'' - bu' = f, \quad u(0) = u(1) = 0$$

Multiply this equation by $v = v_h - \rho h v_h' \operatorname{sgn} b$, where $v_h \in \overset{\circ}{V}_h$ and integrate. Assuming $b > 0$, this yields

$$\int_0^1 \big(u'v_h' + h\rho b u' v_h' - bu'v_h\big)\,dx + \rho h \int_0^1 u'' v_h' dx = \int_0^1 f(v_h - \rho h v_h')dx.$$

In the streamline diffusion discretization, we neglect the term of third order and replace u by u_h:

$$\int_0^1 \big((1 + h\rho b)u_h' v_h' - bu_h' v_h\big)\,dx = \int_0^1 f(v_h - \rho h v_h')dx.$$

Let $\rho = \frac{1}{2}$. Then, the stencil corresponding to the term

$$\int_0^1 \big(h\rho b u_h' v_h' - bu_h' v_h\big)\,dx = b\int_0^1 \big(\frac{1}{2}hu_h'v_h' - u_h'v_h\big)\,dx$$

is

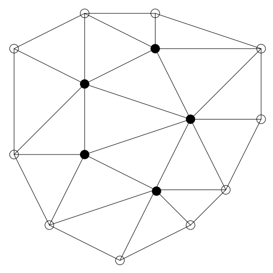
$$b\begin{pmatrix} 1 & -1 & & & \\ 0 & 1 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & 0 & 1 & -1 \\ & & & 0 & 1 \end{pmatrix}.$$

This shows that the finite element streamline diffusion discretization is similar to the FD upwind discretization.

2.3 Linear and Bilinear Finite Elements in 2D

Definition 4. *$\mathcal{T} = \{T_1, \dots, T_M\}$ is a conform triangulation of Ω if*

- $\overline{\Omega} = \bigcup_{i=1}^{M} T_i$, $\quad T_i$ *is a triangle or quadrangle (in 2D) or tetrahedron, hexahedron, prism, or pyramid (in 3D)*
- $T_i \cap T_j$ *is either*
  - *empty or*
  - *one common corner or*
  - *one common edge.*



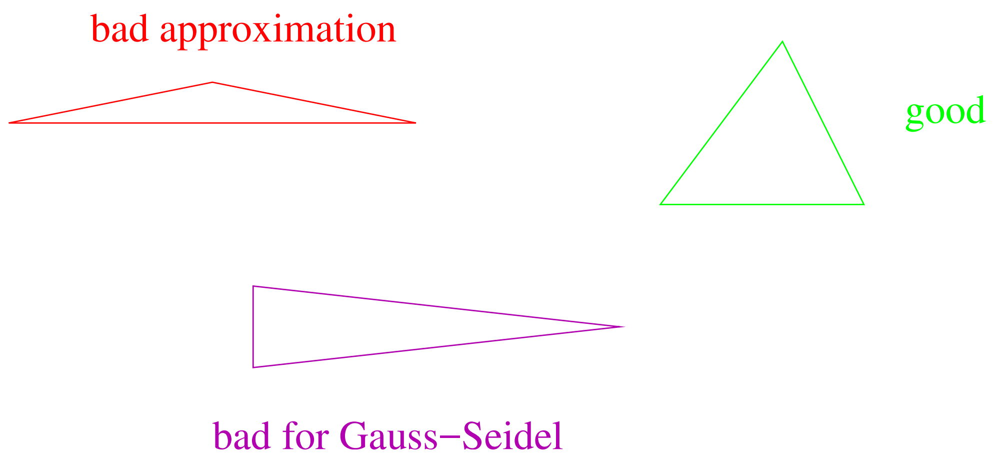
Remark.

- Let us write $\mathcal{T}_h$, if the diameter h_T of every element $T \in \mathcal{T}_h$ is less or equal h:

$$h_T \leq h.$$

- A family of triangulations $\{\mathcal{T}_h\}$ is called quasi-uniform, if there exists a constant $\rho > 0$ such that the radius ρ_T of the largest inner ball of every triangle $T \in \mathcal{T}_h$ satisfies

$$\rho_T > \rho h.$$



Definition 5. *Let $\mathcal{T}_h$ be a triangulation of Ω. Then, let V_h be the space of linear finite elements defined as follows:*

$$\begin{aligned} V_h &= \left\{ v \in C^0(\overline{\Omega}) \;\middle|\; v\big|_T \text{ is linear for every } T \in \mathcal{T}_h \right\} \\ \overset{\circ}{V}_h &= V_h \cap H_0^1(\Omega) \end{aligned}$$

$v\big|_T$ is linear means that $v\big|_T(x,y) = a + bx + cy$.

Definition 6 (Bilinear elements on a Cartesian 2D grid)**.** *Let $\Omega =]0,1[^2$, $h = \frac{1}{m}$ and*

$$\mathcal{T}_h = \left\{ [ih,(i+1)h] \times [jh,(j+1)h] \;\middle|\; i,j = 0,\dots,m-1 \right\}.$$

The space of bilinear finite elements on Ω is defined as follows

$$V_h = \left\{ v \in C^0(\overline{\Omega}) \;\middle|\; v\big|_T \text{ is bilinear for every } T \in \mathcal{T}_H \right\}.$$

$v\big|_T$ is bilinear means that $v\big|_T(x,y) = a + bx + cy + dxy$.

FE Discretization of Poisson's equation:

$$\begin{aligned} -\Delta u &= f \\ u\big|_{\delta\Omega} &= 0. \end{aligned}$$

Thus, for every $v_h \in \overset{\circ}{V}_h$, we get:

$$\begin{aligned}
-\Delta u \, v_h &= f \, v_h \\
&\Downarrow \\
\int_\Omega \nabla u \, \nabla v_h \, \mathrm{d}(x,y) - \int_\Gamma \frac{\partial u}{\partial \vec{n}} \, v_h \, \mathrm{d}(x,y) &= \int_\Omega f \, v_h \, \mathrm{d}(x,y) \\
&\Downarrow \\
\int_\Omega \nabla u \, \nabla v_h \, \mathrm{d}(x,y) &= \int_\Omega f \, v_h \, \mathrm{d}(x,y) \qquad \forall v_h \in \overset{\circ}{V}_h \, .
\end{aligned}$$

FE Discretization: Find $u_h \in \overset{\circ}{V}_h$ such that

$$\int_\Omega \nabla u_h \, \nabla v_h \, \mathrm{d}(x,y) = \int_\Omega f \, v_h \, \mathrm{d}(x,y) \qquad \forall v_h \in \overset{\circ}{V}_h \, . \tag{19}$$

Definition 7. *Let V_h be the space of linear or bilinear finite elements on $\mathcal{T}_h$ and $\mathcal{N}_h$ the set of corners of $\mathcal{T}_h$. Then, define the nodal basis function $v_q \in V_h$ at the point q by:*

$$v_q(x) = \begin{cases} 1 & \text{if } x = q \\ 0 & \text{if } x \neq q \end{cases} \qquad \text{for } x \in \mathcal{N}_h$$

Observe that

$$V_h = span\left\{ v_q \;\middle|\; q \in \mathcal{N}_h \right\}$$

This means that every function $u_h \in V_h$ can be represented as

$$u_h = \sum_{q \in \mathcal{N}_h} \lambda_q v_q$$

<u>Stiffness matrix:</u>

$$\begin{aligned}
a_{p.q} &:= \int_\Omega \nabla v_q \, \nabla v_p \, \mathrm{d}(x,y), \qquad f_p := \int_\Omega f \, v_p \, \mathrm{d}(x,y) \\
A_h &:= (a_{p,q})_{p,q \in \overset{0}{\mathcal{N}}_h}, \qquad \overset{0}{\mathcal{N}}_h := \mathcal{N}_h \cap \Omega \\
u_h &= \sum_{q \in \overset{0}{\mathcal{N}}_h} \lambda_q \, v_q.
\end{aligned}$$

Then, (19) implies

$$\sum_{q \in \overset{0}{\mathcal{N}}_h} \lambda_q \int_\Omega \nabla v_q \, \nabla v_p \; \mathrm{d}(x,y) \;=\; \int_\Omega f \; v_p \; \mathrm{d}(x,y) \qquad \text{for all } q \in \overset{0}{\mathcal{N}}_h$$

$$\Downarrow$$

$$\sum_{q \in \overset{0}{\mathcal{N}}_h} \lambda_q \, a_{p,q} \;=\; f_p \qquad \forall p \in \overset{0}{\mathcal{N}}_h$$

$$\Downarrow$$

$$A_h \, U_h \;=\; F_h \qquad \text{where} \qquad \begin{array}{l} U_h = (\lambda_q)_{q \in \overset{0}{\mathcal{N}}_h} \\ F_h = (f_p)_{p \in \overset{0}{\mathcal{N}}_h} \end{array}$$

The matrix A_h is called the stiffness matrix of the FE discretization.
<u>Example: Bilinear Elements on a Structured Grid</u>
Consider the structured grid on $\Omega =]0,1[^2$:

$$\mathcal{T}_h = \left\{ [ih,(i+1)h] \times [jh,(j+1)h] \;\middle|\; i,j = 0,\dots,m-1 \right\}.$$

$\mathcal{N}_h$ is the set of corresponding nodal points (corner points).
Observe that the nodal basis functions can be decomposed as

$$v_{p_x p_y}(x,y) = v_{p_x}(x) \cdot v_{p_y}(y).$$

Thus,

$$\begin{aligned} \int_\Omega \nabla v_{q_x q_y} \, \nabla v_{p_x p_y} \; \mathrm{d}(x,y) \;&=\; \int_0^1 \frac{\partial v_{p_x}}{\partial x} \frac{\partial v_{q_x}}{\partial x} \; \mathrm{d}x \int_0^1 v_{p_y} v_{q_y} \; \mathrm{d}y \\ &+\; \int_0^1 \frac{\partial v_{p_y}}{\partial y} \frac{\partial v_{q_y}}{\partial y} \; \mathrm{d}y \int_0^1 v_{p_x} v_{q_x} \; \mathrm{d}x. \end{aligned}$$

This shows that the discretization stencil for Poisson's equation is:

$$\begin{aligned} \frac{1}{h} \begin{pmatrix} -1 & 2 & -1 \end{pmatrix} \cdot \frac{h}{6} \begin{pmatrix} 1 \\ 4 \\ 1 \end{pmatrix} \;&+\; \frac{1}{h} \begin{pmatrix} -1 \\ 2 \\ -1 \end{pmatrix} \cdot \frac{h}{6} \begin{pmatrix} 1 & 4 & 1 \end{pmatrix} \\ &=\; \frac{1}{3} \begin{pmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{pmatrix} \end{aligned}$$

and for the right hand side the stencil is:

$$\frac{h}{6}\begin{pmatrix} 1 & 4 & 1 \end{pmatrix} \cdot \frac{h}{6}\begin{pmatrix} 1 \\ 4 \\ 1 \end{pmatrix} \quad = \quad \frac{h^2}{36}\begin{pmatrix} 1 & 4 & 1 \\ 4 & 16 & 4 \\ 1 & 4 & 1 \end{pmatrix}.$$

2.4 Calculation of the Stiffness Matrix

Since $\overline{\Omega} = \bigcup_{i=1}^{M} T_i$, we obtain

$$\int_\Omega \nabla v_q \; \nabla v_p \; \mathrm{d}(x,y) = \sum_{i=1}^{M} \int_{T_i} \nabla v_q \; \nabla v_p \; \mathrm{d}(x,y).$$

For linear or bilinear elements, we obtain

$$\int_{T_i} \nabla v_q \; \nabla v_p \; \mathrm{d}(x,y) \neq 0 \qquad \Leftrightarrow p, q \in T_i.$$

Definition 8. *The matrix*

$$\left(\int_{T_i} \nabla v_q \; \nabla v_p \; d(x,y) \right)_{p,q \in T_i}$$

is called local stiffness matrix at T_i.

Remark:
Observe that the local stiffness matrix is a 3×3 matrix for linear elements on triangles and a 4×4 matrix for bilinear elements.

Reference Element:

To calculate the local stiffness matrices we need a reference element $\hat{T}$ and a mapping

$$\Psi_i : \hat{T} \to T_i$$

for every i.

Example 1. *A reference element for triangles is:*

$$\hat{T} = \{(\xi, \eta) \mid \xi + \eta \leq 1 \quad \textit{and} \quad \xi, \eta \geq 0\}.$$

If T_i consists of the corners P_1, P_2, P_3, then

$$\Psi_i(\xi, \eta) = P_1 + (P_2 - P_1)\xi + (P_3 - P_1)\eta.$$

Example 2. *A reference element for quadrangles is:*

$$\hat{T} = \{(\xi, \eta) \mid 0 \leq \xi, \eta \leq 1\}.$$

Calculation of Local Stiffness Matrices:

Now, the local stiffness element can be calculated by

$$\begin{aligned}\int_{T_i} \nabla v_q^T \, \nabla v_p \, \mathrm{d}(x, y) &= \\ &= \int_{\hat{T}} \left((D\Psi_i)^{-T} \nabla \hat{v}_q\right)^T \left((D\Psi_i)^{-T} \nabla \hat{v}_p\right) |\det D\Psi_i| \ \mathrm{d}(\xi, \eta).\end{aligned}$$

Example 3. *Consider triangles. Then, describe the mapping Ψ_i by*

$$\Psi_i(\xi, \eta) = P_1 + \begin{pmatrix} a \\ b \end{pmatrix} \xi + \begin{pmatrix} c \\ d \end{pmatrix} \eta.$$

Then,

$$D\Psi_i = \begin{pmatrix} a & c \\ b & d \end{pmatrix}.$$

In case of complicated 3D elements like prisms, it may be that $(D\Psi_i)^{-T}$ is a rational polynomial, but no pure polynomial in ξ, η.

Numerical Integration:

Calculate the integral

$$\int_{\hat{T}} \left((D\Psi_i)^{-T} \nabla \hat{v}_q\right)^T \left((D\Psi_i)^{-T} \nabla \hat{v}_p\right) |\det D\Psi_i| \ \mathrm{d}(\xi, \eta).$$

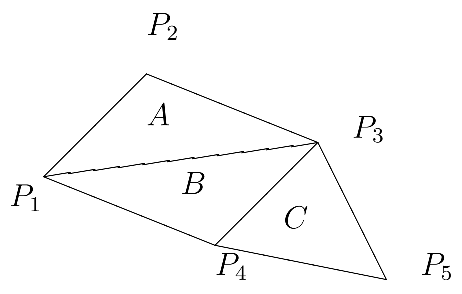
by Gauss quadrature rule.

Example 4. *Consider triangles and choose the above reference element. Then, the first Gauss quadrature rule implies:*

$$\begin{aligned}&\int_{\hat{T}} \left((D\Psi_i)^{-T} \nabla \hat{v}_q\right)^T \left((D\Psi_i)^{-T} \nabla \hat{v}_p\right) | \ det \ D\Psi_i| \ \ d(\xi, \eta) \\ &\approx \ \frac{1}{2} \left((D\Psi_i)^{-T} \nabla \hat{v}_q\right)^T \left((D\Psi_i)^{-T} \nabla \hat{v}_p\right) | \ det \ D\Psi_i| \left(\frac{1}{3}, \frac{1}{3}\right).\end{aligned}$$

Calculation of Stiffness Matrix:

Consider the triangulation:



The stiffness matrix of this triangulation is:

$$A_h =$$

$$\begin{pmatrix} l^A_{11}+l^B_{11} & l^A_{12} & l^A_{13}+l^B_{13} & l^B_{14} & 0 \\ l^A_{21} & l^A_{22} & l^A_{23} & 0 & 0 \\ l^A_{31}+l^B_{31} & l^A_{32} & l^A_{33}+l^B_{33} & l^B_{34}+l^C_{34} & l^C_{35} \\ l^B_{41} & 0 & l^B_{43}+l^C_{43} & l^B_{44}+l^C_{44} & l^C_{45} \\ 0 & 0 & l^C_{53} & l^C_{54} & l^C_{55} \end{pmatrix}$$

Observe:

$$\begin{aligned} \{1,2,3\} &= \mathcal{N}(A) \\ \{1,3,4\} &= \mathcal{N}(B) \\ \{4,3,5\} &= \mathcal{N}(C) \end{aligned}$$

Algorithmic Calculation of Stiffness Matrix:

The calculation of stiffness matrix has to be performed in two steps:

1. Step: Calculate the local stiffness matrix.
2. Step: Calculate the stiffness matrix.

But there exist two approaches:

1. First compute and store the whole local stiffness matrix.

   Then, calculate the stiffness matrix.

   Advantage: The local stiffness matrices can be used for coarsening the local stiffness matrices in a multigrid algorithm.

   Faster code for some non-linear problems.

2. After the calculation of the local stiffness matrix of one element T, add these integrals to the whole stiffness matrix.

   Advantage: Less storage requirement.

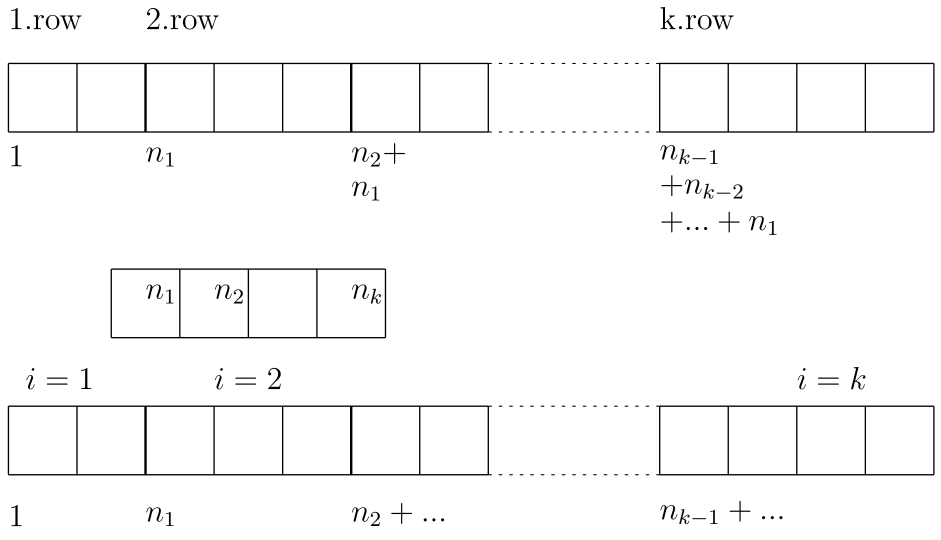
Data Structure for (Local) Stiffness Matrix:

1. Local stiffness matrix: Let n_T be the number of degrees of freedom for an element $T \in \mathcal{T}_h$ (3 for triangle). Then, for every $T \in \mathcal{T}_h$ a $n_T \times n_T$ matrix has to be stored.

   Data structure: list or array for storing $T \in \mathcal{T}_h$. Each entry must contain n_T and a pointer to a $n_T \times n_T$ matrix.

2. Stiffness matrix: For every unknown (grid point) the discretization stencil has no fixed size. Data structure: Sparse matrix.

Sparse Matrix Format:



Algorithm (Stiffness Matrix Calculation):
Let $\mathcal{N}(T)$ be the corner points of the triangle T.

1. Calculate local stiffness matrix $(l^T_{ij})_{i,j\in\mathcal{N}(T)}$ for every finite element $T \in \mathcal{T}_h$.

2. Calculate the number of neighbour points m_i for every point i. This gives the value $n_i = \sum_{s=1}^{i} m_s + 1$ in the sparse matrix of the the stiffness matrix.

3. Go to every grid point i and iterate over the neighbour element $T \in \mathcal{T}_h, i \in \mathcal{N}(T)$. Add the l^T_{ij} to a_{ij} for every $j \in \mathcal{N}(T)$.

⇒ Later we explain how to obtain a suitable data structure for the discretization grid.

Data Structure of the Discretization Grid:

Array (or list) of objects of type `Triangle`. Every triangle has an `id`.

```
class Triangulation_grid {
  int number_triangles;
  Triangle* triangles;   // id is number in list

  int number_points;
  Points* points;        // id is number in list
}
class Triangle {
  int id_point_1, id_point_2, id_point_3;
}
class Points {
  double x,y;  // coordinate of point
  int number_neighbour_points;
  int* id_neighbour_points;
}
```

In certain cases it is important to store the id of each neighbour triangle at every point.

Tafel: Schreib kurz Code für Iteration

2.5 General Formulation of PDE's and Convergence

Let V be a vector space and

$$a : V \times V \to \mathbb{R}$$

a symmetric positive definite bilinear form. a induces the “energy” norm

$$\|u\|_E = \sqrt{a(u,u)}.$$

Furthermore, let

$$f : V \to \mathbb{R}$$

be a $\|\cdot\|_E$ continuous linear functional and let V be complete with respect to $\|\cdot\|_E$.

Problem 1. *Find $u \in V$ such that* $\quad a(u,v) = f(v) \quad \forall v \in V.$

Theorem 1. *The above problem has a unique solution u.*

Examples of PDEs with Weak Formulation:

Example 5 (Poisson's Equation with Reaction Term)**.** *Let $V = H_0^1(\Omega) := \{u \in L^2(\Omega) \mid \nabla u \in L^2(\Omega)\}$ and $c \geq 0$.*

$$a(u,v) = \int_\Omega \big(\nabla u \nabla v + cuv\big)\, d(x,y).$$

Example 6 (Linear Elasticity)**.** *Let $V = (H_0^1(\Omega))^3, u \in V$, $\mathcal{C}$ a suitable 6×6 matrix and Du the vector of symmetric derivatives (see section 4).*

$$a(u,v) = \int_\Omega (Du)^T \mathcal{C} Dv \; d(x,y,z).$$

Example 7 (Maxwell's Equations)**.** *Let V be a suitable vector space similar to $(H_0^1(\Omega))^3$ and $c \geq 0$.*

$$a(u,v) = \int_\Omega (\nabla \times u)^T (\nabla \times v) + cuv \; d(x,y,z).$$

Let V_h be a subspace of V.

Problem 2. *Find $u_h \in V_h$ such that* $\quad a(u_h, v_h) = f(v_h) \quad \forall v_h \in V_h.$

Theorem 2.

$$\|u - u_h\|_E \leq \inf_{v_h \in V_h} \|u - v_h\|_E$$

Example 8. *Consider Poisson's equation. Let V_h be the space of linear elements corresponding to a a familiy of quasi-uniform triangulations. Furthermore, assume that $u \in C^2(\Omega)$ $(H^2(\Omega))$ is the weak solution of Poisson's equation. Then, there is a constant C such that*

$$\|u - u_h\|_E \leq hC$$

for every h, where $u_h \in V_h$ is the finite element solution.

Remark: In case of $H^2(\Omega)$-regularity, one can prove $\|u - u_h\|_{L^2} \leq h^2 C$.

2.6 Operator Formulation

Let V_h be a finite element space and $(v_q)_{q\in\mathcal{N}_h}$ the corresponding nodal basis. Let `u,f` be vectors of length $|\mathcal{N}_h|$. Then,

```
f = Laplace_FE(u);
```

means

$$\mathtt{f} = (f_p)_{p\in\mathcal{N}_h} = \left(\int_\Omega \nabla(\sum_{q\in\mathcal{N}_h} u_q v_q)\nabla v_p \; d(x,y)\right)_{p\in\mathcal{N}_h}$$

and

```
f = Helm_FE(u);
```

means

$$\mathtt{f} = (f_p)_{p\in\mathcal{N}_h} = \left(\int_\Omega (\sum_{q\in\mathcal{N}_h} u_q v_q) v_p \; d(x,y)\right)_{p\in\mathcal{N}_h} .$$

Thus,

```
Laplace_FE( ), Helm_FE( )
```

are operators. Let

```
Diag_Laplace_FE( ), Diag_Helm_FE( )
```

be the corresponding diagonal operators.
Let

```
interior, boundary
```

represent the interior and boundary points of the domain and

```
product(u,v)
```

the scalar product of `u` and `v`.
For testing you FE-code one can do the following tests:

Test 1: Volume Calculation:

Now let us implement the above operators by expression templates.
Then, the code

```
u = 1.0;
f = Helm_FE(u);
cout << product(u,f) << endl;
```

calculates the volume of the domain.

Test 2: Volume Calculation:

Now let us implement the above operators by expression templates.
Then, the code

```
u = X;
f = Poisson_FE(u);
cout << product(u,f) << endl;
```

calculates the volume of the domain.
Here, let `X` be the x-coordinate.

Dirichlet Boundary Conditions:
The problem

Problem 3. *Find $u \in H^1(\Omega)$ such that*

$$\begin{aligned} -\triangle u &= f \quad \textit{on } \Omega, \\ u|_\Gamma &= g \end{aligned}$$

can approximatively be solved by finite elements and the Gauss-Seidel iteration as follows:

```
u = Helm_FE(f);
f = u;
u = g | boundary;
for(i=0;i<i_max;++i)
 u = u - (Laplace_FE(u)-f)/Diag_Laplace_FE()
      |  interior_points;
```

Using a Direct Solver:
Let us assume that there is a good direct solver `ui=Inverse(S,fi)` which calculates

$$ui = S^{-1} fi.$$

Then, apply the code

```
u = Helm_FE(f);
f = u;
u = 0.0 | interior;
u = g   | boundary;
f = f - Laplace_FE(u) | boundary;
u = 0.0 | boundary;
S = Sparse_matrix(Laplace_FE, interior);
fi = vector(f,interior);
ui = vector(u,interior);
ui = Inverse(S,fi);
u  = g   | boundary;
```

2.7 Boundary Conditions

Let us consider the equation

$$\begin{aligned}
-\operatorname{div} \mu \operatorname{grad} u &= f \quad \text{on } \Omega,\\
u &= g \quad \text{on } \Gamma_D,\\
\frac{\partial u}{\partial \vec{n}} &= 0 \quad \text{on } \Gamma_N,\\
\frac{\partial u}{\partial \vec{n}} + \beta(u - u_{ref}) &= 0 \quad \text{on } \Gamma_{third},
\end{aligned}$$

here Ω is a domain and

$$\partial\Omega = \Gamma_D \cup \Gamma_N \cup \Gamma_{third}$$

is a disjunct subdivision of the boundary, where $\Gamma_D \neq \emptyset$.
Furthermore, let $\mu : \Omega \to \mathbb{R}$ be a piecewise constant parameter and $0 < \beta \in \mathbb{R}$.
Define the finite element space

$$\bar{V}_h := \{v_h \in V_h \mid v_h\big|_{\Gamma_D} = 0\}.$$

Then, we obtain

$$\int_\Omega \nabla u \mu \nabla v_h \, d(x,y) \;+\; \beta\mu \int_{\Gamma_{third}} u v_h \, d\sigma \;=\; \beta\mu \int_{\Gamma_{third}} u_{ref} v_h \, d\sigma \;+\; \int_\Omega f v_h d(x,y)$$

for every $v_h \in \bar{V}_h$.
<u>FE Discretization</u>
Find $u_h \in V_h$ such that

$$\begin{aligned}\int_\Omega \nabla u_h \mu \nabla v_h \, d(x,y) \ + \ \beta\mu \int_{\Gamma_{third}} u_h v_h \, d\sigma \ &= \ \beta\mu \int_{\Gamma_{third}} u_{ref} v_h \, d\sigma \ + \ \int_\Omega f v_h d(x,y) \\ &\quad \forall v_h \in \bar{V}_h, \\ u_h(z) \ &= \ g(z) \quad \forall z \in \Omega_h \cap \Gamma_D.\end{aligned}$$

Observe that the bilinear form

$$a(u,v) := \int_\Omega \nabla u \mu \nabla v \, d(x,y) \ + \ \beta\mu \int_{\Gamma_{third}} uv \, d\sigma$$

is symmetric positive definite on the space

$$\{v \in H^1(\Omega) \mid v\big|_{\Gamma_D} = 0\}.$$

<u>Operator Formulation:</u>
Let us implement the operators in section 2.6 by expression templates.
Let `A_FE` be the operator corresponding to the bilinear form $a(u,v)$.
The previous problem can be solved by the Gauss-Seidel iteration as follows:

```
u = Helm_FE(f);
f = u;
u = g | boundary_D;
for(i=0;i<i_max;++i)
    u = u - (A_FE(u)-f)/Diag_A_FE() | grid_space;
```

Here `grid_space` represents the interior points and the boundary points which are no Dirichlet boundary points.

2.8 Pure Neumann Boundary Conditions

Let us consider the equation

$$\begin{aligned} -\triangle u \ &= \ f \quad \text{on } \Omega, && (20) \\ \frac{\partial u}{\partial \vec{n}} \ &= \ 0 \quad \text{on } \Gamma. && (21)\end{aligned}$$

A short calculation shows

$$\int_\Omega f \; d(x,y) = 0. \tag{22}$$

Thus, we assume (22).
Furthermore, observe that the constant function $u = \underline{1}$ satisfies

$$\begin{aligned} -\triangle u &= 0 \quad \text{on } \Omega, \\ \frac{\partial u}{\partial \vec{n}} &= 0 \quad \text{on } \Gamma. \end{aligned}$$

Thus, we need an additional assumption to guarantee a unique solution of (20). There are different possibilities like

$$\int_\Omega u \; d(x,y) = 0$$

or

$$\int_\Gamma u \; d(x,y) = 0$$

and other conditions, which factor out the constants.
A natural way to obtain a well-defined problem is:

Problem 4. *Find $u \in H^1(\Omega)$ such that*

$$\begin{aligned} -\triangle u &= f \quad \textit{on } \Omega, \\ \frac{\partial u}{\partial \vec{n}} &= 0 \quad \textit{on } \Gamma \textit{ and} \int_\Omega u \; d(x,y) = 0, \end{aligned}$$

where we assume

$$\int_\Omega f \; d(x,y) = 0.$$

Operator Formulation:
Let us implement the operators in section 2.6 by expression templates. The previous problem can be solved by Gauss-Seidel iteration as follows:

```
  Eins = 1.0;       // set up for normalization
  IntE = Helm_FE(Eins);
  Eins = Eins / product(Eins,IntE);
```

```
f = f - Eins * product(f,IntE);
u = Helm_FE(f);
f = u;

for(int i=0; i<N;++i) {
  u = u - (A_FE(u) -f) / Diag_A_FE();
  u = u - Eins * product(u,IntE);
}
```

2.9 Streamline Diffusion in 2D

Consider the convection diffusion equation in 1D:

$$\begin{aligned} -\triangle u - \vec{b} \text{ grad } u &= f \\ u\big|_{\partial\Omega} &= 0. \end{aligned}$$

where $\vec{b} : \Omega \to \mathbb{R}^2$ is a vector field.
Multiply this equation by $v = v_h - \rho h \vec{b} \circ \nabla v_h \ \ \|\vec{b}\|_2^{-1}$, where $v_h \in \overset{\circ}{V}_h$ and integrate

$$\begin{aligned} &\int_\Omega \Big(\nabla u \circ \nabla v_h + h\rho \vec{b} \circ \nabla u \ \vec{b} \circ \nabla v_h \ \|\vec{b}\|_2^{-1} - \vec{b} \circ \nabla u \ v_h \Big) \ d(x,y) \\ &\qquad -\rho h \int_\Omega \triangle u \ \vec{b} \circ \nabla v_h \ d(x,y) \ \|\vec{b}\|_2^{-1} = \int_\Omega f(v_h - \rho h \vec{b} \circ \nabla v_h \ \ \|\vec{b}\|_2^{-1}) d(x,y). \end{aligned}$$

In the streamline diffusion discretization, we neglect the term of third order and replace u by u_h: <u>Discretization:</u> Find $u_h \in \overset{\circ}{V}_h$ such that

$$\begin{aligned} &\int_\Omega \Big(\nabla u_h \circ \nabla v_h + h\rho \vec{b} \circ \nabla u_h \ \vec{b} \circ \nabla v_h \ \|\vec{b}\|_2^{-1} - \vec{b} \circ \nabla u_h \ v_h \Big) \ d(x,y) \\ &= \int_\Omega f(v_h - \rho h \vec{b} \circ \nabla v_h \ \ \|\vec{b}\|_2^{-1}) d(x,y) \end{aligned}$$

for every $v_h \in \overset{\circ}{V}_h$.

3 Grids

3.1 Types of Grids

There exist

- Cartesian grids
- block structured grids
- unstructured grids
- ...

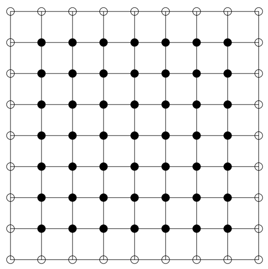
to discretize a domain Ω.

Cartesian Grid:

Example of an Cartesian grid:

$$\Omega_{h,k} = \{(ih, jk) + (x_0, y_0) \mid i = 0, ..., m, \quad j = 0, ..., n\},$$

where $h, k > 0$.



Data structure: Array!

Block Structured Grid:

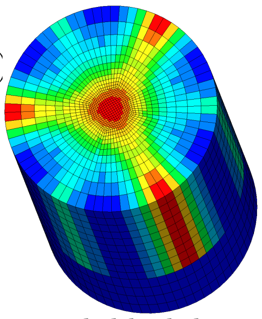
Let

$$\Omega_h^0 = \{(ih, jh) \mid i, j = 0, ..., n\},$$

where $h = \frac{1}{n}$. Furthermore, let $\mathcal{T} = \{T_1, \ldots, T_M\}$ be a subdivision by quadrangles (2D) and

$$\Psi_k : [0,1]^2 \to T_k$$

smooth bijections such that $\Omega = \bigcup_{k=1}^{M} \Psi_k([0,1]^2)$. Then,



$$\Omega_h = \bigcup_{k=1}^{M} \Psi_k(\Omega_h^0)$$

is a block structured grid.

(Generalizations in 3D and for different mesh sizes are possible.)

Data structure: Unstructured grid of quadrangles, each block by array.

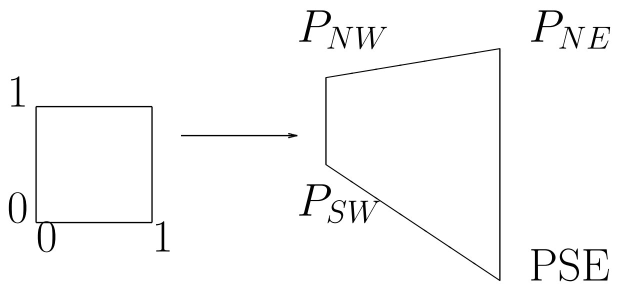
Simple Interpolation:

A simple construction of the mapping

$$\Psi_k : [0,1]^2 \to T_k$$

is

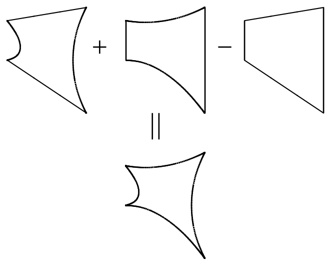
$$\begin{aligned}\Psi_k(\eta,\xi) &= P_{SW} + (P_{SE} - P_{SW})\eta + (P_{NW} - P_{SW})\xi + \\ &\quad (P_{NE} - P_{SE} - P_{NW} + P_{SW})\xi\eta.\end{aligned}$$



Transfinite Interpolation:

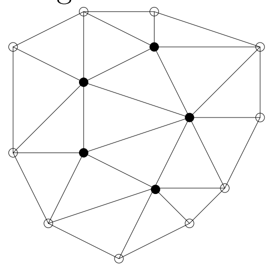
Let $\beta_N, \beta_S, \beta_W, \beta_E : [0,1] \to \mathbb{R}^2$ be parameterizations of the north, south, west and east boundary. Then, the transfinite interpolation is:

$$\begin{aligned}\Psi_k(\eta,\xi) &= \beta_S(\eta) + (\beta_N(\eta) - \beta_S(\eta))\xi \\ &\quad + \beta_W(\xi) + (\beta_E(\xi) - \beta_W(\xi))\eta \\ &\quad - P_{SW} - (P_{SE} - P_{SW})\eta - (P_{NW} - P_{SW})\xi \\ &\quad - (P_{NE} - P_{SE} - P_{NW} + P_{SW})\xi\eta.\end{aligned}$$



Unstructured Grid:

An example of an unstructured grid is:



Data structure for an unstructured grid:

- list or array of corners (information of coordinates)
- list or array of triangles, quadrangles, ... with pointers to corners and number of corners.

By this information one can construct:

- list or array of edges and faces (in 3D).

Visualization of an Unstructured Grid:

Examples of powerful 3D visualization programs are:

- AVS (commercial)
- OpenDx (public domain)
- ParaView (uses vtk Toolkit, public domain)

AVS supports structured grids and unstructured grids. An unstructured grid may consist of geometric elements

- point (0D)
- line (1D)
- triangle, quadrangle (2D)
- tetrahedron, hexahedron, prism, pyramid (3D).

An unstructured grid can be described by an UCD file format. In this file format, every geometric element must be numbered in a certain orientation (see handbook of AVS or OpenDx.)

Example of file in UCD format for AVS:

```
# UCD file format for AVS
22 44 1 0 0
0 0.292053 0.292053 0.292053
1 0.292053 0.292053 0.892053
...
21 0.292053 0.292053 1.00005
1  1  tet  12  2  7  0
...
43  1  tet  19  21  17  1
44  1  tet  21  19  18  1
1 1
variable
0  0.433753
1  -0.296865
...
21  -0.369419
```

Example of file in dx format for OpenDx:

```
# dx file format for OpenDx  unstructured grid
object 1 class array type float rank 1 shape 3 items 22 data follows
 0.292053 0.292053 0.292053
 ...
 0.292053 0.292053 1.00005
object 2 class array type int rank 1 shape 4 items 44 data follows
12  2  7  0
20  0  17  1
...
21  19  18  1
attribute "element type" string "tetrahedra"
attribute "ref" string "positions"
object 3 class array type float rank 0 items 22 data follows
0.433753
...
-0.369419
attribute "dep" string "positions"
object "irregular positions irregular connections" class field
component "positions" value 1
component "connections" value 2
component "data" value 3
end
```

Example of file in dx format for OpenDx:

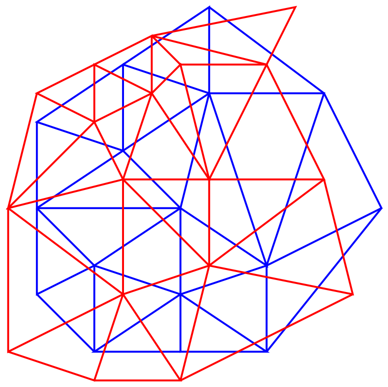
```
# dx file format for OpenDx structured grid
object 1 class gridpositions counts 10 10 10
origin 0.005 0.000 0.005
delta 0.010 0 0
delta 0 0.010 0
delta 0 0 0.010
object 2 class gridconnections counts 100 101 99
attribute "element type" string "cubes"
attribute "ref" string "positions"
object 3 class array type float rank 0 items 1000 data follows
  -0.200
...
  -0.369419
attribute "dep" string "positions"
```

```
object 4 class field
component "positions" value 1
component "connections" value 2
component "data" value 3
end
```

3.2 Interpolation between Grids

Assume that a finite element function u is given on a triangulation $\mathcal{T}_{h_1}$.

- How to find the values of u on a grid Ω_{h_2}?
- How to find the triangles T_p for every $p \in \Omega_{h_2}$?



Test for one Triangle:

Let P_1, P_2, P_3 be the corners of one triangle.
Is a certain point P contained in the triangle $P_1P_2P_3$?
Let (ξ, η) be such that

$$P = P_1 + (P_2 - P_1)\xi + (P_3 - P_1)\eta.$$

Then P is contained in the triangle $P_1P_2P_3$ if and only if

$$\xi + \eta \leq 1 \quad \text{and} \quad \xi, \eta \geq 0.$$

Such a test for all points $P \in \Omega_{h_2}$ and triangles $\mathcal{T}_{h_1}$ is very time consuming.

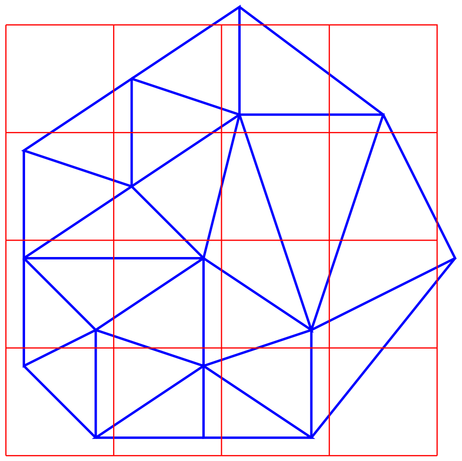
From Structured to Unstructured Grid:

Let the structured grid be

$$\Omega_{h_1} = \{(x_0 + h_1 i, y_0 + h_1 j) \mid i, j = 0, ..., N\}$$

Then, by a simple indices calculation one obtains the index i', j' such that

$$p \in (x_0, y_0) + h_1[i', i' + 1] \times h_1[j', j' + 1].$$



<u>From Unstructured to Structured Grid:</u>

Let the structured grid be

$$\Omega_{h_2} = \{(x_0 + h_2 i, y_0 + h_2 j) \mid i, j = 0, ..., N.\}$$

Now perform the following steps:

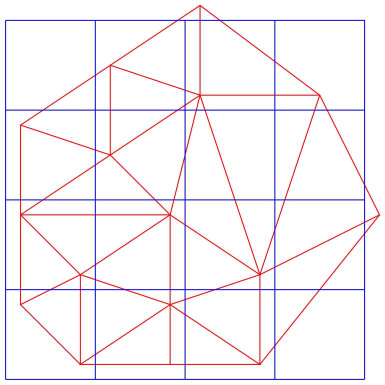
1. For every triangle $T = T((x_1, y_1), (x_2, y_2), (x_3, y_3)) \in \mathcal{T}_{h_1}$ consider the quadrangle

   $Q = [(\min(x_1, x_2, x_3), \min(y_1, y_2, y_3)), (\max(x_1, x_2, x_3), \max(y_1, y_2, y_3))].$

2. For every $p \in Q \cap \Omega_{h_2}$, set $T(p) = T$, if $p \in T$. This means store the index of T at p, if $p \in T$.

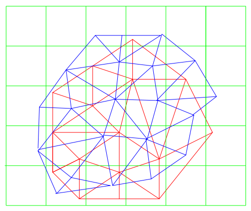
3. Test if $T(p)$ is set for every $p \in \Omega_{h_2}$. If not, then calculate the next point $q \in \Omega_{h_2}$ from p such that $T(q)$ is set and $T(p) = T(q)$.

4. Interpolate data from Ω_{h_1} to Ω_{h_2} by using $T(p)$.

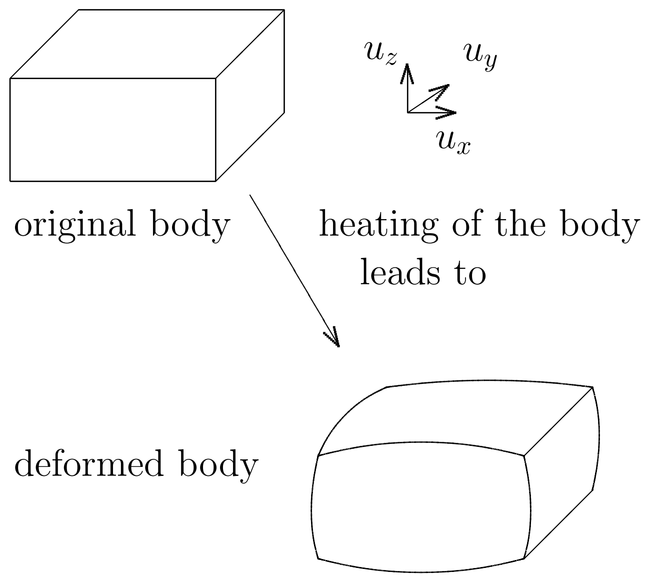


From Unstructured to Unstructured Grid:

- Construct an auxiliary structured grid such that the domain of this grid contains the domain of the two unstructured grids. The meshsize of the auxiliary structured grid should roughly be the meshsize of the two unstructured grids.
- Then, for every triangle $T \in \mathcal{T}_{h_1}$, put T in the cell c of the structured grid, if c intersects with T. (see "From Unstructured to Structured Grid"). This means let $T \in T_c$, if $T \cap c \neq \emptyset$.
- For every $p \in \Omega_{h_2}$, find the structured cell c such that $p \in c$. Then, find triangle $T \in T_c$ such that $p \in T$.



4 Structural Mechanics: Linear Elasticity



- Let $\Omega \subset \mathbb{R}^3$ be the domain of the body.
- Let $T_0 \in \mathbb{R}$ be the original temperature of the body.
- Let $T : \Omega \to \mathbb{R}$ be the temperature of the body after heating.
- Let $\vec{u} = \begin{pmatrix} u_x \\ u_y \\ u_z \end{pmatrix} : \Omega \to \mathbb{R}^3$ be the deformation vector of the body after heating.

Problem: Let Ω, T_0, T be given. Then, calculate $\vec{u}$.

Definition 9. *Let $\vec{u} : \Omega \to \mathbb{R}^3$. The symmetric derivative is defined by:*

$$D\vec{u} := \begin{pmatrix} \frac{\partial u_x}{\partial x} \\ \frac{\partial u_y}{\partial y} \\ \frac{\partial u_z}{\partial z} \\ \frac{1}{2}\left(\frac{\partial u_x}{\partial y} + \frac{\partial u_y}{\partial x}\right) \\ \frac{1}{2}\left(\frac{\partial u_y}{\partial z} + \frac{\partial u_z}{\partial y}\right) \\ \frac{1}{2}\left(\frac{\partial u_x}{\partial z} + \frac{\partial u_z}{\partial x}\right). \end{pmatrix} : \Omega \to \mathbb{R}^6$$

Another notation is

$$\begin{aligned}
\epsilon_{ij} &:= \frac{1}{2}\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right)_{(i,j)\in\Phi} \\
D\vec{u} &:= \begin{pmatrix} \epsilon_{11} \\ \epsilon_{22} \\ \epsilon_{33} \\ \epsilon_{12} \\ \epsilon_{13} \\ \epsilon_{23} \end{pmatrix},
\end{aligned}$$

where $\Phi = \{(1,1),(2,2),(3,3),(1,2),(2,3),(3,1)\}$.

Symmetric Divergence:

$$\operatorname{div}\Big((\sigma_{ij})_{(i,j)\in\Phi}\Big) := \frac{1}{2}\left(\Sigma_j \frac{\partial \sigma_{ij}}{\partial x_j} e_i + \Sigma_i \frac{\partial \sigma_{ij}}{\partial x_i} e_j\right).$$

$\operatorname{div}\Big((\sigma_{ij})_{(i,j)\in\Phi}\Big)$ is the adjoint operator of $D\vec{u}$ in the following sense:

$$\int_\Omega \operatorname{div}\Big((\sigma_{ij})_{(i,j)\in\Phi}\Big) v \; d(x,y,z) = -\int_\Omega (\sigma_{ij})_{(i,j)\in\Phi} Dv \; d(x,y,z)$$

Observe, that for a symmetric matrix $(\sigma_{ij})_{(i,j)\in\Phi}$:

$$\operatorname{div}\Big((\sigma_{ij})_{(i,j)\in\Phi}\Big) = \Sigma_j \frac{\partial \sigma_{ij}}{\partial x_j} e_i.$$

Definition 10. *Let $E > 0$ and $0 < \nu < \frac{1}{2}$ be the physical constants E-Modul and Poisson ratio.*
Then, define the matrix

$$\mathcal{C}^{-1} = \frac{1}{E}\begin{pmatrix} 1 & -\nu & -\nu & & & \\ -\nu & 1 & -\nu & & 0 & \\ -\nu & -\nu & 1 & & & \\ & & & 1+\nu & & \\ & 0 & & & 1+\nu & \\ & & & & & 1+\nu \end{pmatrix},$$

The inverse of this matrix is

$$\mathcal{C} = \begin{pmatrix} \lambda+2\mu & \lambda & \lambda & & & \\ \lambda & \lambda+2\mu & \lambda & & 0 & \\ \lambda & \lambda & \lambda+2\mu & & & \\ & & & 2\mu & & \\ & 0 & & & 2\mu & \\ & & & & & 2\mu \end{pmatrix},$$

where

$$\begin{aligned} \lambda &= \frac{\nu E}{(1+\nu)(1-2\nu)} \\ \mu &= \frac{E}{2(1+\nu)}. \end{aligned}$$

The deformation vector of the body satisfies the equations:

$$\begin{aligned} D\vec{u} &= \begin{pmatrix} \alpha \\ \alpha \\ \alpha \\ 0 \\ 0 \\ 0 \end{pmatrix} (T - T_0) + \mathcal{C}^{-1}\sigma, \\ \text{div } (\sigma) &= 0, \end{aligned}$$

where

- α is a physical constant and
- σ is called stress vector.

Define the bilinear form

$$\begin{aligned} a : (H^1(\Omega))^3 \times (H^1(\Omega))^3 &\rightarrow \mathbb{R} \\ (u, v) &\mapsto \int_\Omega (Du)^T \mathcal{C} Dv \; d(x, y, z). \end{aligned}$$

Let $v \in (H_0^1(\Omega))^3$. Then, we obtain

$$a(u,v) \;=\; \int_\Omega \operatorname{div} \mathcal{C} \begin{pmatrix} \alpha \\ \alpha \\ \alpha \\ 0 \\ 0 \\ 0 \end{pmatrix} (T - T_0)\; v \; d(x,y,z).$$

Matrix of Linear Elasticity:

A short calculation shows that

$$\begin{aligned} a(\vec{u},\vec{v}) \;&= \\ = \int_\Omega F \;& \Big[(1-\nu)\, \tfrac{\partial u_x}{\partial x}\, \tfrac{\partial v_x}{\partial x} + \nu\, \tfrac{\partial u_y}{\partial y}\, \tfrac{\partial v_x}{\partial x} + \nu\, \tfrac{\partial u_z}{\partial z}\, \tfrac{\partial v_x}{\partial x} + \tfrac{1}{2}\,(1-2\nu)\, \tfrac{\partial u_y}{\partial x}\, \tfrac{\partial v_x}{\partial y} + \\ & \tfrac{1}{2}\,(1-2\nu)\, \tfrac{\partial u_x}{\partial y}\, \tfrac{\partial v_x}{\partial y} + \tfrac{1}{2}\,(1-2\nu)\, \tfrac{\partial u_z}{\partial x}\, \tfrac{\partial v_x}{\partial z} + \tfrac{1}{2}\,(1-2\nu)\, \tfrac{\partial u_x}{\partial z}\, \tfrac{\partial v_x}{\partial z} + \\ & \tfrac{1}{2}\,(1-2\nu)\, \tfrac{\partial u_x}{\partial y}\, \tfrac{\partial v_y}{\partial x} + \tfrac{1}{2}\,(1-2\nu)\, \tfrac{\partial u_y}{\partial x}\, \tfrac{\partial v_y}{\partial x} + \nu\, \tfrac{\partial u_x}{\partial x}\, \tfrac{\partial v_y}{\partial y} + \\ & (1-\nu)\, \tfrac{\partial u_y}{\partial y}\, \tfrac{\partial v_y}{\partial y} + \nu\, \tfrac{\partial u_z}{\partial z}\, \tfrac{\partial v_y}{\partial y} + \tfrac{1}{2}\,(1-2\nu)\, \tfrac{\partial u_z}{\partial y}\, \tfrac{\partial v_y}{\partial z} + \tfrac{1}{2}\,(1-2\nu)\, \tfrac{\partial u_y}{\partial z}\, \tfrac{\partial v_y}{\partial z} + \\ & \tfrac{1}{2}\,(1-2\nu)\, \tfrac{\partial u_x}{\partial z}\, \tfrac{\partial v_z}{\partial x} + \tfrac{1}{2}\,(1-2\nu)\, \tfrac{\partial u_z}{\partial x}\, \tfrac{\partial v_z}{\partial x} + \tfrac{1}{2}\,(1-2\nu)\, \tfrac{\partial u_y}{\partial z}\, \tfrac{\partial v_z}{\partial y} + \\ & \tfrac{1}{2}\,(1-2\nu)\, \tfrac{\partial u_z}{\partial y}\, \tfrac{\partial v_z}{\partial y} + \nu\, \tfrac{\partial u_x}{\partial x}\, \tfrac{\partial v_z}{\partial z} + \nu\, \tfrac{\partial u_y}{\partial y}\, \tfrac{\partial v_z}{\partial z} + (1-\nu)\, \tfrac{\partial u_z}{\partial z}\, \tfrac{\partial v_z}{\partial z} \Big]\, d(x,y,z). \end{aligned}$$

For the implementation of this bilinear form, it is helpful to sort the terms of this bilinear form by a 3×3 matrix:

$$\begin{bmatrix} M_{11} & M_{12} & M_{13} \\ M_{21} & M_{22} & M_{23} \\ M_{31} & M_{32} & M_{33} \end{bmatrix}.$$

RHS of Linear Elasticity:

The right hand side can be written as

$$-\int_\Omega F \left[(1+\nu)\, \alpha_T\, \Delta T\, \frac{\partial v_x}{\partial x} + (1+\nu)\, \alpha_T\, \Delta T\, \frac{\partial v_y}{\partial y} + (1+\nu)\, \alpha_T\, \Delta T\, \frac{\partial v_z}{\partial z} \right] d(x,y,z). \tag{23}$$

For the implementation of the right hand side, it is helpful to sort the above terms:

$$\begin{bmatrix} F_1 \\ F_2 \\ F_3 \end{bmatrix}. \tag{24}$$

<u>Boundary Conditions for Linear Elasticity:</u>
Let $\Gamma_D \subset \partial\Omega$ be the fixed boundary of the deformation process. Let the rest of the boundary be free.
Then, define

$$V = \{v \in H^1(\Omega) \mid v|_{\Gamma_D} = 0\}.$$

Problem 5 (Weak formulation with boundary condition)**.** *Find $u \in V$ such that*

$$a(u,v) \;=\; \int_\Omega div\mathcal{C} \begin{pmatrix} \alpha \\ \alpha \\ \alpha \\ 0 \\ 0 \\ 0 \end{pmatrix} (T - T_0)\; v\; d(x,y,z)$$

for every $v \in V$.

<u>FE Discretization of Linear Elasticity:</u>
Let V_h be the space of trilinear finite elements. Then, define

$$\vec{V_h} = \{\vec{v} \in (V_h)^3 \mid \vec{v}|_{\Gamma_D} = 0\}.$$

Problem 6 (Weak formulation with boundary condition)**.** *Find $u_h \in \vec{V_h}$ such that*

$$a(u_h,v_h) \;=\; \int_\Omega div\mathcal{C} \begin{pmatrix} \alpha \\ \alpha \\ \alpha \\ 0 \\ 0 \\ 0 \end{pmatrix} (T - T_0)\; v_h\; d(x,y,z)$$

for every $v_h \in \vec{V_h}$.

Rigid Body Modes:

Consider the vector space functions

$$\mathcal{M} := \operatorname{span}\left\{ \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}, \begin{pmatrix} y \\ -x \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ z \\ -y \end{pmatrix}, \begin{pmatrix} z \\ 0 \\ -x \end{pmatrix} \right\}$$

$\mathcal{M}$ is the kernel of the bilinear form a. This means

$$a(\vec{m}, \vec{v}) = 0 \qquad \forall \vec{m} \in \mathcal{M},\ \forall \vec{v} \in V.$$

Therefore, in case of Neumann boundary conditions, we have to construct V such that $V \cap \mathcal{M} = \{\vec{0}\}$.
In case of pure boundary conditions, define V as follows:

$$V = \{v \in H^1(\Omega) \mid \langle v, \vec{m} \rangle = 0 \ \ \forall \vec{m} \in \mathcal{M}\}.$$

Superconvergence of the Gradient:

The stress σ has to be calculated by $D\vec{u}$.
The finite element theory for linear and trilinear finite elements shows

$$\|D\vec{u} - D\vec{u}_h\|_{L^2} = O(h).$$

This is a slow convergence. But one can prove the following superconvergence of the gradient in case of structured grids:
Let Σ_h be the cell points of the structured grid. Then, for a sufficient smooth solution $\vec{u}$ and a not complicated boundary Γ, we obtain:

$$\max_{p \in \Sigma_h} \|(D\vec{u} - D\vec{u}_h)(p)\| = O(h^2).$$

Therefore, in case of linear elasticity, apply

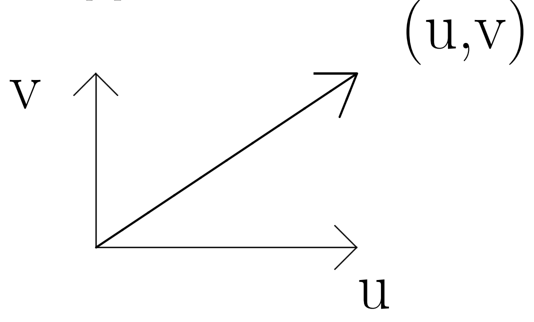
- trilinear elements on a block-structured grid or
- quadratic finite elements.

5 Fluid Dynamics

5.1 Navier Stokes Equations

Let us describe a two dimensional flow by:

- u x-component of the velocity vector of the flow,
- v y-component of the velocity vector of the flow,
- p pressure of the flow.



Navier-Stokes-Equations:

The Navier-Stokes-equations are:

$$
\begin{aligned}
\frac{\partial u}{\partial t} + \frac{\partial p}{\partial x} + \frac{\partial (u^2)}{\partial x} + \frac{\partial (uv)}{\partial y} &= \frac{1}{\mathrm{Re}}\Delta u \\
\frac{\partial v}{\partial t} + \frac{\partial p}{\partial y} + \frac{\partial (uv)}{\partial x} + \frac{\partial (v^2)}{\partial y} &= \frac{1}{\mathrm{Re}}\Delta v \\
\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} &= 0
\end{aligned}
$$

There exist different kind of boundary conditions:
input, output, slip, and no-slip boundary conditions.

Boundary Conditions::

- input boundary condition: Dirichlet boundary condition.

  Usually it is

$$(u, v) \circ \vec{t} = 0.$$

- output boundary condition: Neumann boundary condition or better boundary conditions.
- no-slip boundary condition: Dirichlet boundary condition:
- slip boundary condition:

$$(u,v)\circ\vec{n}=0, \quad \frac{\partial (u,v)\circ\vec{t}}{\partial\vec{n}}=0.$$

Here $\vec{t}$ and $\vec{n}$ are the tangential and normal boundary vectors.

<u>Stokes-Equations::</u>
The Stokes equations are:

$$\begin{aligned} -\Delta u+\frac{\partial p}{\partial x} &= f_x, \\ -\Delta v+\frac{\partial p}{\partial y} &= f_y, \\ \frac{\partial u}{\partial x}+\frac{\partial v}{\partial y} &= 0. \end{aligned}$$

There exist several different kind of implicit, semi-implicit, and explicit discretizations of the Navier-Stokes equations.
Important is the stability of these discretizations in space and time.
Stability in time can be analyzed by Fourier analysis.

<u>Checkerboard Function:</u>
Let us discretize Stokes equations by finite difference discretization as follows:

- all unknowns at the grid points:

$$\Omega_h=\Big\{(i,j)h \mid i,j=0,...,m\Big\},$$

- five point stencil for $\triangle u$, and
- central difference for $\frac{\partial p}{\partial x}$ and $\frac{\partial p}{\partial y}$.

Then, the pressure function

```
a b a b a
b a b a b ...
a b a b a
b a b a b
   ...
```

is contained in the kernel of the discrete Stokes operator.
Unstable discretization!

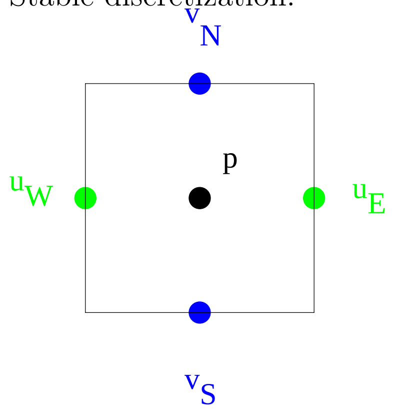
Staggered Grid:
Let us define the following three kind of grids:

$$\begin{aligned}
\Omega_{h,u} &= \Big\{(i, j-0.5)h \mid i = 0, ..., m, \quad j = 1, ..., m\Big\},\\
\Omega_{h,v} &= \Big\{(i-0.5, j)h \mid i = 1, ..., m, \quad j = 0, ..., m\Big\},\\
\Omega_{h,p} &= \Big\{(i-0.5, j-0.5)h \mid i, j = 1, ..., m\Big\}.
\end{aligned}$$

Apply the discretization:

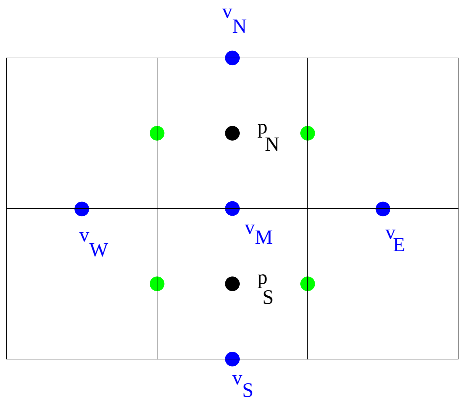
- five point stencil for $\triangle u$ at $\Omega_{h,u}$ and for $\triangle v$ at $\Omega_{h,v}$,
- central difference for $\frac{\partial p}{\partial x}$ and $\frac{\partial p}{\partial y}$ at $\Omega_{h,p}$,
- central difference for $\frac{\partial u}{\partial x} + \frac{\partial u}{\partial y}$ at $\Omega_{h,p}$.

Here apply the central difference discretization with respect to the meshsize $\frac{h}{2}$.
Stable discretization!



The finite difference discretization on a staggered grid leads to

$$\left(\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}\right)(x,y) \quad \approx \quad \frac{u_E-u_W+v_N-v_S}{h}=0.$$



The finite difference discretization on a staggered grid leads to

$$-\triangle v(x,y)+\frac{\partial p}{\partial y}(x,y) \quad \approx \quad \frac{-v_N-v_S-v_E-v_W+4v_M}{h^2}+\frac{p_N-p_S}{h}=f_y(x,y).$$

- The staggered grid discretization is similar to the finite volume discretization.
- There exist several stable finite element discretizations.

5.2 The Lattice Boltzmann Method

5.2.1 Basic Physics

Definition 11 (Particle Distribution)**.** *The fundamental variable in kinematic theory is the particle distribution $f(\mathbf{x},\xi,t)$ with respect to velocity ξ at spatial coordinate $\mathbf{x}$ and time t. This means that the density of particles at point $\mathbf{x}$ and time t, which move with velocity ξ, is $f(\mathbf{x},\xi,t)$. Here, $\mathbf{x}\in\Omega\subset\mathbb{R}^3$ and $\xi\in\mathbb{R}^3$.*

Obviously, the density of the fluid is

$$\rho(\mathbf{x},t)=\int f(\mathbf{x},\xi,t)\;d\xi \tag{25}$$

and the velocity can implicitly be calculated by

$$\mathbf{u}(\mathbf{x}, t)\rho(\mathbf{x}, t) = \int \xi f(\mathbf{x}, \xi, t)\ d\xi$$

Kinematic theory tells that a gas tends to reach state of equilibrium which statisfies the Boltzmann distribution:

$$f^{\mathrm{eq}}(\mathbf{x}, |\mathbf{v}|, t) = \rho \left(\frac{1}{2\pi RT}\right)^{3/2} e^{|\mathbf{v}|^2/(2RT)}, \tag{26}$$

where T is the temperature, $\mathbf{v}$ is the velocity, and R is the specific gass constant. The Boltzmann equation is

$$\frac{df}{dt} = \Omega(f) := -\frac{1}{\tau}(f - f^{\mathrm{eq}}(\mathbf{x}, |\mathbf{v}|, t)), \tag{27}$$

where $\Omega(f)$ is called collision operator and τ relaxation time.
Observe, that $\frac{df}{dt}$ is the totel derivative:

$$\frac{df}{dt} = \frac{\partial f}{\partial t} + \xi \frac{\partial f}{\partial \mathbf{x}}.$$

5.2.2 Lattice Boltzmann Discretization

Here, we explain the Lattice Boltzmann discretization using the D2Q9 scheme. It applies a discretization grid Ω_h of meshsize h with cell center points $\mathbf{x}_k$ (see Figure 5.2.1). Motivated by the particle distribution $f(\mathbf{x}, \xi, t)$, Lattice Boltzmann discretization uses 9 functions

$$f_i(\mathbf{x}_k, t),$$

where $i \in \{0, 1, 2, 3, ..., 8\}$. The discrete distribution function $f_i(\mathbf{x}_k, t)$ is related to the discrete velocities c_i (see Table 5.2.2) The discretized density is

$$\rho_h(\mathbf{x}_k, t) = \sum_{i=0}^{8} f_i(\mathbf{x}_k, t) \tag{28}$$

and the discretized velocity is implicitly defined by

$$\mathbf{u}_h(\mathbf{x}_k, t)\rho_h(\mathbf{x}_k, t) = \sum_{i=0}^{8} c_i f_i(\mathbf{x}_k, t) \tag{29}$$

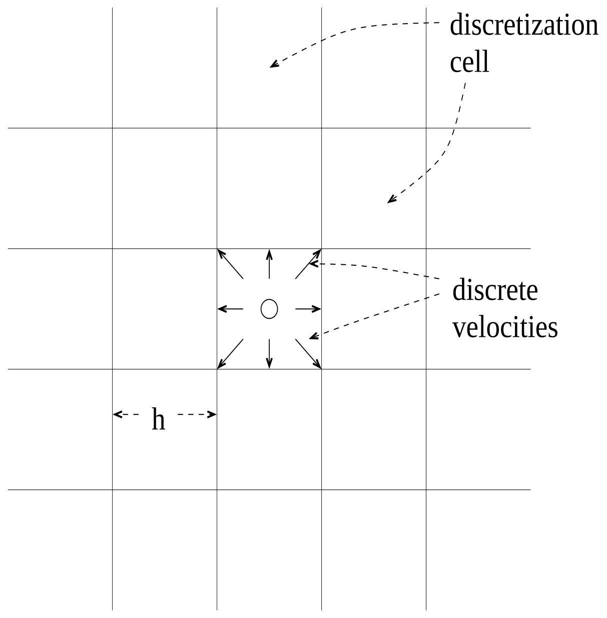


Figure 5: Lattice Boltzmann Grid.

Observe the property:

$$\mathbf{x}_k + hc_i \in \Omega_h \quad \text{for all interior points } \mathbf{x}_k \in \Omega_h.$$

The Boltzmann distribution (26) can be discretized as follows

$$f_i^{\text{eq}}(\mathbf{x}_k, t) = w_i \rho_h(\mathbf{x}_k, t) \left(1 + \frac{\mathbf{u}_h c_i}{c_s^2} + \frac{(\mathbf{u}_h c_i)^2}{4c_s^4} - \frac{\mathbf{u}_u \mathbf{u}_h}{2c_s^2} \right),$$

where c_s is speed of sound and the discetization constants w_i are given in Table 5.2.2.
Furthermore, the Boltzmann equation (27) is discretized by

$$\frac{f_i(\mathbf{x}_k + hc_i, t + \triangle t) - f_i(\mathbf{x}_k, t)}{\triangle t} = -\frac{1}{\tau}(f_i - f_i^{\text{eq}}(\mathbf{x}_k, t)),$$

where $\triangle t$ is the time step related to the velocities c_i and meshsize h.

i	0	1	2	3	4	5	6	7	8
c_i	(0,0)	(1,0)	(0,1)	(-1,0)	(0,-1)	(1,1)	(-1,1)	(-1,-1)	(1,-1)
w_i	$\frac{4}{9}$	$\frac{1}{9}$	$\frac{1}{9}$	$\frac{1}{9}$	$\frac{1}{9}$	$\frac{1}{36}$	$\frac{1}{36}$	$\frac{1}{36}$	$\frac{1}{36}$

Algorithm 1 (Lattice Boltzmann Algorithm)**.**

1. *Calculate density and velocity by (28) and (29).*
2. *Calculate collision term by*
$$\Omega_i(f) := -\frac{\triangle t}{\tau}(f_i - f_i^{eq}(\mathbf{x}_k, t))$$
3. *Calculate streaming by*
$$f_i(\mathbf{x}_k + hc_i, t + \triangle t) = f_i(\mathbf{x}_k, t) + \Omega_i(f).$$

Remark: Observe that f_i might take negative values. This clearly shows that f_i is not a discretization of f. Instead f_i is an auxilliary variable which has simular properties like f (e.g. see (25) and (28)). Nevertheless the important unknows velocity $\mathbf{u}_u$ and density ρ_h converge to the physical quantities $\mathbf{u}$ and ρ under suitable conditions.

Properties of Lattice Boltzmann discretization: The basic Lattice Boltzmann discretization is easier to implement and to parallelize than finite difference discretization. However, a large number of time step is need, since $\triangle t$ is bounded by the meshsize and the disctrete velocities. Therefore, Lattice Boltzmann discretization is less suitable for stationary fluid dynamic problems, since a large number of iterations is needed until convergence is reached. On the other side, a finite difference discetization can apply a multigrid method in order to obtain fast convergence.

6 Maxwell's Equations

6.1 Maxwell's Equations

The solution of Maxwell's equations in 3D is

- $\vec{E}$: the electrical field and

- $\vec{H}$: the magnetic field.

Given are

- μ: magnetic permeability,
- ϵ: electric permittivity,
- $\vec{M}$: equivalent magnetic current density,
- $\vec{J}$: electric current density.

Maxwell's equations are:

$$\begin{aligned} \frac{\partial \vec{H}}{\partial t} &= -\frac{1}{\mu}\nabla \times \vec{E} - \frac{1}{\mu}\vec{M}, \\ \frac{\partial \vec{E}}{\partial t} &= \frac{1}{\epsilon}\nabla \times \vec{H} - \frac{1}{\epsilon}\vec{J}. \end{aligned}$$

6.2 Finite Difference Time Domain Discretization (FDTD)

Let τ be a time step.
Time approximation:

- $\vec{E}|^{n+\frac{1}{2}}$: approximation at time point $(n+\frac{1}{2})\tau$.
- $\vec{H}|^{n}$: approximation at time point $n\tau$.

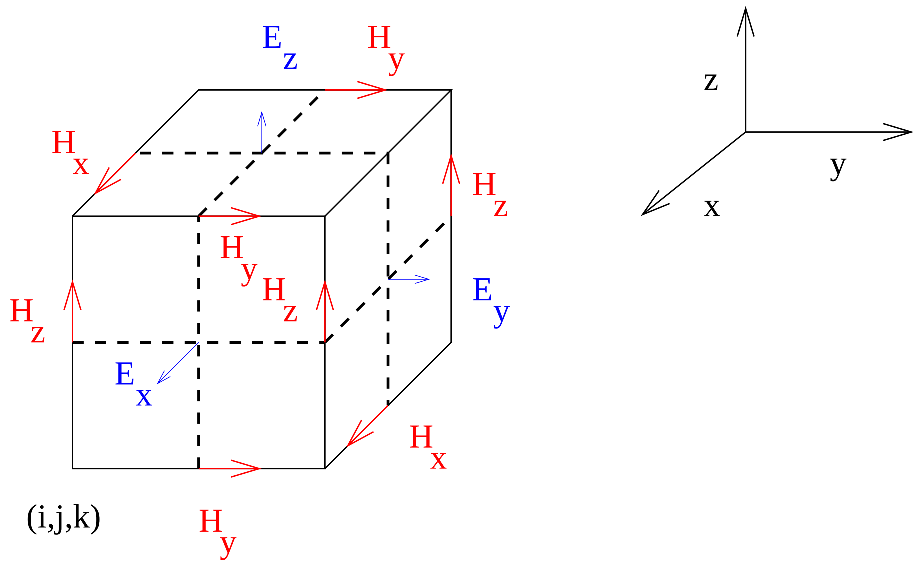
Furthermore, let us use the following abbreviation:

$$\begin{aligned} \vec{H}|^{n+\frac{1}{2}} &:= \frac{1}{2}\left(\vec{H}|^{n+1} + \vec{H}|^{n}\right), \\ \vec{E}|^{n} &:= \frac{1}{2}\left(\vec{E}|^{n+\frac{1}{2}} + \vec{E}|^{n-\frac{1}{2}}\right). \end{aligned}$$

Let h be a mesh size.
Space approximation:

- $E_x|^{n+\frac{1}{2}}_{i,j+\frac{1}{2},k+\frac{1}{2}}$: at point $(ih, (j+\frac{1}{2})h, (k+\frac{1}{2})h)$ (yz-face) .

- $E_y|^{n+\frac{1}{2}}_{i+\frac{1}{2},j,k+\frac{1}{2}}$: at point $((i+\frac{1}{2})h, jh, (k+\frac{1}{2})h)$ (xz-face).
- $E_z|^{n+\frac{1}{2}}_{i+\frac{1}{2},j+\frac{1}{2},k}$: at point $((i+\frac{1}{2})h, (j+\frac{1}{2})h, kh)$ (xy-face).
- $H_x|^{n}_{i+\frac{1}{2},j,k}$: at point $((i+\frac{1}{2})h, jh, kh)$ (x-edge).
- $H_y|^{n}_{i,j+\frac{1}{2},k}$: at point $(ih, (j+\frac{1}{2})h, kh)$ (y-edge).
- $H_z|^{n}_{i,j,k+\frac{1}{2}}$: at point $(ih, jh, (k+\frac{1}{2})h)$ (z-edge).



Now, the Maxwell's equations

$$\frac{\partial E_x}{\partial t} = \frac{1}{\epsilon}\left(\frac{\partial H_y}{\partial z} - \frac{\partial H_z}{\partial y} - J_x\right)$$

is discretized as follows:

$$\frac{E_x|^{n+\frac{1}{2}}_{i,j+\frac{1}{2},k+\frac{1}{2}} - E_x|^{n-\frac{1}{2}}_{i,j+\frac{1}{2},k+\frac{1}{2}}}{\tau} =$$

$$\frac{1}{\epsilon_{i,j+\frac{1}{2},k+\frac{1}{2}}} \left(\frac{H_y|^{n}_{i,j+\frac{1}{2},k+1} - H_y|^{n}_{i,j+\frac{1}{2},k}}{h} - \frac{H_z|^{n}_{i,j+1,k+\frac{1}{2}} - H_z|^{n}_{i,j,k+\frac{1}{2}}}{h} - J_x|^{n}_{i,j+\frac{1}{2},k+\frac{1}{2}}\right)$$

The other Maxwell's equations are discretized analogously.
$\vec{J}$ has to be composed as follows:

$$\vec{J} = \vec{J}_{source} + \sigma \vec{E},$$

where σ is the electric conductivity.
$\vec{E}$ is approximated by

$$\vec{E}|^{n} = \frac{1}{2}\left(\vec{E}|^{n+\frac{1}{2}} + \vec{E}|^{n-\frac{1}{2}}\right).$$

Reflecting boundary conditions can be modeled by pure Dirichlet boundary conditions.
Non-reflecting boundary conditions can be discretized by the Perfect Matched Layer (PML) method. These are not Neumann boundary conditions!

References

[1] D. Braess. *Finite Elemente.* Springer, New York, Berlin, Heidelberg, 1997. English translation available.

[2] William L. Briggs, Van Emden Henson, and Steve F. McCormick. *A Multigrid Tutorial, Second Edition.* Frontiers in Applied Mathematics. SIAM, Philadelphia, 2000.

[3] J. H. Ferziger and M. Peric. *Computational Methods for Fluid Dynamics.* Springer, 1996.

[4] C.A.J. Fletcher. *Computational Techniques for Fluid Dynamics.* Springer, Berlin, 2nd edition, 1991.

[5] A. Taflove and S. C. Hagness. *Computational Electrodynamics.* Artech House, Boston, London, 2000.