The heterogeneous multiscale finite element method and its implementation using DUNE.

Patrick Henning and Mario Ohlberger

Dune User Meeting 2010

Stuttgart, October 6, 2010

Westfälische
Wilhelms-Universität
Münster

Outline:

- Motivation: multiscale problems

Institut für Numerische und Angewandte Mathematik

Outline:

- Motivation: multiscale problems
- The heterogeneous multiscale finite element method (HMM)

Outline:

- Motivation: multiscale problems
- The heterogeneous multiscale finite element method (HMM)
- Implementation in DUNE

Institut für Numerische und Angewandte Mathematik

Outline:

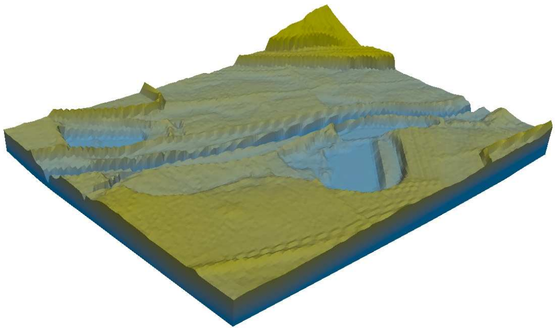
- Motivation: multiscale problems
- The heterogeneous multiscale finite element method (HMM)
- Implementation in DUNE
- Numerical results

Institut für Numerische und Angewandte Mathematik

Motivation: multiscale problems

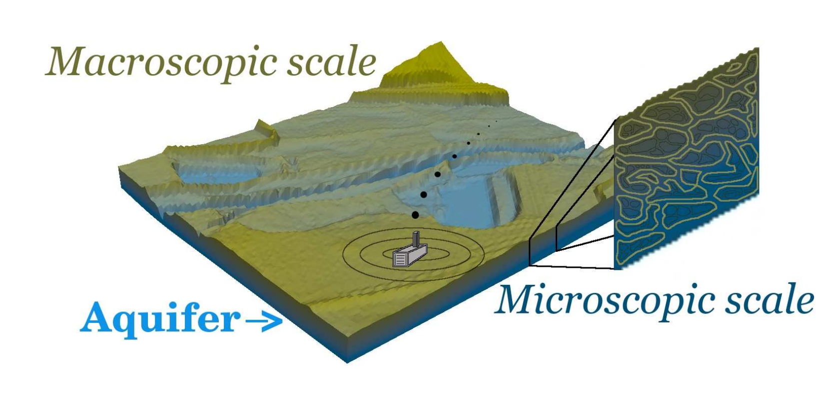
Institut für Numerische
und Angewandte
Mathematik

Introducing example:



Groundwater: important source of drinking water.

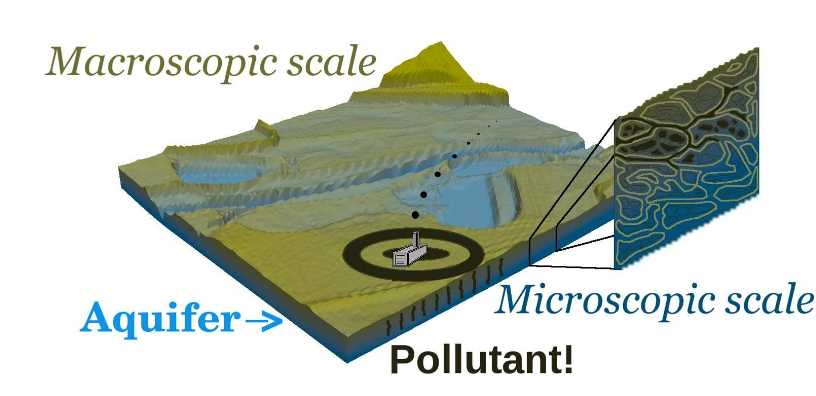
Introducing example:



Groundwater: important source of drinking water.

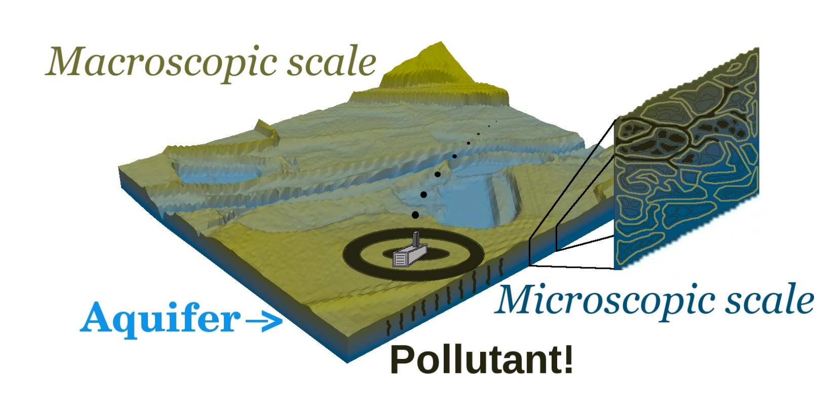
Institut für Numerische
und Angewandte
Mathematik

Introducing example:



Groundwater: important source of drinking water.

Introducing example:



$\Rightarrow$ Problem: concentration of a pollutant in groundwater?

Transport of pollutants in groundwater.

Simple approaches to modelling:

Institut für Numerische und Angewandte Mathematik

Transport of pollutants in groundwater.

Simple approaches to modelling:

1. Stationary problem / linear elliptic.

$$\nabla \cdot (A^{\varepsilon}(x)\nabla u_{\varepsilon}(x)) = f(x) \quad \text{in } \Omega,$$
$$u_{\varepsilon}(x) = 0 \quad \text{on } \partial\Omega.$$

- u_{ε}: concentration of pollutant,
- ε: indicator for (representative) size of small scale,

Transport of pollutants in groundwater.

Simple approaches to modelling:

1. Stationary problem / linear elliptic.

$$\nabla \cdot (A^{\varepsilon}(x)\nabla u_{\varepsilon}(x)) = f(x) \quad \text{in } \Omega,$$
$$u_{\varepsilon}(x) = 0 \quad \text{on } \partial\Omega.$$

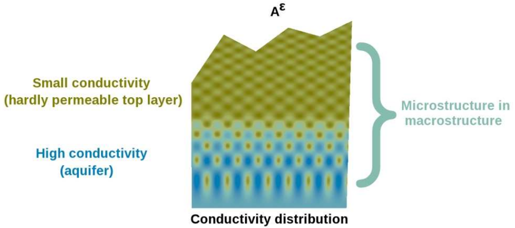
- u_{ε}: concentration of pollutant,
- ε: indicator for (representative) size of small scale,
- A^{ε}: conductivity / diffusion operator.

Transport of pollutants in groundwater.

Simple approaches to modelling:

1. Stationary problem / linear elliptic.

$$
\begin{aligned}
\nabla \cdot (A^{\varepsilon}(x) \nabla u_{\varepsilon}(x)) &= f(x) \quad \text{in } \Omega, \\
u_{\varepsilon}(x) &= 0 \quad \text{on } \partial\Omega.
\end{aligned}
$$

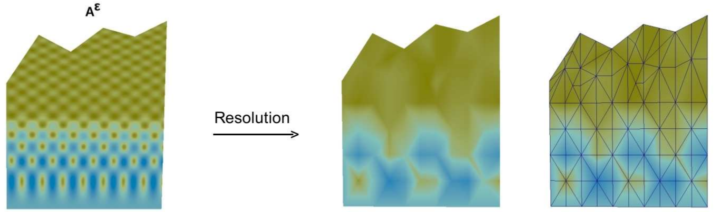


Transport of pollutants in groundwater.

Simple approaches to modelling:

1. Stationary problem / linear elliptic.

$$\nabla \cdot (A^{\varepsilon}(x)\nabla u_{\varepsilon}(x)) = f(x) \quad \text{in } \Omega,$$
$$u_{\varepsilon}(x) = 0 \quad \text{on } \partial\Omega.$$



Conductivity distribution
(Still relatively coarse!)

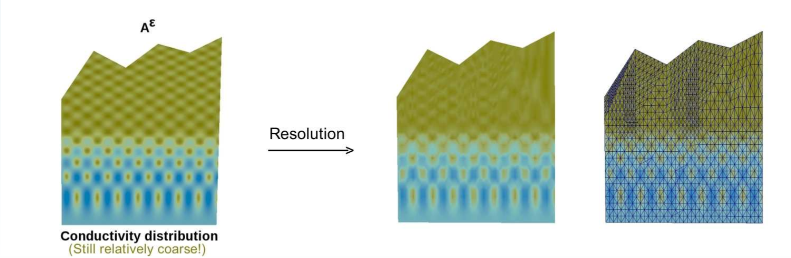
Institut für Numerische und Angewandte Mathematik

Transport of pollutants in groundwater.

Simple approaches to modelling:

1. Stationary problem / linear elliptic.

$$\nabla \cdot (A^{\varepsilon}(x)\nabla u_{\varepsilon}(x)) = f(x) \quad \text{in } \Omega,$$
$$u_{\varepsilon}(x) = 0 \quad \text{on } \partial\Omega.$$



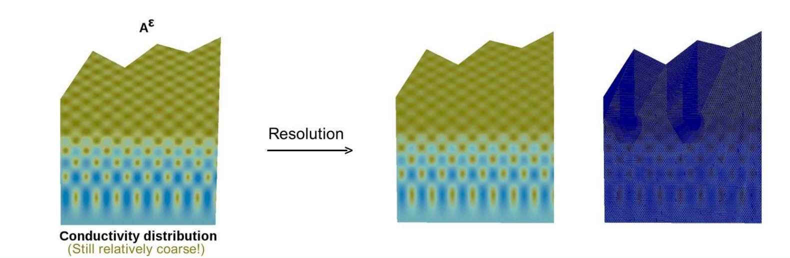
Institut für Numerische und Angewandte Mathematik

Transport of pollutants in groundwater.

Simple approaches to modelling:

1. Stationary problem / linear elliptic.

$$\nabla \cdot (A^{\varepsilon}(x)\nabla u_{\varepsilon}(x)) = f(x) \quad \text{in } \Omega,$$
$$u_{\varepsilon}(x) = 0 \quad \text{on } \partial\Omega.$$

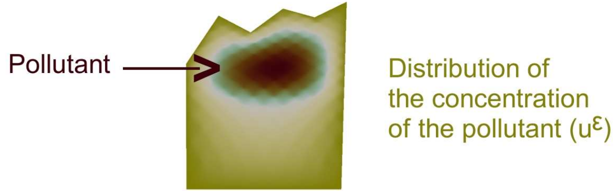


Transport of pollutants in groundwater.

Simple approaches to modelling:

1. Stationary problem / linear elliptic.

$$\nabla \cdot (A^{\varepsilon}(x) \nabla u_{\varepsilon}(x)) = f(x) \quad \text{in } \Omega,$$
$$u_{\varepsilon}(x) = 0 \quad \text{on } \partial\Omega.$$



Transport of pollutants in groundwater.

Simple approaches to modelling:

2. Stationary problem / nonlinear elliptic.

$$\nabla \cdot A^{\varepsilon}(x, \nabla u_{\varepsilon}(x)) = f(x) \quad \text{in } \Omega,$$
$$u_{\varepsilon}(x) = 0 \quad \text{on } \partial\Omega.$$

- u_{ε}: concentration of pollutant,
- ε: indicator for (representative) size of small scale,
- A^{ε}: conductivity / diffusion operator.

Transport of pollutants in groundwater.

Simple approaches to modelling:

3. Nonstationary problem / linear parabolic.

$$\partial_t u_\varepsilon - \nabla \cdot (A^\varepsilon \nabla u_\varepsilon) + \mathrm{Pe}\, b^\varepsilon \cdot \nabla u_\varepsilon = 0 \quad \text{in } \mathbb{R}^d \times (0, T_0),$$
$$u_\varepsilon(\cdot, 0) = v_0 \quad \text{in } \mathbb{R}^d.$$

- u_ε: concentration of pollutant,
- ε: indicator for (representative) size of small scale,
- A^ε: conductivity / diffusion operator.

Institut für Numerische und Angewandte Mathematik

Transport of pollutants in groundwater.

Simple approaches to modelling:

3. Nonstationary problem / linear parabolic.

$$\partial_t u_\varepsilon - \nabla \cdot (A^\varepsilon \nabla u_\varepsilon) + \mathrm{Pe}\, b^\varepsilon \cdot \nabla u_\varepsilon = 0 \quad \text{in } \mathbb{R}^d \times (0, T_0),$$
$$u_\varepsilon(\cdot, 0) = v_0 \quad \text{in } \mathbb{R}^d.$$

- u_ε: concentration of pollutant,
- ε: indicator for (representative) size of small scale,
- A^ε: conductivity / diffusion operator.
- b^ε: transport velocity / advection operator,

Transport of pollutants in groundwater.

Simple approaches to modelling:

3. Nonstationary problem / linear parabolic.

$$\partial_t u_\varepsilon - \nabla \cdot (A^\varepsilon \nabla u_\varepsilon) + \mathrm{Pe}\, b^\varepsilon \cdot \nabla u_\varepsilon = 0 \quad \text{in } \mathbb{R}^d \times (0, T_0),$$
$$u_\varepsilon(\cdot, 0) = v_0 \quad \text{in } \mathbb{R}^d.$$

- u_ε: concentration of pollutant,
- ε: indicator for (representative) size of small scale,
- A^ε: conductivity / diffusion operator.
- b^ε: transport velocity / advection operator,
- v_0: initial concentration of the pollutant.

Transport of pollutants in groundwater.

Simple approaches to modelling:

3. Nonstationary problem / linear parabolic.

$$\partial_t u_\varepsilon - \nabla \cdot (A^\varepsilon \nabla u_\varepsilon) + \mathrm{Pe}\, b^\varepsilon \cdot \nabla u_\varepsilon = 0 \quad \text{in } \mathbb{R}^d \times (0, T_0),$$
$$u_\varepsilon(\cdot, 0) = v_0 \quad \text{in } \mathbb{R}^d.$$

- u_ε: concentration of pollutant,
- ε: indicator for (representative) size of small scale,
- A^ε: conductivity / diffusion operator.
- b^ε: transport velocity / large Péclet number,
- v_0: initial concentration of the pollutant.

Westfälische Wilhelms-Universität Münster

Transport of pollutants in groundwater.

Simple approaches to modelling:

3. Nonstationary problem / linear parabolic.

$$\partial_t u_\varepsilon - \nabla \cdot (A^\varepsilon \nabla u_\varepsilon) + \varepsilon^{-1} b^\varepsilon \cdot \nabla u_\varepsilon = 0 \quad \text{in } \mathbb{R}^d \times (0, T_0),$$
$$u_\varepsilon(\cdot, 0) = v_0 \quad \text{in } \mathbb{R}^d.$$

- u_ε: concentration of pollutant,
- ε: indicator for (representative) size of small scale,
- A^ε: conductivity / diffusion operator.
- b^ε: transport velocity / large Péclet number, large drift,
- v_0: initial concentration of the pollutant.

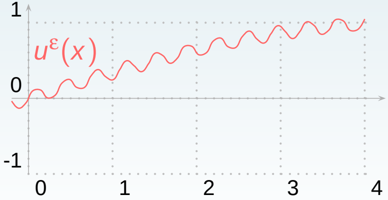
Institut für Numerische und Angewandte Mathematik

The heterogeneous multiscale method

Institut für Numerische und Angewandte Mathematik

Idea in short (linear elliptic case):

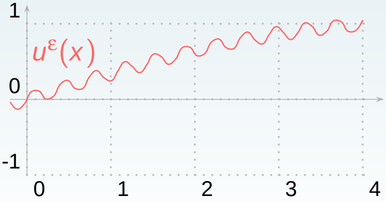
$u^{\varepsilon}(x)$



Institut für Numerische und Angewandte Mathematik

Idea in short (linear elliptic case):

$$u^{\varepsilon}(x)$$
$$\pitchfork$$
$$H(\Omega)$$



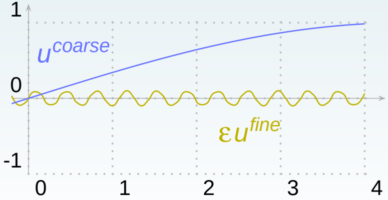
Institut für Numerische und Angewandte Mathematik

Idea in short (linear elliptic case):

$$u^{\varepsilon}(x) = u^{coarse}(x) + \varepsilon\, u^{fine}(x)$$

$$\pitchfork \qquad \pitchfork \qquad \pitchfork$$

$$H(\Omega) = H^{c}(\Omega) \oplus \varepsilon\, H^{f}(\Omega)$$



Institut für Numerische und Angewandte Mathematik

Idea in short (linear elliptic case):

$$u^{\varepsilon}(x) = u^{coarse}(x) + \varepsilon\, u^{fine}(x)$$

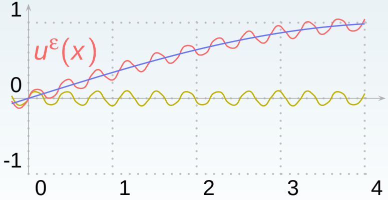
$$\pitchfork \qquad \pitchfork \qquad \pitchfork$$

$$H(\Omega) = H^{c}(\Omega) \oplus \varepsilon\, H^{f}(\Omega)$$

$$\downarrow \qquad \downarrow \qquad \downarrow$$

$$V_{H,h} = V_{H} \oplus \varepsilon\, W_{h}$$

DISCRETIZATION

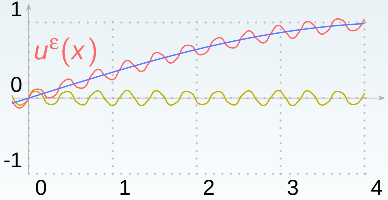


Institut für Numerische und Angewandte Mathematik

Idea in short (linear elliptic case):

$$
\begin{array}{ccccc}
u^{\varepsilon}(x) & = & u^{coarse}(x) & + & \varepsilon\, u^{fine}(x) \\
\pitchfork & & \pitchfork & & \pitchfork \\
H(\Omega) & = & H^{c}(\Omega) & \oplus & \varepsilon\, H^{f}(\Omega) \\
\downarrow & & \downarrow & & \downarrow \\
V_{H,h} & = & V_H & \oplus & \varepsilon\, W_h \\
\Cup & & \Cup & & \Cup \\
R_h(u_H) & = & u_H & + & \varepsilon\, Q_h(u_H)
\end{array}
$$

DISCRETIZATION

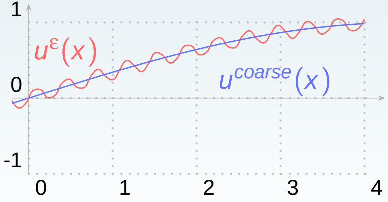


Institut für Numerische und Angewandte Mathematik

Idea in short (linear elliptic case):

$$
\begin{array}{ccccc}
u^{\varepsilon}(x) & = & u^{coarse}(x) & + & \varepsilon\, u^{fine}(x) \\
\pitchfork & & \pitchfork & & \pitchfork \\
H(\Omega) & = & H^{c}(\Omega) & \oplus & \varepsilon\, H^{f}(\Omega) \\
\downarrow & & \downarrow & & \downarrow \\
V_{H,h} & = & V_{H} & \oplus & \varepsilon\, W_{h} \\
\Cup & & \Cup & & \Cup \\
R_h(u_H) & = & u_H & + & \varepsilon\, Q_h(u_H)
\end{array}
$$

DISCRETIZATION



GOAL:

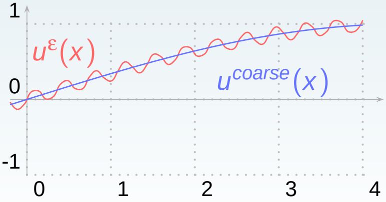
Approximation of $u^{coarse} \approx u^{\varepsilon}$

Institut für Numerische
und Angewandte
Mathematik

Idea in short (linear elliptic case):

$$
\begin{array}{ccccc}
u^{\varepsilon}(x) & = & u^{coarse}(x) & + & \varepsilon\, u^{fine}(x) \\
\Cap & & \Cap & & \Cap \\
H(\Omega) & = & H^{c}(\Omega) & \oplus & \varepsilon\, H^{f}(\Omega) \\
\downarrow & & \downarrow & & \downarrow \\
V_{H,h} & = & V_H & \oplus & \varepsilon\, W_h \\
\Cup & & \Cup & & \Cup \\
R_h(u_H) & = & \boxed{u_H} & + & \varepsilon\, Q_h(u_H)
\end{array}
$$

DISCRETIZATION



GOAL:

Approximation of $u^{coarse} \approx u^{\varepsilon}$, i.e.

FIND $\boxed{u_H}$!

Institut für Numerische und Angewandte Mathematik

Idea in short (linear elliptic case):

$$
\begin{array}{ccccc}
H(\Omega) & = & H^c(\Omega) & \oplus & \varepsilon H^f(\Omega) \\
\Cup & = & \Cup & & \Cup \\
u^\varepsilon(x) & = & u^{coarse}(x) & + & \varepsilon u^{fine}(x)
\end{array}
$$

$$\int_\Omega A^\varepsilon \nabla u^\varepsilon \cdot \nabla \Phi^\varepsilon = \int_\Omega f \Phi^\varepsilon \qquad \forall \Phi^\varepsilon \in \mathring{H}^1(\Omega)$$

Institut für Numerische und Angewandte Mathematik

Idea in short (linear elliptic case):

$$\begin{array}{ccccc} H(\Omega) & = & H^c(\Omega) & \oplus & \varepsilon H^f(\Omega) \\ \rotatebox{90}{$\in$} & = & \rotatebox{90}{$\in$} & & \rotatebox{90}{$\in$} \\ u^\varepsilon(x) & = & u^{coarse}(x) & + & \varepsilon u^{fine}(x) \end{array}$$

$$\int_\Omega A^\varepsilon \nabla u^\varepsilon \cdot \nabla \Phi^\varepsilon = \int_\Omega f \Phi^\varepsilon \qquad \forall \Phi^\varepsilon \in \mathring{H}^1(\Omega)$$

$$\int_\Omega A^\varepsilon \nabla u^\varepsilon \cdot \left(\nabla \Phi^{coarse} + \varepsilon \nabla \Phi^{fine} \right) = \int_\Omega f \left(\Phi^{coarse} + \varepsilon \Phi^{fine} \right)$$

$$\forall \Phi^{coarse} \in H^c(\Omega), \quad \forall \Phi^{fine} \in H^f(\Omega).$$

Institut für Numerische und Angewandte Mathematik

Idea in short (linear elliptic case):

$$\begin{array}{cccc} H(\Omega) & = H^c(\Omega) & \oplus\, \varepsilon H^f(\Omega) \\ \Cup & = \quad \Cup & \Cup \\ u^\varepsilon(x) & = u^{coarse}(x) & +\, \varepsilon\, u^{fine}(x) \end{array}$$

$$\int_\Omega A^\varepsilon \nabla u^\varepsilon \cdot \nabla \Phi^\varepsilon = \int_\Omega f \Phi^\varepsilon \qquad \forall \Phi^\varepsilon \in \mathring{H}^1(\Omega)$$

$$\int_\Omega A^\varepsilon \nabla u^\varepsilon \cdot \nabla \Phi^{coarse} = \int_\Omega f\, \Phi^{coarse}$$

$$\forall \Phi^{coarse} \in H^c(\Omega)$$

$$\Phi^{fine} = 0$$

Idea in short (linear elliptic case):

$$H(\Omega) = H^c(\Omega) \oplus \varepsilon H^f(\Omega)$$
$$\Cup \quad = \quad \Cup \quad \Cup$$
$$u^\varepsilon(x) = u^{coarse}(x) + \varepsilon u^{fine}(x)$$

$$\int_\Omega A^\varepsilon \nabla u^\varepsilon \cdot \nabla \Phi^\varepsilon = \int_\Omega f \Phi^\varepsilon \qquad \forall \Phi^\varepsilon \in \mathring{H}^1(\Omega)$$

$$\int_\Omega A^\varepsilon \nabla u^\varepsilon \cdot \quad \varepsilon \nabla \Phi^{fine} = \int_\Omega f \quad \varepsilon \Phi^{fine}$$
$$\forall \Phi^{fine} \in H^f(\Omega).$$

$$\Phi^{coarse} = 0$$

Institut für Numerische und Angewandte Mathematik

Idea in short (linear elliptic case):

$$\begin{array}{ccccc} H(\Omega) & = H^{c}(\Omega) & \oplus\, \varepsilon H^{f}(\Omega) \\ \Cup & \Cup & \Cup \\ u^{\varepsilon}(x) & = u^{coarse}(x) & +\, \varepsilon\, u^{fine}(x) \end{array}$$

$$\int_{\Omega} A^{\varepsilon}(\nabla u^{\varepsilon}) \cdot \nabla \Phi^{coarse} = \int_{\Omega} f \Phi^{coarse} \qquad \forall \Phi^{coarse} \in H^{c}(\Omega),$$

$$\int_{\Omega} A^{\varepsilon}(\nabla u^{\varepsilon}) \cdot \varepsilon \nabla \Phi^{fine} = \int_{\Omega} f \varepsilon \Phi^{fine} \qquad \forall \Phi^{fine} \in H^{f}(\Omega)$$

Idea in short (linear elliptic case):

$$\begin{array}{ccccc} H(\Omega) & = H^c(\Omega) & \oplus\, \varepsilon H^f(\Omega) \\ \Cup & \Cup & \Cup \\ u^\varepsilon(x) & = u^{coarse}(x) & +\, \varepsilon\, u^{fine}(x) \end{array}$$

$$\int_\Omega A^\varepsilon(\nabla u^\varepsilon)\cdot\nabla\Phi^{coarse} = \int_\Omega f\,\Phi^{coarse} \qquad \forall \Phi^{coarse}\in H^c(\Omega),$$

$$\int_\Omega A^\varepsilon(\nabla u^\varepsilon)\cdot\varepsilon\nabla\Phi^{fine} = \int_\Omega f\,\varepsilon\Phi^{fine} \qquad \forall \Phi^{fine}\in H^f(\Omega)$$

Idea in short (linear elliptic case):

$$\begin{array}{ccccc} H(\Omega) & = H^c(\Omega) & \oplus\, \varepsilon H^f(\Omega) \\ \Cup & \Cup & \Cup \\ u^\varepsilon(x) & = u^{coarse}(x) & +\, \varepsilon\, u^{fine}(x) \end{array}$$

$$\int_\Omega A^\varepsilon(\nabla u^\varepsilon) \cdot \nabla \Phi^{coarse} = \int_\Omega f \Phi^{coarse} \qquad \forall \Phi^{coarse} \in H^c(\Omega),$$

$$\int_\Omega A^\varepsilon(\nabla u^\varepsilon) \cdot \varepsilon \nabla \Phi^{fine} = \int_\Omega f \varepsilon \Phi^{fine} \approx 0 \quad \forall \Phi^{fine} \in H^f(\Omega)$$

Institut für Numerische
und Angewandte
Mathematik

Idea in short (linear elliptic case):

$$H(\Omega) = H^c(\Omega) \oplus \varepsilon H^f(\Omega)$$
$$\Cup \qquad \Cup \qquad \Cup$$
$$u^\varepsilon(x) = u^{coarse}(x) + \varepsilon u^{fine}(x)$$

$$\int_\Omega A^\varepsilon(\nabla u^\varepsilon) \cdot \nabla \Phi^{coarse} = \int_\Omega f \Phi^{coarse} \qquad \forall \Phi^{coarse} \in H^c(\Omega),$$
$$\int_\Omega A^\varepsilon(\nabla u^\varepsilon) \cdot \varepsilon \nabla \Phi^{fine} = \int_\Omega f \varepsilon \Phi^{fine} \approx 0 \qquad \forall \Phi^{fine} \in H^f(\Omega)$$

Idea in short (linear elliptic case):

$$\begin{array}{cccc} H(\Omega) & = H^c(\Omega) & \oplus\, \varepsilon H^f(\Omega) \\ \Cup & \Cup & \Cup \\ u^\varepsilon(x) & = u^{coarse}(x) & +\, \varepsilon\, u^{fine}(x) \end{array}$$

$$\int_\Omega A^\varepsilon(\nabla u^\varepsilon) \cdot \nabla \Phi^{coarse} = \int_\Omega f \Phi^{coarse} \qquad \forall \Phi^{coarse} \in H^c(\Omega),$$

$$\int_\Omega A^\varepsilon(\nabla u^\varepsilon) \cdot \varepsilon \nabla \Phi^{fine} \approx 0 \qquad \forall \Phi^{fine} \in H^f(\Omega)$$

Institut für Numerische und Angewandte Mathematik

Idea in short (linear elliptic case):

$$
\begin{array}{cccc}
H(\Omega) & = H^c(\Omega) & \oplus\, \varepsilon H^f(\Omega) \\
\Cup & \Cup & \Cup \\
u^\varepsilon(x) & = u^{coarse}(x) & +\, \varepsilon\, u^{fine}(x)
\end{array}
$$

$$\int_\Omega A^\varepsilon(\nabla u^\varepsilon) \cdot \nabla \Phi^{coarse} = \int_\Omega f \Phi^{coarse} \qquad \forall \Phi^{coarse} \in H^c(\Omega),$$

$$\int_\Omega A^\varepsilon(\nabla u^\varepsilon) \cdot \varepsilon \nabla \Phi^{fine} \approx 0 \qquad \forall \Phi^{fine} \in H^f(\Omega)$$

Institut für Numerische
und Angewandte
Mathematik

Idea in short (linear elliptic case):

$$\begin{array}{lll} R_h(u_H) & = u_H & + \varepsilon Q_h(u_H) \\ \wr\wr & \wr\wr & \wr\wr \\ u^\varepsilon & = u^{coarse} & + \varepsilon u^{fine} \end{array}$$

$$\int_\Omega A^\varepsilon(\nabla u^\varepsilon) \cdot \nabla \Phi^{coarse} = \int_\Omega f \Phi^{coarse} \qquad \forall \Phi^{coarse} \in H^c(\Omega),$$

$$\int_\Omega A^\varepsilon(\nabla u^\varepsilon) \cdot \varepsilon \nabla \Phi^{fine} \approx 0 \qquad \forall \Phi^{fine} \in H^f(\Omega)$$

$$\int_\Omega A^\varepsilon(\nabla R_h(u_H)) \cdot \nabla \Phi_H = \int_\Omega f \Phi_H \qquad \forall \Phi_H \in V_H,$$

Institut für Numerische und Angewandte Mathematik

Idea in short (linear elliptic case):

$$
\begin{array}{ccccc}
R_h(u_H) & = u_H & & + \, \varepsilon Q_h(u_H) \\
\wr & \wr & & \wr \\
u^{\varepsilon} & = u^{coarse} & & + \, \varepsilon \, u^{fine}
\end{array}
$$

$$\int_\Omega A^\varepsilon(\nabla u^\varepsilon) \cdot \nabla \Phi^{coarse} = \int_\Omega f \Phi^{coarse} \qquad \forall \Phi^{coarse} \in H^c(\Omega),$$

$$\int_\Omega A^\varepsilon(\nabla u^\varepsilon) \cdot \varepsilon \nabla \Phi^{fine} \approx 0 \qquad \forall \Phi^{fine} \in H^f(\Omega)$$

$$\int_\Omega A^\varepsilon(\nabla R_h(u_H)) \cdot \nabla \Phi_H = \int_\Omega f \Phi_H \qquad \forall \Phi_H \in V_H,$$

$$\int_\Omega A^\varepsilon(\nabla R_h(u_H)) \cdot \nabla \phi_h = 0 \qquad \forall \phi_h \in W_h.$$

Idea in short (linear elliptic case):

$$\int_\Omega A^\varepsilon(\nabla R_h(u_H)) \cdot \nabla \Phi_H = \int_\Omega f \Phi_H \qquad \forall \Phi_H \in V_H,$$

$$\int_\Omega A^\varepsilon(\nabla R_h(u_H)) \cdot \nabla \phi_h = 0 \qquad \forall \phi_h \in W_h.$$

Idea in short (linear elliptic case):

$$\int_\Omega A^\varepsilon(\nabla R_h(u_H)) \cdot \nabla \Phi_H = \int_\Omega f \Phi_H \qquad \forall \Phi_H \in V_H,$$

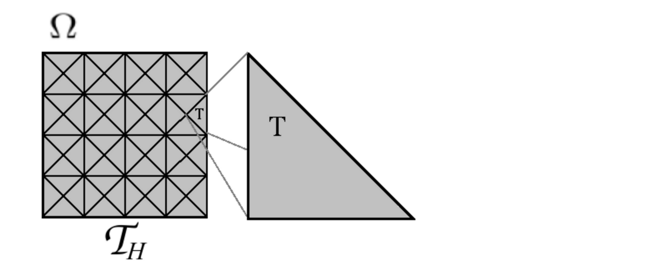
$$\int_\Omega A^\varepsilon(\nabla R_h(u_H)) \cdot \nabla \phi_h = 0 \qquad \forall \phi_h \in W_h.$$

Institut für Numerische und Angewandte Mathematik

Idea in short (linear elliptic case):

$$\sum_{T \in \mathcal{T}_H} |T| \fint_T A^\varepsilon(\nabla R_h(u_H)) \cdot \nabla \Phi_H = \int_\Omega f \Phi_H \qquad \forall \Phi_H \in V_H,$$

$$\int_\Omega A^\varepsilon(\nabla R_h(u_H)) \cdot \nabla \phi_h = 0 \qquad \forall \phi_h \in W_h.$$

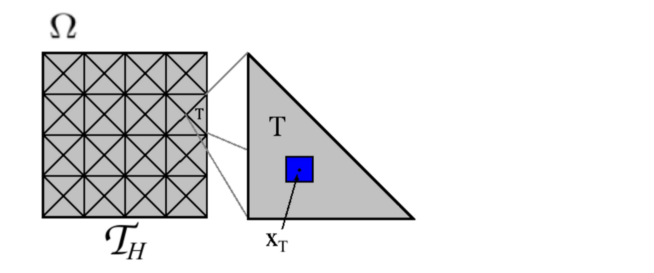


Institut für Numerische und Angewandte Mathematik

Idea in short (linear elliptic case):

$$\sum_{T\in\mathcal{T}_H} |T| \fint_T A^\varepsilon(\nabla R_h(u_H))\cdot\nabla\Phi_H = \int_\Omega f\Phi_H \qquad \forall\Phi_H\in V_H,$$

$$\int_\Omega A^\varepsilon(\nabla R_h(u_H))\cdot\nabla\phi_h = 0 \qquad \forall\phi_h\in W_h.$$

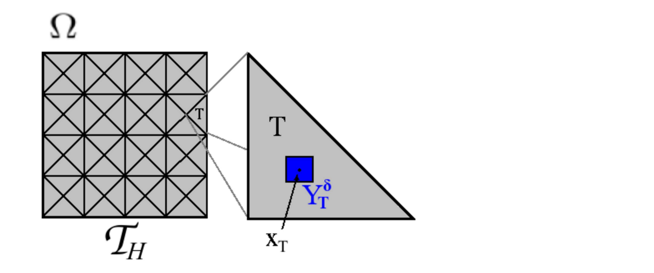


Institut für Numerische und Angewandte Mathematik

Idea in short (linear elliptic case):

$$\sum_{T \in \mathcal{T}_H} |T| \fint_{Y_T^\delta} A^\varepsilon(\nabla R_h(u_H)) \cdot \nabla \Phi_H = \int_\Omega f \Phi_H \qquad \forall \Phi_H \in V_H,$$

$$\int_\Omega A^\varepsilon(\nabla R_h(u_H)) \cdot \nabla \phi_h = 0 \qquad \forall \phi_h \in W_h.$$

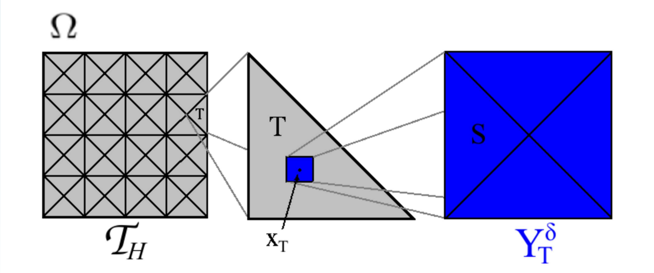


Institut für Numerische und Angewandte Mathematik

Idea in short (linear elliptic case):

$$\sum_{T \in \mathcal{T}_H} |T| \fint_{Y_T^\delta} A^\varepsilon(\nabla R_h(u_H)) \cdot \nabla \Phi_H = \int_\Omega f \Phi_H \qquad \forall \Phi_H \in V_H,$$

$$\int_{Y_T^\delta} A^\varepsilon(\nabla R_h(u_H)) \cdot \nabla \phi_h = 0 \qquad \forall \phi_h \in W_h(Y_T^\delta).$$

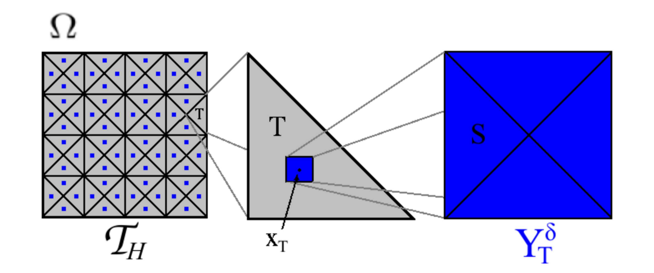


Institut für Numerische und Angewandte Mathematik

Idea in short (linear elliptic case):

$$\sum_{T\in\mathcal{T}_H} |T| \fint_{Y_T^\delta} A^\varepsilon(\nabla R_h(u_H))\cdot\nabla\Phi_H = \int_\Omega f\Phi_H \qquad \forall\Phi_H\in V_H,$$

$$\int_{Y_T^\delta} A^\varepsilon(\nabla R_h(u_H))\cdot\nabla\phi_h = 0 \qquad \forall\phi_h\in W_h(Y_T^\delta).$$



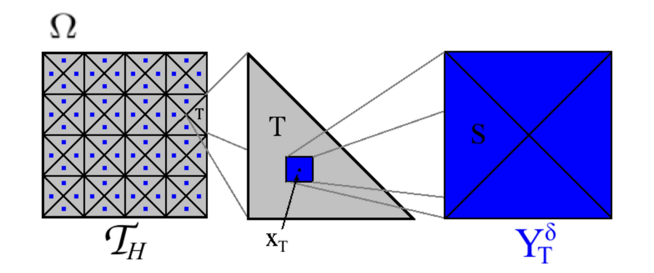
Institut für Numerische und Angewandte Mathematik

Idea in short (linear elliptic case):

$$\sum_{T \in \mathcal{T}_H} |T| \fint_{Y_T^\delta} A^\varepsilon(\nabla R_h(u_H)) \cdot \nabla \Phi_H = \int_\Omega f \Phi_H \qquad \forall \Phi_H \in V_H,$$

$$\int_{Y_T^\delta} A^\varepsilon(\nabla R_h(u_H)) \cdot \nabla \phi_h = 0 \qquad \forall \phi_h \in W_h(Y_T^\delta).$$

But, boundary condition?



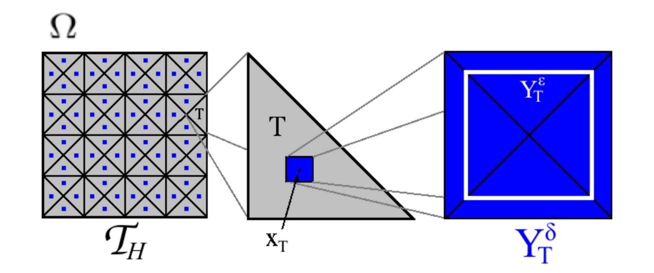
Institut für Numerische und Angewandte Mathematik

Idea in short (linear elliptic case):

$$\sum_{T \in \mathcal{T}_H} |T| \fint_{Y_T^\delta} A^\varepsilon(\nabla R_h(u_H)) \cdot \nabla \Phi_H = \int_\Omega f \Phi_H \qquad \forall \Phi_H \in V_H,$$

$$\int_{Y_T^\delta} A^\varepsilon(\nabla R_h(u_H)) \cdot \nabla \phi_h = 0 \qquad \forall \phi_h \in W_h(Y_T^\delta).$$

But, boundary condition?

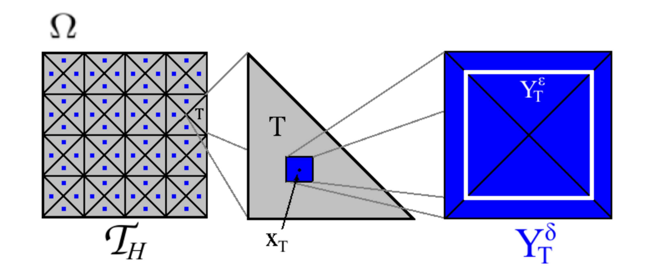


Idea in short (linear elliptic case):

$$\sum_{T\in\mathcal{T}_H} |T| \fint_{Y_T^{\varepsilon}} A^{\varepsilon}(\nabla R_h(u_H))\cdot\nabla\Phi_H = \int_{\Omega} f\Phi_H \qquad \forall \Phi_H \in V_H,$$

$$\int_{Y_T^{\delta}} A^{\varepsilon}(\nabla R_h(u_H))\cdot\nabla\phi_h = 0 \qquad \forall \phi_h \in W_h(Y_T^{\delta}).$$

But, boundary condition? Oversampling!

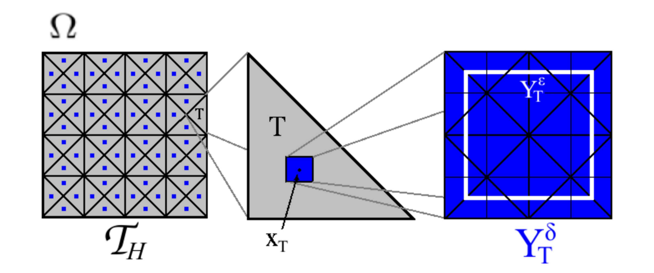


Idea in short (linear elliptic case):

$$\sum_{T\in\mathcal{T}_H} |T| \fint_{Y_T^\varepsilon} A^\varepsilon(\nabla R_h(u_H))\cdot\nabla\Phi_H = \int_\Omega f\Phi_H \qquad \forall\Phi_H\in V_H,$$

$$\int_{Y_T^\delta} A^\varepsilon(\nabla R_h(u_H))\cdot\nabla\phi_h = 0 \qquad \forall\phi_h\in W_h(Y_T^\delta).$$

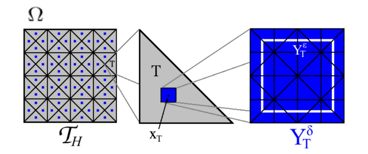
But, boundary condition? Oversampling!



Definition (HMM)

Find $u_H \in V_H$ with

$$\sum_{T \in \mathcal{T}_H} |T| \fint_{Y_T^\varepsilon} A^\varepsilon (\nabla R_T(u_H)) \cdot \nabla \Phi_H = \int_\Omega f \Phi_H \qquad \forall \Phi_H \in V_H,$$



Institut für Numerische und Angewandte Mathematik

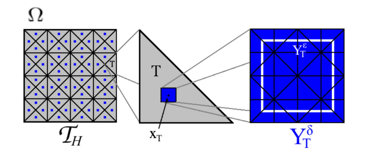
Definition (HMM)

Find $u_H \in V_H$ with

$$\sum_{T \in \mathcal{T}_H} |T| \fint_{Y_T^\varepsilon} A^\varepsilon(\nabla R_T(u_H)) \cdot \nabla \Phi_H = \int_\Omega f \Phi_H \qquad \forall \Phi_H \in V_H,$$

where $R_T(u_H) \in u_H + W_h(Y_T^\delta)$ solves

$$\int_{Y_T^\delta} A^\varepsilon(\nabla R_T(u_H)) \cdot \nabla \phi_h = 0 \qquad \forall \phi_h \in W_h(Y_T^\delta) \subset H^1_{per}(Y_T^\delta).$$



Institut für Numerische und Angewandte Mathematik

Implementation

Institut für Numerische und Angewandte Mathematik

Discrete macro-problem: find $u_H \in V_H$ with

$$\sum_{T \in \mathcal{T}_H} |T| \fint_{Y_T^\varepsilon} A^\varepsilon(\nabla R_T(u_H)) \cdot \nabla \Phi_H = \int_\Omega f \Phi_H \qquad \forall \Phi_H \in V_H.$$

Institut für Numerische
und Angewandte
Mathematik

Discrete macro-problem: find $u_H \in V_H$ with

$$A_H(u_H, \Phi_H) := \sum_{T \in \mathcal{T}_H} |T| \fint_{Y_T^\varepsilon} A^\varepsilon(\nabla R_T(u_H)) \cdot \nabla \Phi_H = \int_\Omega f \Phi_H \qquad \forall \Phi_H \in V_H.$$

Discrete macro-problem: find $u_H \in V_H$ with

$$A_H(u_H, \Phi_H) := \sum_{T \in \mathcal{T}_H} |T| \fint_{Y_T^\varepsilon} A^\varepsilon(\nabla R_T(u_H)) \cdot \nabla \Phi_H = \int_\Omega f \Phi_H \qquad \forall \Phi_H \in V_H.$$

Linear!

Discrete macro-problem: find $u_H \in V_H$ with

$$A_H(u_H, \Phi_H) := \sum_{T \in \mathcal{T}_H} |T| \fint_{Y_T^\varepsilon} A^\varepsilon(\nabla R_T(u_H)) \cdot \nabla \Phi_H = \int_\Omega f \Phi_H \qquad \forall \Phi_H \in V_H.$$

We obtain the following algebraic system:

Institut für Numerische
und Angewandte
Mathematik

Discrete macro-problem: find $u_H \in V_H$ with

$$A_H(u_H, \Phi_H) := \sum_{T \in \mathcal{T}_H} |T| \fint_{Y_T^\varepsilon} A^\varepsilon(\nabla R_T(u_H)) \cdot \nabla \Phi_H = \int_\Omega f \Phi_H \qquad \forall \Phi_H \in V_H.$$

We obtain the following algebraic system:

$$\begin{pmatrix} A_H(\Phi_1, \Phi_1) & \dots & A_H(\Phi_1, \Phi_N) \\ \vdots & & \vdots \\ A_H(\Phi_N, \Phi_1) & \dots & A_H(\Phi_N, \Phi_N) \end{pmatrix} \begin{pmatrix} \alpha_1 \\ \vdots \\ \alpha_N \end{pmatrix} = \begin{pmatrix} (f, \Phi_1)_{L^2(\Omega)} \\ \vdots \\ (f, \Phi_N)_{L^2(\Omega)} \end{pmatrix}$$

Discrete macro-problem: find $u_H \in V_H$ with

$$A_H(u_H, \Phi_H) := \sum_{T \in \mathcal{T}_H} |T| \fint_{Y_T^\varepsilon} A^\varepsilon(\nabla R_T(u_H)) \cdot \nabla \Phi_H = \int_\Omega f \Phi_H \qquad \forall \Phi_H \in V_H.$$

We obtain the following algebraic system:

$$\begin{pmatrix} A_H(\Phi_1, \Phi_1) & \dots & A_H(\Phi_1, \Phi_N) \\ \vdots & & \vdots \\ A_H(\Phi_N, \Phi_1) & \dots & A_H(\Phi_N, \Phi_N) \end{pmatrix} \begin{pmatrix} \alpha^1 \\ \vdots \\ \alpha^N \end{pmatrix} = \begin{pmatrix} (f, \Phi_1)_{L^2(\Omega)} \\ \vdots \\ (f, \Phi_N)_{L^2(\Omega)} \end{pmatrix}$$

(with $u_H = \sum_i \alpha_i \Phi_i$)

Institut für Numerische und Angewandte Mathematik

Corresponding matrix assembler requires computation of entries:

$$A_H(\Phi_i,\Phi_j) = \sum_{T\in\mathcal{T}_H} |T| \fint_{Y_T^\varepsilon} A^\varepsilon(\nabla R_T(\Phi_i))\cdot\nabla\Phi_j.$$

We obtain the following algebraic system:

$$\begin{pmatrix} A_H(\Phi_1,\Phi_1) & \dots & A_H(\Phi_1,\Phi_N) \\ \vdots & & \vdots \\ A_H(\Phi_N,\Phi_1) & \dots & A_H(\Phi_N,\Phi_N) \end{pmatrix} \begin{pmatrix} \alpha_1 \\ \vdots \\ \alpha_N \end{pmatrix} = \begin{pmatrix} (f,\Phi_1)_{L^2(\Omega)} \\ \vdots \\ (f,\Phi_N)_{L^2(\Omega)} \end{pmatrix}$$

(with $u_H = \sum_i \alpha_i \Phi_i$)

Goal: computation of

$$A_H(\Phi_i,\Phi_j) = \sum_{T\in\mathcal{T}_H} |T| \fint_{Y_T^\varepsilon} A^\varepsilon\left(\nabla R_T(\Phi_i)\right)\cdot\nabla\Phi_j.$$

Institut für Numerische
und Angewandte
Mathematik

Goal: computation of

$$A_H(\Phi_i,\Phi_j) = \sum_{T\in\mathcal{T}_H} |T| \fint_{Y_T^\varepsilon} A^\varepsilon(\nabla R_T(\Phi_i))\cdot\nabla\Phi_j.$$

Discrete cell-problem (for Φ_i): find $R_T(\Phi_i) \in \Phi_i + W_h(Y_T^\delta)$ with

$$\int_{Y_T^\delta} A^\varepsilon(\nabla R_T(\Phi_i))\cdot\nabla\phi_h = 0 \qquad \forall\phi_h \in W_h(Y_T^\delta).$$

Institut für Numerische und Angewandte Mathematik

Goal: computation of

$$A_H(\Phi_i,\Phi_j) = \sum_{T\in\mathcal{T}_H} |T| \fint_{Y_T^\varepsilon} A^\varepsilon(\nabla R_T(\Phi_i))\cdot\nabla\Phi_j.$$

Discrete cell-problem (for Φ_i): find $R_T(\Phi_i) \in \Phi_i + W_h(Y_T^\delta)$ with

$$\int_{Y_T^\delta} A^\varepsilon(\nabla R_T(\Phi_i))\cdot\nabla\phi_h = 0 \qquad \forall\phi_h \in W_h(Y_T^\delta).$$

Define fine-scale part $Q_T(\Phi_H)$ by:

$$R_T(\Phi_H)(x) = \Phi_H(x_T) + \delta Q_T(\Phi_H)\left(\frac{x-x_T}{\delta}\right).$$

Goal: computation of

$$A_H(\Phi_i,\Phi_j) = \sum_{T\in\mathcal{T}_H} |T| \fint_{Y_T^\varepsilon} A^\varepsilon(\nabla R_T(\Phi_i))\cdot\nabla\Phi_j .$$

Discrete cell-problem (for Φ_i): find $R_T(\Phi_i) \in \Phi_i + W_h(Y_T^\delta)$ with

$$\int_{Y_T^\delta} A^\varepsilon(\nabla R_T(\Phi_i))\cdot\nabla\phi_h = 0 \qquad \forall\phi_h \in W_h(Y_T^\delta).$$

Define fine-scale part $Q_T(\Phi_H)$ by:

$$R_T(\Phi_H)(x) = \Phi_H(x_T) + \delta Q_T(\Phi_H)\left(\frac{x-x_T}{\delta}\right).$$

Transform equation to 0-centered unit cube $Y = [-\frac{1}{2},\frac{1}{2}]^N$:

Goal: computation of

$$A_H(\Phi_i,\Phi_j) = \sum_{T\in\mathcal{T}_H} |T| \fint_{Y_T^\varepsilon} A^\varepsilon(\nabla R_T(\Phi_i))\cdot\nabla\Phi_j.$$

Discrete cell-problem (for Φ_i): find $R_T(\Phi_i) \in \Phi_i + W_h(Y_T^\delta)$ with

$$\int_{x_T+\delta Y} A^\varepsilon(\nabla R_T(\Phi_i))\cdot\nabla\phi_h = 0 \qquad \forall \phi_h \in W_h(Y_T^\delta).$$

Define fine-scale part $Q_T(\Phi_H)$ by:

$$R_T(\Phi_H)(x) = \Phi_H(x_T) + \delta Q_T(\Phi_H)\left(\frac{x-x_T}{\delta}\right).$$

Transform equation to 0-centered unit cube $Y = [-\frac{1}{2},\frac{1}{2}]^N$:

Goal: computation of

$$A_H(\Phi_i,\Phi_j) = \sum_{T\in\mathcal{T}_H} |T| \fint_{Y_T^\varepsilon} A^\varepsilon(\nabla R_T(\Phi_i))\cdot\nabla\Phi_j.$$

Discrete cell-problem (for Φ_i): find $R_T(\Phi_i) \in \Phi_i + W_h(Y_T^\delta)$ with

$$\int_{x_T+\delta Y} A^\varepsilon(\nabla R_T(\Phi_i))\cdot\nabla\phi_h = 0 \qquad \forall\phi_h \in W_h(Y_T^\delta).$$

Define fine-scale part $Q_T(\Phi_H)$ by:

$$R_T(\Phi_H)(x) = \Phi_H(x_T) + \delta Q_T(\Phi_H)\left(\frac{x-x_T}{\delta}\right).$$

Transform equation to 0-centered unit cube $Y = [-\frac{1}{2},\frac{1}{2}]^N$:
Find $Q_T(\Phi_H) \in W_h(Y)$ with

$$\int_Y A^\varepsilon(x_T+\delta y)\left(\nabla_x\Phi_H(x_T) + \nabla_y Q_T(\Phi_H)(y)\right)\cdot\nabla\phi_h(y)\,dy = 0$$

$$\forall\phi_h \in W_h(Y).$$

$$\int_Y A^{\varepsilon}(x_T + \delta y)\left(\nabla_x \Phi_H(x_T) + \nabla_y Q_T(\Phi_H)(y)\right) \cdot \nabla \phi_h(y)\, dy = 0 \quad \forall \phi_h \in W_h(Y).$$

Institut für Numerische und Angewandte Mathematik

$$\int_Y A^\varepsilon(x_T + \delta y)\left(\nabla_x \Phi_H(x_T) + \nabla_y Q_T(\Phi_H)(y)\right) \cdot \nabla \phi_h(y)\, dy = 0 \quad \forall \phi_h \in W_h(Y).$$

$\Rightarrow$ with $w_h := Q_T(\Phi_H)$ and $G := -A^\varepsilon(x_T + \delta\cdot)\nabla_x \Phi_H(x_T)$ we obtain a standard elliptic problem:

Institut für Numerische und Angewandte Mathematik

$$\int_Y A^{\varepsilon}(x_T + \delta y)\left(\nabla_x \Phi_H(x_T) + \nabla_y Q_T(\Phi_H)(y)\right) \cdot \nabla \phi_h(y)\, dy = 0 \quad \forall \phi_h \in W_h(Y).$$

$\Rightarrow$ with $w_h := Q_T(\Phi_H)$ and $G := -A^{\varepsilon}(x_T + \delta \cdot)\nabla_x \Phi_H(x_T)$ we obtain a standard elliptic problem:

$$\int_Y A \nabla w_h \cdot \nabla \phi_h = \int_Y G \cdot \nabla \phi_h \quad \forall \phi_h \in W_h(Y).$$

Institut für Numerische und Angewandte Mathematik

$$\int_Y A^\varepsilon(x_T + \delta y)\left(\nabla_x \Phi_H(x_T) + \nabla_y Q_T(\Phi_H)(y)\right) \cdot \nabla \phi_h(y)\, dy = 0$$
$$\forall \phi_h \in W_h(Y).$$

$\Rightarrow$ with $w_h := Q_T(\Phi_H)$ and $G := -A^\varepsilon(x_T + \delta \cdot)\nabla_x \Phi_H(x_T)$ we obtain a standard elliptic problem:

$$\int_Y A \nabla w_h \cdot \nabla \phi_h = \int_Y G \cdot \nabla \phi_h \quad \forall \phi_h \in W_h(Y).$$

Use standard tools of **dune-fem** to solve these cell problems!

$$\int_Y A^{\varepsilon}(x_T + \delta y)(\nabla_x \Phi_H(x_T) + \nabla_y Q_T(\Phi_H)(y)) \cdot \nabla \phi_h(y)\, dy = 0 \quad \forall \phi_h \in W_h(Y).$$

$\Rightarrow$ with $w_h := Q_T(\Phi_H)$ and $G := -A^{\varepsilon}(x_T + \delta \cdot)\nabla_x \Phi_H(x_T)$ we obtain a standard elliptic problem

$$\int_Y A(y)\nabla w_h(y) \cdot \nabla \phi_h(y)\, dy = \int_Y G(y) \cdot \nabla \phi_h(y) \quad \forall \phi_h \in W_h(Y).$$

Usage of **dune-fem** tools to solve these cell problems:

Institut für Numerische
und Angewandte
Mathematik

$$\int_Y A^\varepsilon(x_T+\delta y)\left(\nabla_x \Phi_H(x_T)+\nabla_y Q_T(\Phi_H)(y)\right)\cdot\nabla\phi_h(y)\,dy = 0 \quad \forall \phi_h \in W_h(Y).$$

$\Rightarrow$ with $w_h := Q_T(\Phi_H)$ and $G := -A^\varepsilon(x_T+\delta\cdot)\nabla_x \Phi_H(x_T)$ we obtain a standard elliptic problem

$$\int_Y A(y)\nabla w_h(y)\cdot\nabla\phi_h(y)\,dy = \int_Y G(y)\cdot\nabla\phi_h(y) \quad \forall \phi_h \in W_h(Y).$$

Usage of **dune-fem** tools to solve these cell problems:

```
using namespace Dune;
typedef FunctionSpace < double , double , WORLDDIM , 1 > Function_Space;
typedef LagrangeDiscreteFunctionSpace < Function_Space, Periodic_Grid_Part, POLORDER >
Discrete_Function_Space;
typedef AdaptiveDiscreteFunction < Discrete_Function_Space > Discrete_Function;
typedef SparseRowMatrixOperator< Discrete_Function, Discrete_Function, MatrixTraits > FEM_Matrix;
typedef OEMBICGSQOp< Discrete_Function, FEM_Matrix > Inverse_FEM_Matrix;
typedef DiffusionOperator< Function_Space > Diffusion;
```

Institut für Numerische
und Angewandte
Mathematik

$$\int_Y A^\varepsilon(x_T+\delta y)\left(\nabla_x \Phi_H(x_T)+\nabla_y Q_T(\Phi_H)(y)\right)\cdot\nabla\phi_h(y)\,dy=0\quad\forall\phi_h\in W_h(Y).$$

$\Rightarrow$ with $w_h := Q_T(\Phi_H)$ and $G := -A^\varepsilon(x_T+\delta\cdot)\nabla_x \Phi_H(x_T)$ we obtain a standard elliptic problem

$$\int_Y A(y)\nabla w_h(y)\cdot\nabla\phi_h(y)\,dy=\int_Y G(y)\cdot\nabla\phi_h(y)\quad\forall\phi_h\in W_h(Y).$$

Usage of **dune-fem** tools to solve these cell problems:

```
using namespace Dune;
typedef FunctionSpace < double , double , WORLDDIM , 1 > Function_Space;
typedef LagrangeDiscreteFunctionSpace < Function_Space, Periodic_Grid_Part, POLORDER >
Discrete_Function_Space;
typedef AdaptiveDiscreteFunction < Discrete_Function_Space > Discrete_Function;
typedef SparseRowMatrixOperator< Discrete_Function, Discrete_Function, MatrixTraits > FEM_Matrix;
typedef OEMBICGSQOp< Discrete_Function, FEM_Matrix > Inverse_FEM_Matrix;
typedef DiffusionOperator< Function_Space > Diffusion;
```

```
// methods of Diffusion class:

// get the diffusive flux:   A^ε(x,ξ) = flux
void diffusiveFlux ( const DomainType &x, const JacobianRangeType &xi, JacobianRangeType &flux ) const
{
  ...
}
```

Institut für Numerische und Angewandte Mathematik

$$\int_Y A^\varepsilon(x_T + \delta y)(\nabla_x \Phi_H(x_T) + \nabla_y Q_T(\Phi_H)(y)) \cdot \nabla\phi_h(y)\, dy = 0 \quad \forall \phi_h \in W_h(Y).$$

$\Rightarrow$ with $w_h := Q_T(\Phi_H)$ and $G := -A^\varepsilon(x_T + \delta\cdot)\nabla_x \Phi_H(x_T)$ we obtain a standard elliptic problem

$$\int_Y A(y)\nabla w_h(y) \cdot \nabla\phi_h(y)\, dy = \int_Y G(y) \cdot \nabla\phi_h(y) \quad \forall \phi_h \in W_h(Y).$$

Usage of **dune-fem** tools to solve these cell problems:

```
using namespace Dune;
typedef FunctionSpace < double , double , WORLDDIM , 1 > Function_Space;
typedef LagrangeDiscreteFunctionSpace < Function_Space, Periodic_Grid_Part, POLORDER >
Discrete_Function_Space;
typedef AdaptiveDiscreteFunction < Discrete_Function_Space > Discrete_Function;
typedef SparseRowMatrixOperator< Discrete_Function, Discrete_Function, MatrixTraits > FEM_Matrix;
typedef OEMBICGSQOp< Discrete_Function, FEM_Matrix > Inverse_FEM_Matrix;
typedef DiffusionOperator< Function_Space > Diffusion;
```

```
// methods of Diffusion class:

// get the diffusive flux:   A^ε(x,ξ) = flux
void diffusiveFlux ( const DomainType &x, const JacobianRangeType &xi, JacobianRangeType &flux ) const
{
  ...
}

// for nonlinear diffusion operator:
// get the derivative of the diffusive flux:   JA^ε(x,ξ)direction = flux
void jacobianDiffusiveFlux ( const DomainType &x, const JacobianRangeType &xi,
                             const JacobianRangeType &direction_gradient, JacobianRangeType &flux ) const
{
  ...
}
```

$$\int_Y A^{\varepsilon}(x_T + \delta y)\left(\nabla_x \Phi_H(x_T) + \nabla_y Q_T(\Phi_H)(y)\right) \cdot \nabla \phi_h(y) \, dy = 0 \quad \forall \phi_h \in W_h(Y).$$

$\Rightarrow$ with $w_h := Q_T(\Phi_H)$ and $G := -A^{\varepsilon}(x_T + \delta\cdot)\nabla_x \Phi_H(x_T)$ we obtain a standard elliptic problem

$$\int_Y A(y)\nabla w_h(y) \cdot \nabla \phi_h(y) \, dy = \int_Y G(y) \cdot \nabla \phi_h(y) \quad \forall \phi_h \in W_h(Y).$$

Usage of **dune-fem** tools to solve these cell problems:

```
using namespace Dune;
typedef FunctionSpace < double , double , WORLDDIM , 1 > Function_Space;
typedef LagrangeDiscreteFunctionSpace < Function_Space, Periodic_Grid_Part, POLORDER >
Discrete_Function_Space;
typedef AdaptiveDiscreteFunction < Discrete_Function_Space > Discrete_Function;
typedef SparseRowMatrixOperator< Discrete_Function, Discrete_Function, MatrixTraits > FEM_Matrix;
typedef OEMBICGSQOp< Discrete_Function, FEM_Matrix > Inverse_FEM_Matrix;
typedef DiffusionOperator< Function_Space > Diffusion;
// non-standard dune-fem class (describes how to assemble the stiffness matrix for our problem)
typedef DiscreteEllipticOperator< Discrete_Function, Diffusion, Reaction > Discrete_Elliptic_Operator;
//-----------------------------------------------------------------------------------------------
```

$$\int_Y A^{\varepsilon}(x_T + \delta y)\left(\nabla_x \Phi_H(x_T) + \nabla_y Q_T(\Phi_H)(y)\right) \cdot \nabla \phi_h(y)\, dy = 0 \quad \forall \phi_h \in W_h(Y).$$

$\Rightarrow$ with $w_h := Q_T(\Phi_H)$ and $G := -A^{\varepsilon}(x_T + \delta \cdot)\nabla_x \Phi_H(x_T)$ we obtain a standard elliptic problem

$$\int_Y A(y)\nabla w_h(y) \cdot \nabla \phi_h(y)\, dy = \int_Y G(y) \cdot \nabla \phi_h(y) \quad \forall \phi_h \in W_h(Y).$$

Usage of **dune-fem** tools to solve these cell problems:

```
using namespace Dune;
typedef FunctionSpace < double , double , WORLDDIM , 1 > Function_Space;
typedef LagrangeDiscreteFunctionSpace < Function_Space, Periodic_Grid_Part, POLORDER >
Discrete_Function_Space;
typedef AdaptiveDiscreteFunction < Discrete_Function_Space > Discrete_Function;
typedef SparseRowMatrixOperator< Discrete_Function, Discrete_Function, MatrixTraits > FEM_Matrix;
typedef OEMBICGSQOp< Discrete_Function, FEM_Matrix > Inverse_FEM_Matrix;
typedef DiffusionOperator< Function_Space > Diffusion;
// non-standard dune-fem class (describes how to assemble the stiffness matrix for our problem)
typedef DiscreteEllipticOperator< Discrete_Function, Diffusion, Reaction > Discrete_Elliptic_Operator;
//----------------------------------------------------------------------------------------------------
Discrete_Function_Space discreteFunctionSpace( periodicGridPart );
Diffusion A;
Discrete_Elliptic_Operator discreteEllipticOp( discreteFunctionSpace, A );
//----------------------------------------------------------------------------------------------------
Discrete_Function solution( "solution", discreteFunctionSpace );
Discrete_Function rhs( "right hand side ", discreteFunctionSpace );
FEM_Matrix stiffness_matrix( "FEM stiffness matrix", discreteFunctionSpace, discreteFunctionSpace );
Inverse_FEM_Matrix biCGStab( stiffness_matrix, 1e-8, 1e-8, 20000, VERBOSE );
//----------------------------------------------------------------------------------------------------
```

$$\int_Y A^{\varepsilon}(x_T + \delta y)\left(\nabla_x \Phi_H(x_T) + \nabla_y Q_T(\Phi_H)(y)\right) \cdot \nabla \phi_h(y)\, dy = 0 \quad \forall \phi_h \in W_h(Y).$$

$\Rightarrow$ with $w_h := Q_T(\Phi_H)$ and $G := -A^{\varepsilon}(x_T + \delta\cdot)\nabla_x \Phi_H(x_T)$ we obtain a standard elliptic problem

$$\int_Y A(y)\nabla w_h(y) \cdot \nabla \phi_h(y)\, dy = \int_Y G(y) \cdot \nabla \phi_h(y) \quad \forall \phi_h \in W_h(Y).$$

Usage of **dune-fem** tools to solve these cell problems:

```
using namespace Dune;
typedef FunctionSpace < double , double , WORLDDIM , 1 > Function_Space;
typedef LagrangeDiscreteFunctionSpace < Function_Space, Periodic_Grid_Part, POLORDER >
Discrete_Function_Space;
typedef AdaptiveDiscreteFunction < Discrete_Function_Space > Discrete_Function;
typedef SparseRowMatrixOperator< Discrete_Function, Discrete_Function, MatrixTraits > FEM_Matrix;
typedef OEMBICGSQOp< Discrete_Function, FEM_Matrix > Inverse_FEM_Matrix;
typedef DiffusionOperator< Function_Space > Diffusion;
// non-standard dune-fem class (describes how to assemble the stiffness matrix for our problem)
typedef DiscreteEllipticOperator< Discrete_Function, Diffusion, Reaction > Discrete_Elliptic_Operator;
//----------------------------------------------------------------------------------------
Discrete_Function_Space discreteFunctionSpace( periodicGridPart );
Diffusion A;
Discrete_Elliptic_Operator discreteEllipticOp( discreteFunctionSpace, A );
//----------------------------------------------------------------------------------------
Discrete_Function solution( "solution", discreteFunctionSpace );
Discrete_Function rhs( "right hand side ", discreteFunctionSpace );
FEM_Matrix stiffness_matrix( "FEM stiffness matrix", discreteFunctionSpace, discreteFunctionSpace );
Inverse_FEM_Matrix biCGStab( stiffness_matrix, 1e-8, 1e-8, 20000, VERBOSE );
//----------------------------------------------------------------------------------------
rhsassembler.assemble < QUADORDER >( G , rhs);
discreteEllipticOp.assemble_matrix( stiffness_matrix );
biCGStab( rhs, solution );
```

$$\int_Y A^\varepsilon(x_T + \delta y)\left(\nabla_x \Phi_H(x_T) + \nabla_y Q_T(\Phi_H)(y)\right) \cdot \nabla \phi_h(y)\, dy = 0 \quad \forall \phi_h \in W_h(Y).$$

$\Rightarrow$ with $w_h := Q_T(\Phi_H)$ and $G := -A^\varepsilon(x_T + \delta\cdot)\nabla_x \Phi_H(x_T)$ we obtain a standard elliptic problem

$$\int_Y A(y)\nabla w_h(y) \cdot \nabla \phi_h(y)\, dy = \int_Y G(y) \cdot \nabla \phi_h(y) \quad \forall \phi_h \in W_h(Y).$$

Usage of **dune-fem** tools to solve these cell problems:

```
using namespace Dune;
typedef FunctionSpace < double , double , WORLDDIM , 1 > Function_Space;
typedef LagrangeDiscreteFunctionSpace < Function_Space, Periodic_Grid_Part, POLORDER >
Discrete_Function_Space;
typedef AdaptiveDiscreteFunction < Discrete_Function_Space > Discrete_Function;
typedef SparseRowMatrixOperator< Discrete_Function, Discrete_Function, MatrixTraits > FEM_Matrix;
typedef OEMBICGSQOp< Discrete_Function, FEM_Matrix > Inverse_FEM_Matrix;
typedef DiffusionOperator< Function_Space > Diffusion;
// non-standard dune-fem class (describes how to assemble the stiffness matrix for our problem)
typedef DiscreteEllipticOperator< Discrete_Function, Diffusion, Reaction > Discrete_Elliptic_Operator;
//------------------------------------------------------------------------------------------------
Discrete_Function_Space discreteFunctionSpace( periodicGridPart );
Diffusion A;
Discrete_Elliptic_Operator discreteEllipticOp( discreteFunctionSpace, A );
//------------------------------------------------------------------------------------------------
Discrete_Function solution( "solution", discreteFunctionSpace );
Discrete_Function rhs( "right hand side ", discreteFunctionSpace );
FEM_Matrix stiffness_matrix( "FEM stiffness matrix", discreteFunctionSpace, discreteFunctionSpace );
Inverse_FEM_Matrix biCGStab( stiffness_matrix, 1e-8, 1e-8, 20000, VERBOSE );
//------------------------------------------------------------------------------------------------
rhsassembler.assemble < quadOrder >( G , rhs);
discreteEllipticOp.assemble_matrix( stiffness_matrix );
biCGStab( rhs, solution );
```

Institut für Numerische und Angewandte Mathematik

$$\int_Y A^\varepsilon(x_T+\delta y)(\nabla_x \Phi_H(x_T)+\nabla_y Q_T(\Phi_H)(y))\cdot\nabla\phi_h(y)\,dy=0 \quad \forall \phi_h\in W_h(Y).$$

$\Rightarrow$ with $w_h := Q_T(\Phi_H)$ and $G := -A^\varepsilon(x_T+\delta\cdot)\nabla_x \Phi_H(x_T)$ we obtain a standard elliptic problem

$$\int_Y A(y)\nabla w_h(y)\cdot\nabla\phi_h(y)\,dy = \int_Y G(y)\cdot\nabla\phi_h(y) \quad \forall \phi_h\in W_h(Y).$$

Usage of **dune-fem** tools to solve these cell problems:

```
using namespace Dune;
typedef FunctionSpace < double , double , WORLDDIM , 1 > Function_Space;
typedef LagrangeDiscreteFunctionSpace < Function_Space, Periodic_Grid_Part, POLORDER >
Discrete_Function_Space;
typedef AdaptiveDiscreteFunction < Discrete_Function_Space > Discrete_Function;
typedef SparseRowMatrixOperator< Discrete_Function, Discrete_Function, MatrixTraits > FEM_Matrix;
typedef OEMBICGSQOp< Discrete_Function, FEM_Matrix > Inverse_FEM_Matrix;
typedef DiffusionOperator< Function_Space > Diffusion;
// non-standard dune-fem class (describes how to assemble the stiffness matrix for our problem)
typedef DiscreteEllipticOperator< Discrete_Function, Diffusion, Reaction > Discrete_Elliptic_Operator;
//-----------------------------------------------------------------------------------------
Discrete_Function_Space discreteFunctionSpace( periodicGridPart );
Diffusion A;
Discrete_Elliptic_Operator discreteEllipticOp( discreteFunctionSpace, A );
//-----------------------------------------------------------------------------------------
Discrete_Function solution( "solution", discreteFunctionSpace );
Discrete_Function rhs( "right hand side ", discreteFunctionSpace );
FEM_Matrix stiffness_matrix( "FEM stiffness matrix", discreteFunctionSpace, discreteFunctionSpace );
Inverse_FEM_Matrix biCGStab( stiffness_matrix, 1e-8, 1e-8, 20000, VERBOSE );
//-----------------------------------------------------------------------------------------
rhsassembler.assemble < quadOrder >( G , rhs);
discreteEllipticOp.assemble_matrix( stiffness_matrix );
biCGStab( rhs, solution );
```

Write solutions of cell problems (discrete functions) to file for later usage!

Institut für Numerische und Angewandte Mathematik

Cell problems (saved in file):

$$\int_Y A^\varepsilon(x_T + \delta y)(\nabla_x \Phi_i(x_T) + \nabla_y Q_T(\Phi_i)(y)) \cdot \nabla \phi_h(y) \; dy = 0.$$

Algebraic macro-problem: find $\alpha \in \mathbb{R}^N$, $u_H = \sum_i \alpha_i \Phi_i$, with

$$\begin{pmatrix} A_H(\Phi_1,\Phi_1) & \dots & A_H(\Phi_1,\Phi_N) \\ \vdots & & \vdots \\ A_H(\Phi_N,\Phi_1) & \dots & A_H(\Phi_N,\Phi_N) \end{pmatrix} \begin{pmatrix} \alpha_1 \\ \vdots \\ \alpha_N \end{pmatrix} = \begin{pmatrix} (f,\Phi_1)_{L^2(\Omega)} \\ \vdots \\ (f,\Phi_N)_{L^2(\Omega)} \end{pmatrix},$$

where

$$A_H(\Phi_i,\Phi_j) = \sum_{T \in \mathcal{T}_H} |T| \fint_{Y_T^\varepsilon} A^\varepsilon (\nabla R_T(\Phi_i)) \cdot \nabla \Phi_j .$$

Cell problems (saved in file):

$$\int_Y A^{\varepsilon}(x_T + \delta y)(\nabla_x \Phi_i(x_T) + \nabla_y Q_T(\Phi_i)(y)) \cdot \nabla \phi_h(y) \; dy = 0.$$

Algebraic macro-problem: find $\alpha \in \mathbb{R}^N$, $u_H = \sum_i \alpha_i \Phi_i$, with

$$\begin{pmatrix} A_H(\Phi_1,\Phi_1) & \dots & A_H(\Phi_1,\Phi_N) \\ \vdots & & \vdots \\ A_H(\Phi_N,\Phi_1) & \dots & A_H(\Phi_N,\Phi_N) \end{pmatrix} \begin{pmatrix} \alpha_1 \\ \vdots \\ \alpha_N \end{pmatrix} = \begin{pmatrix} (f,\Phi_1)_{L^2(\Omega)} \\ \vdots \\ (f,\Phi_N)_{L^2(\Omega)} \end{pmatrix},$$

where

$$A_H(\Phi_i,\Phi_j) = \sum_{T \in \mathcal{T}_H} |T| \fint_{\frac{\varepsilon}{\delta} Y} A^{\varepsilon}(x_T + \delta y)(\nabla_x \Phi_i(x_T) + \nabla_y Q_T(\Phi_i)(y)) \cdot \nabla \Phi_j(x_T) \; dy.$$

Institut für Numerische und Angewandte Mathematik

Cell problems (saved in file):
$$\int_Y A^\varepsilon(x_T + \delta y)(\nabla_x \Phi_i(x_T) + \nabla_y Q_T(\Phi_i)(y)) \cdot \nabla \phi_h(y)\ dy = 0.$$

Algebraic macro-problem: find $\alpha \in \mathbb{R}^N$, $u_H = \sum_i \alpha_i \Phi_i$, with

$$\begin{pmatrix} A_H(\Phi_1,\Phi_1) & \dots & A_H(\Phi_1,\Phi_N) \\ \vdots & & \vdots \\ A_H(\Phi_N,\Phi_1) & \dots & A_H(\Phi_N,\Phi_N) \end{pmatrix} \begin{pmatrix} \alpha_1 \\ \vdots \\ \alpha_N \end{pmatrix} = \begin{pmatrix} (f,\Phi_1)_{L^2(\Omega)} \\ \vdots \\ (f,\Phi_N)_{L^2(\Omega)} \end{pmatrix},$$

where

$$A_H(\Phi_i,\Phi_j) = \sum_{T \in \mathcal{T}_H} |T| \fint_{\frac{\varepsilon}{\delta}Y} A^\varepsilon(x_T + \delta y)(\nabla_x \Phi_i(x_T) + \nabla_y Q_T(\Phi_i)(y)) \cdot \nabla \Phi_j(x_T)\ dy.$$

$Q_T(\Phi_i)$ were computed in pre-process

Cell problems (saved in file):
$$\int_Y A^{\varepsilon}(x_T + \delta y)(\nabla_x \Phi_i(x_T) + \nabla_y Q_T(\Phi_i)(y)) \cdot \nabla \phi_h(y)\ dy = 0.$$

Algebraic macro-problem: find $\alpha \in \mathbb{R}^N$, $u_H = \sum_i \alpha_i \Phi_i$, with

$$\begin{pmatrix} A_H(\Phi_1,\Phi_1) & \dots & A_H(\Phi_1,\Phi_N) \\ \vdots & & \vdots \\ A_H(\Phi_N,\Phi_1) & \dots & A_H(\Phi_N,\Phi_N) \end{pmatrix} \begin{pmatrix} \alpha_1 \\ \vdots \\ \alpha_N \end{pmatrix} = \begin{pmatrix} (f,\Phi_1)_{L^2(\Omega)} \\ \vdots \\ (f,\Phi_N)_{L^2(\Omega)} \end{pmatrix},$$

where

$$A_H(\Phi_i,\Phi_j) = \sum_{T \in \mathcal{T}_H} |T| \fint_{\frac{\varepsilon}{\delta}Y} A^{\varepsilon}(x_T + \delta y)(\nabla_x \Phi_i(x_T) + \nabla_y Q_T(\Phi_i)(y)) \cdot \nabla \Phi_j(x_T)\ dy.$$

$Q_T(\Phi_i)$ were computed in pre-process $\Rightarrow$ algebraic macro-problem available.

Institut für Numerische und Angewandte Mathematik

Cell problems (saved in file):
$$\int_Y A^\varepsilon(x_T+\delta y)(\nabla_x \Phi_i(x_T) + \nabla_y Q_T(\Phi_i)(y)) \cdot \nabla\phi_h(y)\, dy = 0.$$

Algebraic macro-problem: find $\alpha \in \mathbb{R}^N$, $u_H = \sum_i \alpha_i \Phi_i$, with

$$\begin{pmatrix} A_H(\Phi_1,\Phi_1) & \dots & A_H(\Phi_1,\Phi_N) \\ \vdots & & \vdots \\ A_H(\Phi_N,\Phi_1) & \dots & A_H(\Phi_N,\Phi_N) \end{pmatrix} \begin{pmatrix} \alpha_1 \\ \vdots \\ \alpha_N \end{pmatrix} = \begin{pmatrix} (f,\Phi_1)_{L^2(\Omega)} \\ \vdots \\ (f,\Phi_N)_{L^2(\Omega)} \end{pmatrix},$$

where

$$A_H(\Phi_i,\Phi_j) = \sum_{T\in\mathcal{T}_H} |T| \fint_{\frac{\varepsilon}{\delta}Y} A^\varepsilon(x_T+\delta y)(\nabla_x \Phi_i(x_T) + \nabla_y Q_T(\Phi_i)(y)) \cdot \nabla\Phi_j(x_T)\, dy.$$

$Q_T(\Phi_i)$ were computed in pre-process $\Rightarrow$ algebraic macro-problem available.
$\Rightarrow$ Again, usage of standard **dune-fem** tools to solve this problem!

Institut für Numerische und Angewandte Mathematik

Cell problems (saved in file):

$$\int_Y A^\varepsilon(x_T + \delta y)(\nabla_x \Phi_i(x_T) + \nabla_y Q_T(\Phi_i)(y)) \cdot \nabla \phi_h(y) \; dy = 0.$$

Algebraic macro-problem: find $\alpha \in \mathbb{R}^N$, $u_H = \sum_i \alpha_i \Phi_i$, with

$$\begin{pmatrix} A_H(\Phi_1,\Phi_1) & \dots & A_H(\Phi_1,\Phi_N) \\ \vdots & & \vdots \\ A_H(\Phi_N,\Phi_1) & \dots & A_H(\Phi_N,\Phi_N) \end{pmatrix} \begin{pmatrix} \alpha_1 \\ \vdots \\ \alpha_N \end{pmatrix} = \begin{pmatrix} (f,\Phi_1)_{L^2(\Omega)} \\ \vdots \\ (f,\Phi_N)_{L^2(\Omega)} \end{pmatrix},$$

where

$$A_H(\Phi_i,\Phi_j) = \sum_{T \in \mathcal{T}_H} |T| \fint_{\frac{\varepsilon}{\delta}Y} A^\varepsilon(x_T + \delta y)(\nabla_x \Phi_i(x_T) + \nabla_y Q_T(\Phi_i)(y)) \cdot \nabla \Phi_j(x_T) \; dy.$$

$Q_T(\Phi_i)$ were computed in pre-process $\Rightarrow$ algebraic macro-problem available.
$\Rightarrow$ Again, usage of standard **dune-fem** tools to solve this problem!

```
using namespace Dune;
typedef FunctionSpace < double , double , WORLDDIM , 1 > Function_Space;
typedef LagrangeDiscreteFunctionSpace < Function_Space, Grid_Part, POLORDER > Discrete_Function_Space;
typedef AdaptiveDiscreteFunction < Discrete_Function_Space > Discrete_Function;
typedef SparseRowMatrixOperator < Discrete_Function, Discrete_Function, MatrixTraits > FEM_Matrix;
typedef OEMBICGSQOp < Discrete_Function, FEM_Matrix > Inverse_FEM_Matrix;
typedef DiffusionOperator < Function_Space > Diffusion;
```

Cell problems (saved in file):
$$\int_Y A^\varepsilon(x_T+\delta y)(\nabla_x \Phi_i(x_T) + \nabla_y Q_T(\Phi_i)(y)) \cdot \nabla\phi_h(y)\, dy = 0.$$

Algebraic macro-problem: find $\alpha \in \mathbb{R}^N$, $u_H = \sum_i \alpha_i \Phi_i$, with

$$\begin{pmatrix} A_H(\Phi_1,\Phi_1) & \dots & A_H(\Phi_1,\Phi_N) \\ \vdots & & \vdots \\ A_H(\Phi_N,\Phi_1) & \dots & A_H(\Phi_N,\Phi_N) \end{pmatrix} \begin{pmatrix} \alpha_1 \\ \vdots \\ \alpha_N \end{pmatrix} = \begin{pmatrix} (f,\Phi_1)_{L^2(\Omega)} \\ \vdots \\ (f,\Phi_N)_{L^2(\Omega)} \end{pmatrix},$$

where

$$A_H(\Phi_i,\Phi_j) = \sum_{T\in\mathcal{T}_H} |T| \fint_{\frac{\varepsilon}{\delta}Y} A^\varepsilon(x_T+\delta y)(\nabla_x \Phi_i(x_T) + \nabla_y Q_T(\Phi_i)(y)) \cdot \nabla\Phi_j(x_T)\, dy.$$

$Q_T(\Phi_i)$ were computed in pre-process $\Rightarrow$ algebraic macro-problem available.
$\Rightarrow$ Again, usage of standard **dune-fem** tools to solve this problem! Only slight differences.

```
using namespace Dune;
typedef FunctionSpace < double , double , WORLDDIM , 1 > Function_Space;
typedef LagrangeDiscreteFunctionSpace < Function_Space, Grid_Part, POLORDER > Discrete_Function_Space;
typedef AdaptiveDiscreteFunction < Discrete_Function_Space > Discrete_Function;
typedef SparseRowMatrixOperator < Discrete_Function, Discrete_Function, MatrixTraits > FEM_Matrix;
typedef OEMBICGSQOp < Discrete_Function, FEM_Matrix > Inverse_FEM_Matrix;
typedef DiffusionOperator < Function_Space > Diffusion;
```

Cell problems (saved in file):

$$\int_Y A^\varepsilon(x_T + \delta y)(\nabla_x \Phi_i(x_T) + \nabla_y Q_T(\Phi_i)(y)) \cdot \nabla \phi_h(y) \; dy = 0.$$

Algebraic macro-problem: find $\alpha \in \mathbb{R}^N$, $u_H = \sum_i \alpha_i \Phi_i$, with

$$\begin{pmatrix} A_H(\Phi_1, \Phi_1) & \dots & A_H(\Phi_1, \Phi_N) \\ \vdots & & \vdots \\ A_H(\Phi_N, \Phi_1) & \dots & A_H(\Phi_N, \Phi_N) \end{pmatrix} \begin{pmatrix} \alpha_1 \\ \vdots \\ \alpha_N \end{pmatrix} = \begin{pmatrix} (f, \Phi_1)_{L^2(\Omega)} \\ \vdots \\ (f, \Phi_N)_{L^2(\Omega)} \end{pmatrix},$$

where

$$A_H(\Phi_i, \Phi_j) = \sum_{T \in \mathcal{T}_H} |T| \fint_{\frac{\varepsilon}{\delta} Y} A^\varepsilon(x_T + \delta y)(\nabla_x \Phi_i(x_T) + \nabla_y Q_T(\Phi_i)(y)) \cdot \nabla \Phi_j(x_T) \, dy.$$

$Q_T(\Phi_i)$ were computed in pre-process $\Rightarrow$ algebraic macro-problem available.

$\Rightarrow$ Again, usage of standard **dune-fem** tools to solve this problem! Only slight differences.

```
using namespace Dune;
typedef FunctionSpace < double , double , WORLDDIM , 1 > Function_Space;
typedef LagrangeDiscreteFunctionSpace < Function_Space, Grid_Part, POLORDER > Discrete_Function_Space;
typedef AdaptiveDiscreteFunction < Discrete_Function_Space > Discrete_Function;
typedef SparseRowMatrixOperator < Discrete_Function, Discrete_Function, MatrixTraits > FEM_Matrix;
typedef OEMBICGSQOp < Discrete_Function, FEM_Matrix > Inverse_FEM_Matrix;
typedef DiffusionOperator < Function_Space > Diffusion;

// non-standard dune-fem class (describes how to assemble the stiffness matrix for our problem)
typedef EllipticHMMOperator < Discrete_Function, Diffusion > Elliptic_HMM_Operator;
```

Cell problems (saved in file):

$$A_H(\Phi_i,\Phi_j) = \sum_{T\in\mathcal{T}_H} |T| \fint_{\frac{\varepsilon}{\delta}Y} A^{\varepsilon}(x_T+\delta y)\left(\nabla_x\Phi_i(x_T) + \nabla_y Q_T(\Phi_i)(y)\right)\cdot\nabla_x\Phi_j(x_T)\,dy.$$

Algebraic macro-problem: find $\alpha\in\mathbb{R}^N$, $u_H = \sum_i \alpha_i\,\Phi_i$, with

$$\begin{pmatrix} A_H(\Phi_1,\Phi_1) & \dots & A_H(\Phi_1,\Phi_N) \\ \vdots & & \vdots \\ A_H(\Phi_N,\Phi_1) & \dots & A_H(\Phi_N,\Phi_N) \end{pmatrix} \begin{pmatrix} \alpha_1 \\ \vdots \\ \alpha_N \end{pmatrix} = \begin{pmatrix} (f,\Phi_1)_{L^2(\Omega)} \\ \vdots \\ (f,\Phi_N)_{L^2(\Omega)} \end{pmatrix}.$$

Institut für Numerische
und Angewandte
Mathematik

Cell problems (saved in file):

$$A_H(\Phi_i,\Phi_j) = \sum_{T\in\mathcal{T}_H} |T| \fint_{\frac{\varepsilon}{\delta}Y} A^\varepsilon(x_T+\delta y)\left(\nabla_x \Phi_i(x_T) + \nabla_y Q_T(\Phi_i)(y)\right)\cdot \nabla_x \Phi_j(x_T)\, dy.$$

Algebraic macro-problem: find $\alpha \in \mathbb{R}^N$, $u_H = \sum_i \alpha_i \Phi_i$, with

$$\begin{pmatrix} A_H(\Phi_1,\Phi_1) & \dots & A_H(\Phi_1,\Phi_N) \\ \vdots & & \vdots \\ A_H(\Phi_N,\Phi_1) & \dots & A_H(\Phi_N,\Phi_N) \end{pmatrix} \begin{pmatrix} \alpha_1 \\ \vdots \\ \alpha_N \end{pmatrix} = \begin{pmatrix} (f,\Phi_1)_{L^2(\Omega)} \\ \vdots \\ (f,\Phi_N)_{L^2(\Omega)} \end{pmatrix}.$$

```
using namespace Dune;
typedef FunctionSpace < double , double , WORLDDIM , 1 > FunctionSpace;
typedef LagrangeDiscreteFunctionSpace < FunctionSpace, GridPart, POLORDER > DiscreteFunctionSpace;
typedef AdaptiveDiscreteFunction < DiscreteFunctionSpace > DiscreteFunction;
typedef SparseRowMatrixOperator < DiscreteFunction, DiscreteFunction, MatrixTraits > FEMMatrix;
typedef OEMBICGSQOp < DiscreteFunction, FEMMatrix > InverseFEMMatrix;
typedef DiffusionOperator < FunctionSpace > Diffusion;

// non-standard dune-fem class (describes how to assemble the stiffness matrix for our problem)
typedef EllipticHMMOperator < DiscreteFunction, Diffusion > EllipticHMMOperator;
//------------------------------------------------------------------------------------------------
```

Cell problems (saved in file):

$$A_H(\Phi_i,\Phi_j) = \sum_{T\in\mathcal{T}_H} |T| \fint_{\frac{\varepsilon}{\delta}Y} A^{\varepsilon}(x_T+\delta y)(\nabla_x\Phi_i(x_T)+\nabla_y Q_T(\Phi_i)(y))\cdot\nabla_x\Phi_j(x_T)\,dy.$$

Algebraic macro-problem: find $\alpha \in \mathbb{R}^N$, $u_H = \sum_i \alpha_i \Phi_i$, with

$$\begin{pmatrix} A_H(\Phi_1,\Phi_1) & \dots & A_H(\Phi_1,\Phi_N) \\ \vdots & & \vdots \\ A_H(\Phi_N,\Phi_1) & \dots & A_H(\Phi_N,\Phi_N) \end{pmatrix} \begin{pmatrix} \alpha_1 \\ \vdots \\ \alpha_N \end{pmatrix} = \begin{pmatrix} (f,\Phi_1)_{L^2(\Omega)} \\ \vdots \\ (f,\Phi_N)_{L^2(\Omega)} \end{pmatrix}.$$

```
using namespace Dune;
typedef FunctionSpace < double , double , WORLDDIM , 1 > FunctionSpace;
typedef LagrangeDiscreteFunctionSpace < FunctionSpace, GridPart, POLORDER > DiscreteFunctionSpace;
typedef AdaptiveDiscreteFunction < DiscreteFunctionSpace > DiscreteFunction;
typedef SparseRowMatrixOperator < DiscreteFunction, DiscreteFunction, MatrixTraits > FEMMatrix;
typedef OEMBICGSQOp < DiscreteFunction, FEMMatrix > InverseFEMMatrix;
typedef DiffusionOperator < FunctionSpace > Diffusion;

// non-standard dune-fem class (describes how to assemble the stiffness matrix for our problem)
typedef EllipticHMMOperator < DiscreteFunction, Diffusion > EllipticHMMOperator;
//------------------------------------------------------------------------------------------
DiscreteFunctionSpace discreteFunctionSpace( gridPart );
```

Institut für Numerische und Angewandte Mathematik

Cell problems (saved in file):

$$A_H(\Phi_i,\Phi_j) = \sum_{T\in\mathcal{T}_H} |T| \fint_{\frac{\varepsilon}{\delta}Y} A^{\varepsilon}(x_T+\delta y)\left(\nabla_x\Phi_i(x_T)+\nabla_y Q_T(\Phi_i)(y)\right)\cdot\nabla_x\Phi_j(x_T)\ dy.$$

Algebraic macro-problem: find $\alpha \in \mathbb{R}^N$, $u_H = \sum_i \alpha_i \Phi_i$, with

$$\begin{pmatrix} A_H(\Phi_1,\Phi_1) & \dots & A_H(\Phi_1,\Phi_N) \\ \vdots & & \vdots \\ A_H(\Phi_N,\Phi_1) & \dots & A_H(\Phi_N,\Phi_N) \end{pmatrix} \begin{pmatrix} \alpha_1 \\ \vdots \\ \alpha_N \end{pmatrix} = \begin{pmatrix} (f,\Phi_1)_{L^2(\Omega)} \\ \vdots \\ (f,\Phi_N)_{L^2(\Omega)} \end{pmatrix}.$$

```
using namespace Dune;
typedef FunctionSpace < double , double , WORLDDIM , 1 > FunctionSpace;
typedef LagrangeDiscreteFunctionSpace < FunctionSpace, GridPart, POLORDER > DiscreteFunctionSpace;
typedef AdaptiveDiscreteFunction < DiscreteFunctionSpace > DiscreteFunction;
typedef SparseRowMatrixOperator < DiscreteFunction, DiscreteFunction, MatrixTraits > FEMMatrix;
typedef OEMBICGSQOp < DiscreteFunction, FEMMatrix > InverseFEMMatrix;
typedef DiffusionOperator < FunctionSpace > Diffusion;

// non-standard dune-fem class (describes how to assemble the stiffness matrix for our problem)
typedef EllipticHMMOperator < DiscreteFunction, Diffusion > EllipticHMMOperator;
//----------------------------------------------------------------------------------------------
DiscreteFunctionSpace discreteFunctionSpace( gridPart );
Diffusion A;
```

Institut für Numerische und Angewandte Mathematik

Cell problems (saved in file):

$$A_H(\Phi_i,\Phi_j) = \sum_{T\in\mathcal{T}_H} |T| \fint_{\frac{\varepsilon}{\delta}Y} A^\varepsilon(x_T+\delta y)(\nabla_x \Phi_i(x_T) + \nabla_y Q_T(\Phi_i)(y)) \cdot \nabla_x \Phi_j(x_T)\, dy.$$

Algebraic macro-problem: find $\alpha \in \mathbb{R}^N$, $u_H = \sum_i \alpha_i \Phi_i$, with

$$\begin{pmatrix} A_H(\Phi_1,\Phi_1) & \dots & A_H(\Phi_1,\Phi_N) \\ \vdots & & \vdots \\ A_H(\Phi_N,\Phi_1) & \dots & A_H(\Phi_N,\Phi_N) \end{pmatrix} \begin{pmatrix} \alpha_1 \\ \vdots \\ \alpha_N \end{pmatrix} = \begin{pmatrix} (f,\Phi_1)_{L^2(\Omega)} \\ \vdots \\ (f,\Phi_N)_{L^2(\Omega)} \end{pmatrix}.$$

```
using namespace Dune;
typedef FunctionSpace < double , double , WORLDDIM , 1 > FunctionSpace;
typedef LagrangeDiscreteFunctionSpace < FunctionSpace, GridPart, POLORDER > DiscreteFunctionSpace;
typedef AdaptiveDiscreteFunction < DiscreteFunctionSpace > DiscreteFunction;
typedef SparseRowMatrixOperator < DiscreteFunction, DiscreteFunction, MatrixTraits > FEMMatrix;
typedef OEMBICGSQOp < DiscreteFunction, FEMMatrix > InverseFEMMatrix;
typedef DiffusionOperator < FunctionSpace > Diffusion;

// non-standard dune-fem class (describes how to assemble the stiffness matrix for our problem)
typedef EllipticHMMOperator < DiscreteFunction, Diffusion > EllipticHMMOperator;
//--------------------------------------------------------------------------------------------
DiscreteFunctionSpace discreteFunctionSpace( gridPart );
Diffusion A;
EllipticHMMOperator ellipticHMMOperator( discreteFunctionSpace, A );
```

Institut für Numerische
und Angewandte
Mathematik

Cell problems (saved in file):

$$A_H(\Phi_i,\Phi_j) = \sum_{T\in\mathcal{T}_H} |T| \fint_{\frac{\varepsilon}{\delta}Y} A^\varepsilon(x_T+\delta y)\left(\nabla_x\Phi_i(x_T)+\nabla_y Q_T(\Phi_i)(y)\right)\cdot\nabla_x\Phi_j(x_T)\, dy.$$

Algebraic macro-problem: find $\alpha\in\mathbb{R}^N$, $u_H=\sum_i \alpha_i\,\Phi_i$, with

$$\begin{pmatrix} A_H(\Phi_1,\Phi_1) & \dots & A_H(\Phi_1,\Phi_N) \\ \vdots & & \vdots \\ A_H(\Phi_N,\Phi_1) & \dots & A_H(\Phi_N,\Phi_N)\end{pmatrix}\begin{pmatrix}\alpha_1\\ \vdots \\ \alpha_N\end{pmatrix} = \begin{pmatrix}(f,\Phi_1)_{L^2(\Omega)} \\ \vdots \\ (f,\Phi_N)_{L^2(\Omega)}\end{pmatrix}.$$

```
using namespace Dune;
typedef FunctionSpace < double , double , WORLDDIM , 1 > FunctionSpace;
typedef LagrangeDiscreteFunctionSpace < FunctionSpace, GridPart, POLORDER > DiscreteFunctionSpace;
typedef AdaptiveDiscreteFunction < DiscreteFunctionSpace > DiscreteFunction;
typedef SparseRowMatrixOperator < DiscreteFunction, DiscreteFunction, MatrixTraits > FEM_Matrix;
typedef OEMBICGSQOp < DiscreteFunction, FEM_Matrix > InverseFEM_Matrix;
typedef DiffusionOperator < FunctionSpace > Diffusion;

// non-standard dune-fem class (describes how to assemble the stiffness matrix for our problem)
typedef EllipticHMMOperator < DiscreteFunction, Diffusion > Elliptic_HMM_Operator;
//------------------------------------------------------------------------------------------
DiscreteFunctionSpace discreteFunctionSpace( gridPart );
Diffusion A;
Elliptic_HMM_Operator ellipticHMMOperator( discreteFunctionSpace, A );
//------------------------------------------------------------------------------------------
DiscreteFunction solution( "solution", discreteFunctionSpace );
DiscreteFunction rhs( "right hand side ", discreteFunctionSpace );
FEM_Matrix stiffness_matrix( "FEM stiffness matrix", discreteFunctionSpace, discreteFunctionSpace );
InverseFEM_Matrix bicgstab( stiffness_matrix, 1e-8, 1e-8, 20000, VERBOSE );
//------------------------------------------------------------------------------------------
rhsassembler.assemble< QUADORDER >( f , rhs);
ellipticHMMOperator.assemble_matrix( stiffness_matrix );
bicgstab( rhs, solution );
```

Numerical examples

Numerical Experiment - Nonlinear elliptic problem:

Model Problem 1 ($\varepsilon = 10^{-7}$)

$$-\nabla \cdot A^{\varepsilon}(\cdot, \nabla u_{\varepsilon}) = 1 \text{ in } [0,2]^2,$$
$$u^{\varepsilon} = 0 \text{ on } \partial[0,2]^2.$$

A^{ε} is given by:

$$A^{\varepsilon}(x,\xi) = \begin{pmatrix} (0.1 + \cos(2\pi \frac{x_1}{\varepsilon})^2) \cdot (\xi_1 + \frac{1}{3}\xi_1^3) \\ (0.101 + (0.1\sin(2\pi \frac{x_2}{\varepsilon}))) \cdot (\xi_2 + \frac{1}{3}\xi_2^3) \end{pmatrix}$$

Institut für Numerische und Angewandte Mathematik

H	h	$\|u_{HMM} - u_{\varepsilon}\|_{L^2(\Omega)}$
2^{-1}	2^{-2}	$2.62 \cdot 10^{-1}$
2^{-2}	2^{-3}	$8.06 \cdot 10^{-2}$
2^{-3}	2^{-3}	$5.62 \cdot 10^{-2}$
2^{-3}	2^{-4}	$2.34 \cdot 10^{-2}$
2^{-4}	2^{-5}	$5.47 \cdot 10^{-3}$
2^{-4}	2^{-6}	$2.55 \cdot 10^{-3}$
2^{-5}	2^{-6}	$9.91 \cdot 10^{-4}$

Table: *Error table. H = macro grid size, δh = micro grid size.*

$(H,h) \rightarrow (\frac{H}{2}, \frac{h}{2})$	$\mathrm{EOC}(e^N)$
$(2^{-1}, 2^{-2}) \rightarrow (2^{-2}, 2^{-3})$	1.7018
$(2^{-2}, 2^{-3}) \rightarrow (2^{-3}, 2^{-4})$	1.7864
$(2^{-3}, 2^{-4}) \rightarrow (2^{-4}, 2^{-5})$	2.0941
$(2^{-4}, 2^{-5}) \rightarrow (2^{-5}, 2^{-6})$	2.4645

Table: *EOC table.*

Institut für Numerische und Angewandte Mathematik

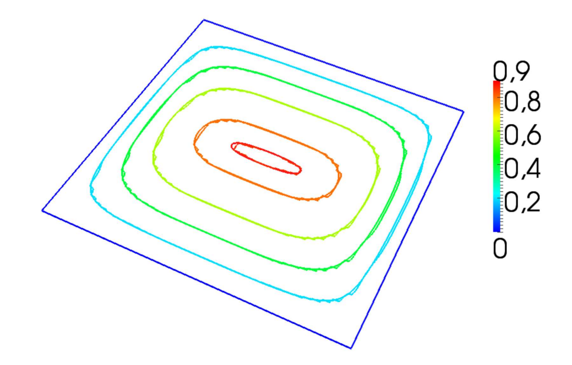


Figure: *Comparison of isolines: HMM approximation for* $(H,h) = (2^{-5}, 2^{-6})$ *and solution of highly resolved fine scale computation for* $\varepsilon = 0.05$.

Numerical Experiment - Advection diffusion, Adaptive algorithms:

Model Problem 2 (non-periodic setting; $\varepsilon = 10^{-2}$)

$$\partial_t u_\varepsilon - \nabla \cdot \varepsilon (A^\varepsilon(x)\nabla u_\varepsilon) + \frac{3}{2} b^\varepsilon(x)\nabla u_\varepsilon = 0 \text{ in } [0,\frac{1}{2}] \times \mathbb{R}^2,$$
$$u^\varepsilon(0,\cdot) = v_0 \text{ in } \mathbb{R}^2.$$

b is given by:

$$b^\varepsilon(x) := \begin{pmatrix} \sin(2\pi\frac{x_1}{\varepsilon})\sin(2\pi\frac{x_2}{\varepsilon}) + 3 \\ \cos(2\pi\frac{x_1}{\varepsilon})\cos(2\pi\frac{x_2}{\varepsilon}) - \frac{9}{2} \end{pmatrix}.$$

A is given by:

$$A^\varepsilon(x) := \begin{pmatrix} \log\left(4 + 2\sin^2(2\pi\frac{\log(|x_1|+1)}{\varepsilon})\right) & 0 \\ 0 & \log\left(4 + 2\cos^2(2\pi\frac{\log(|x_2|+1)}{\varepsilon})\right) \end{pmatrix}.$$

Table: $\|e_u^N\|_{L^2(\mathbb{R}^2)}^{rel}$ *denotes the error and* $\eta_H^{N,rel}$ *the estimated error. The numbers in the first line refer to different uniform computations.*

Uniform comp.	H	h	$\|e_u^N\|_{L^2(\mathbb{R}^2)}^{rel}$	$\eta_H^{N,rel}$
0	2^{-3}	2^{-4}	0.27411	0.91614
1	$2^{-\frac{7}{2}}$	2^{-4}	0.08612	0.64487
2	2^{-4}	2^{-4}	0.05191	0.32194
3	$2^{-\frac{9}{2}}$	2^{-4}	0.02726	0.18859
4	2^{-5}	2^{-4}	0.01245	0.09268

Table: $\|e_a^N\|^{rel}_{L^2(\mathbb{R}^2)}$ *denotes the error and* $\eta_H^{N,rel}$ *the estimated error. The numbers in the first line refer to different adaptive strategies, the numbers in line 4 to different uniform computations (Table 3).*

n_a	$\|e_a^N\|^{rel}_{L^2(\mathbb{R}^2)}$	$\eta_H^{N,rel}$	n_u	ϑ_e	ϑ_t
1	0.04518	0.21635	2	12.96%	19.38%
2	0.02724	0.13866	3	0.07%	42.4%
3	0.0243	0.13932	3	10.86%	38.59%
4	0.02298	0.13679	3	15.7%	32.87%
5	0.0123	0.08064	4	1.2%	25.58%
6	0.01081	0.07942	4	13.17%	18.32%

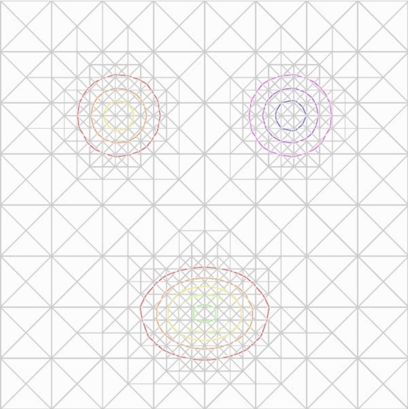


Figure: *HMM approximation at $t = 0.5$, determined by adaptive computation.*

Thank you for your attention!

Institut für Numerische und Angewandte Mathematik