Running computations with the Multi-scale Finite Element approach (MsFEM) in FreeFem++

C. Le Bris [1], F. Legoll [1], P. L. Rothé [1]

[1]Ecole des Ponts, Champs sur Marne, France - Inria Matherials

e-mail: {claude.le-bris,frederic.legoll,pierre-loik.rothe}@enpc.fr

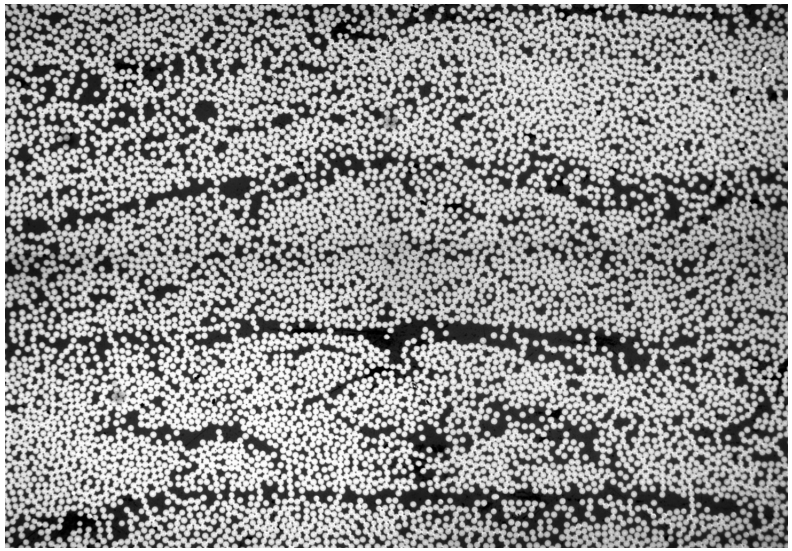
13th December 2018

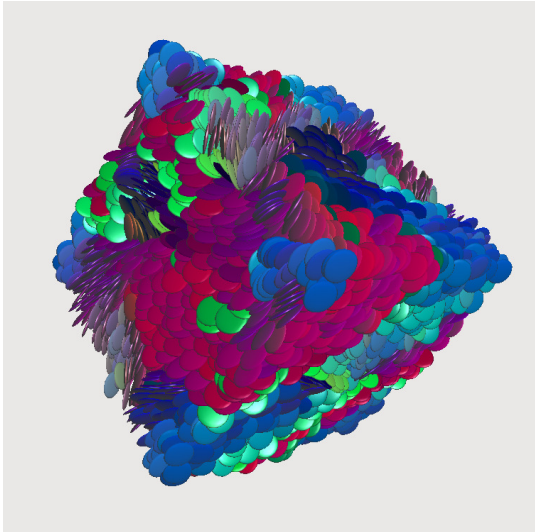
10th FreeFem++ days

Outline

Motivation

Highly heterogeneous materials are widely used in the industry





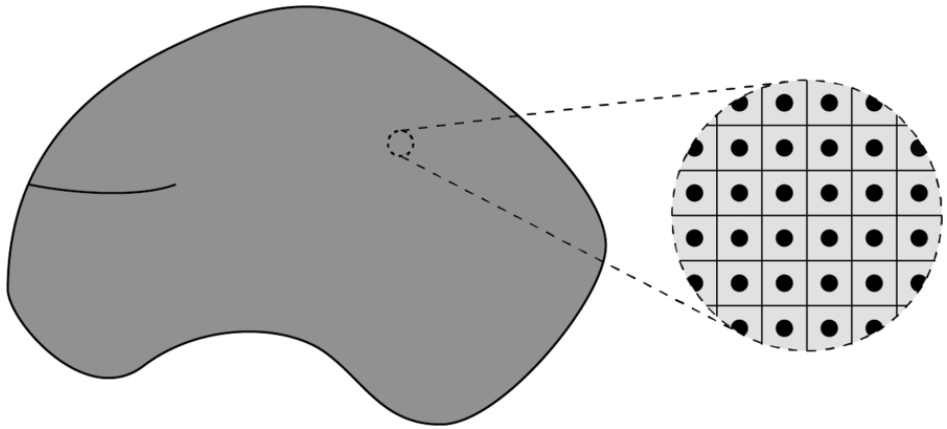
Courtesy M. Thomas (Airbus) and S. Brisard (ENPC)

Problem

Compute u_ε, the response of a material whose properties are defined by the oscillatory coefficient A_ε.

Elliptic equation with highly oscillating coefficients

$$\begin{cases} -\operatorname{div}\left[A_\varepsilon(x)\,\nabla u_\varepsilon(x)\right] = f(x) \quad \text{in } D, \\ u_\varepsilon(\cdot) = 0 \quad \text{on } \partial D. \end{cases}$$



where $D \subset \mathbb{R}^d$, $\varepsilon \ll \operatorname{diam}(D)$, and the matrix A_ε is elliptic, bounded and is varying at the small scale ε.

Galerkine approach

Galerkine variational problem

Find $u_H \in V_H \subset H_0^1(\Omega)$ such that

$$a_\varepsilon(u_H, v_H) = \int_\Omega (A_\varepsilon \nabla u_H) \cdot \nabla v_H = b(v_H) = \int_\Omega f\, v_H, \quad \forall v_H \in V_H$$

Denoting $\{\phi_i^H\}_{\{i=1..N_H\}}$ a basis of V_H, it is equivalent to solving the linear system

Matrix problem

Find $U \in \mathbb{R}^{N_H}$ such that

$$KU = B,$$

with $K_{i,j} = a_\varepsilon(\phi_i^H, \phi_j^H)$, and $B_i = b(\phi_i^H)$.

Toy example in 1D

1D Toy problem

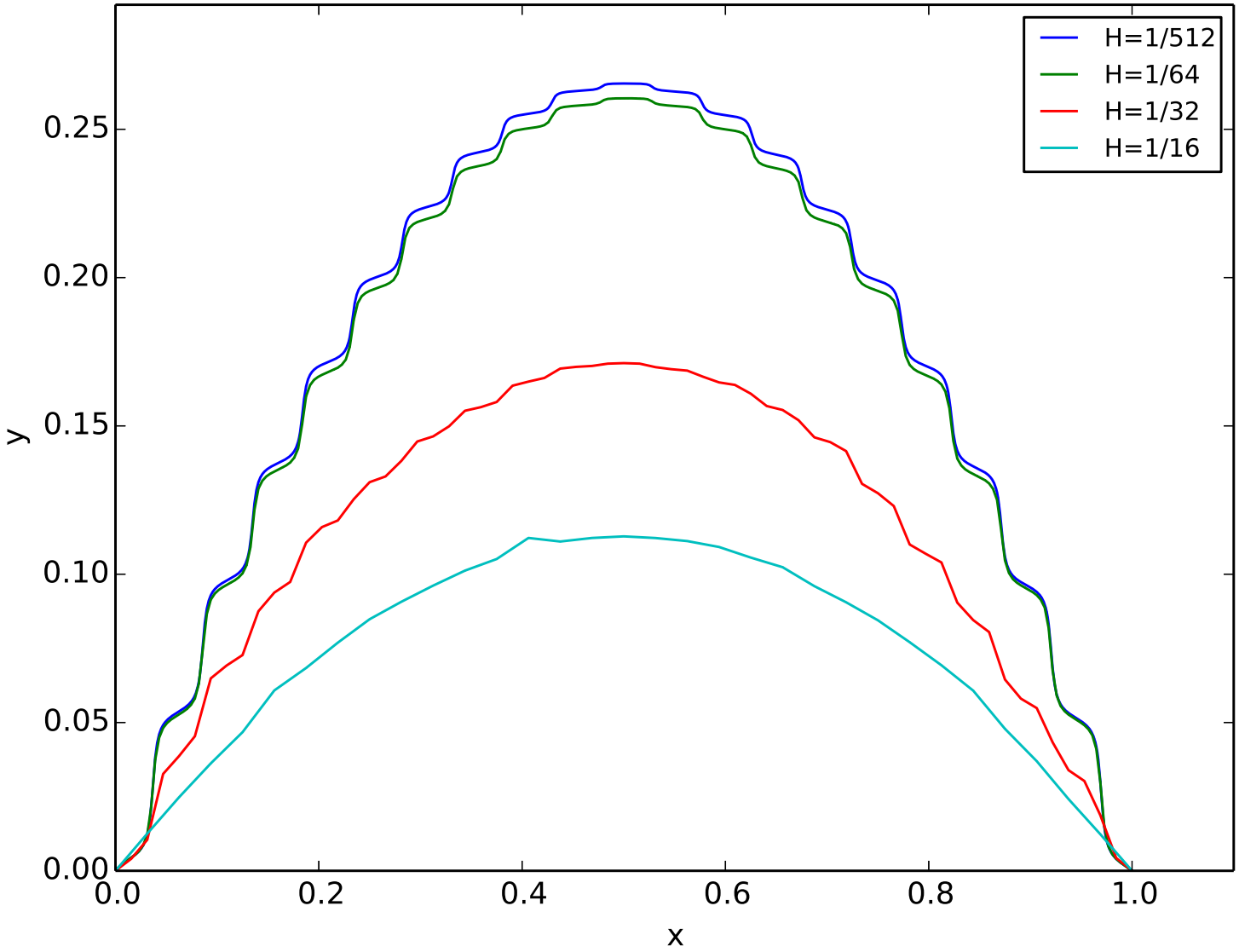
$$\begin{cases} -\left(a_\varepsilon(x)u'_\varepsilon(x)\right)' = 1 \quad \text{in } [0,1], \\ u_\varepsilon(0) = u_\varepsilon(1) = 0, \end{cases}$$

with $a_\varepsilon(x) = a(\frac{x}{\varepsilon}) = 1.1 + \sin(\frac{x}{\varepsilon})$.

Numerical experiment:

- Set $\varepsilon = \frac{1}{128}$: A_ε is periodic with period $\simeq \frac{1}{20}$
- Compute FEM solution for $H = \frac{1}{16}, \frac{1}{32}, \frac{1}{64}$
- Compare to reference solution (obtained for tiny $H = \frac{1}{512}$)

Oscillating problem in 1D - FEM Results

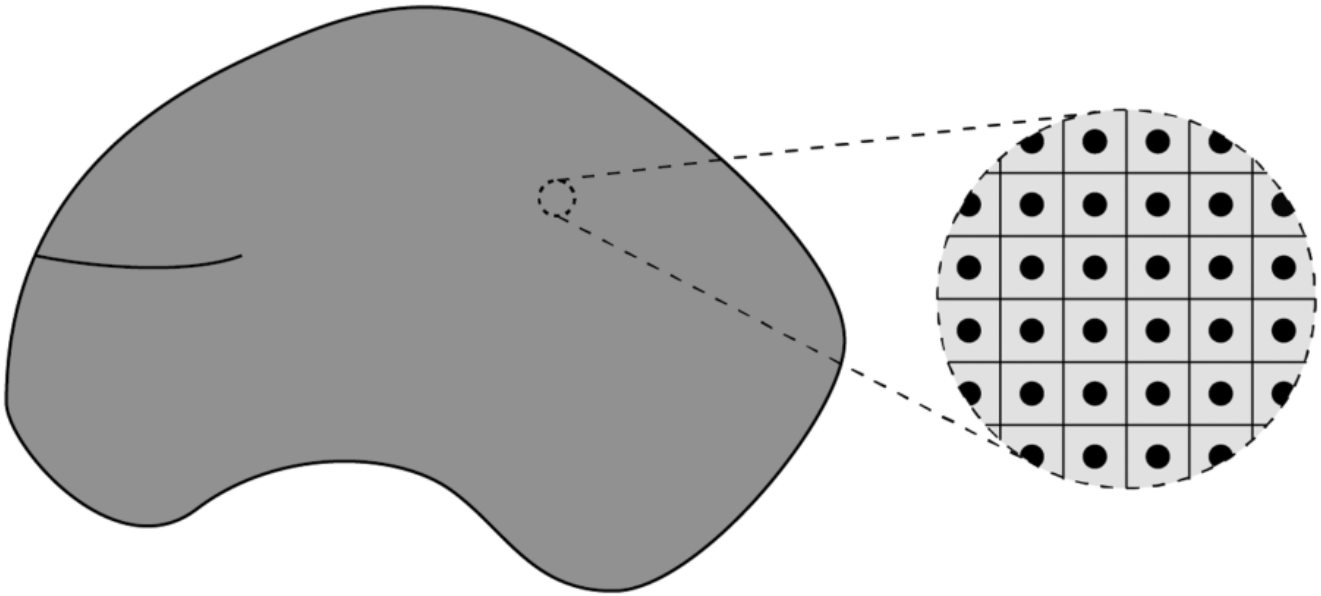


Multi-scale methods: bottom up approach

We want to use the micro-scale knowledge of the problem to compute solutions accurately at a limited computational cost.

Different ways to exploit this knowledge lead to different multi-scale methods.

Usually, local computations at a fine scale are used to improve accuracy on a global scale.



Some examples of multi-scale methods

Under some geometric (periodicity, stationarity...) assumptions:

- Homogenization techniques [Bensoussan, Lions and Papanicolaou, 1978]

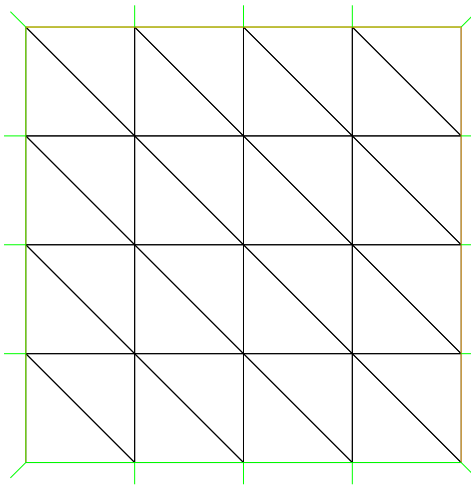
Numerical approaches without such assumption:

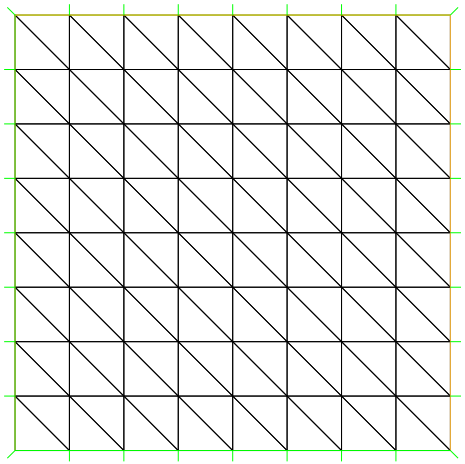
- Multi-scale Finite Element Method (MsFEM) [Hou and Wu, 1997]
- The Heterogeneous Multi-scale Method (HMM) [E, Engquist, 2003], very similar to the FE^2 technique [Feyel,Chaboche, 2000]

We focus here on the MsFEM.

Mesh used

Let us consider a coarse mesh (mesh size H) of D

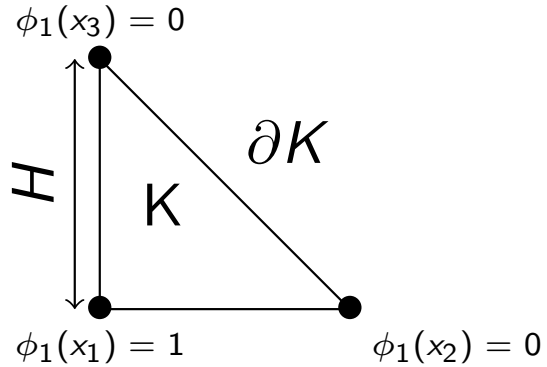




Linear MsFEM - 1

- Offline step: Create a nodal basis on each element K which is adapted to the problem.

$$\begin{cases} -\mathrm{div}(A_\varepsilon \nabla \phi_i) = 0 \quad \text{in } K \\ \phi_i \text{ is linear on the edges} \\ \phi_i(x_j) = \delta_{i,j} \end{cases}$$



- Online step: Solve the coarse Galerkine problem for any source term f: find $u_{MsFEM} \in V_{MsFEM}$ such that

$$a_\varepsilon(u_{MsFEM}, v) =< f, v >, \quad \forall v \in V_{MsFEM} = \mathrm{Span}(\{\phi_i\})$$

Approximation result, periodic case [Hou and Wu, 1999]

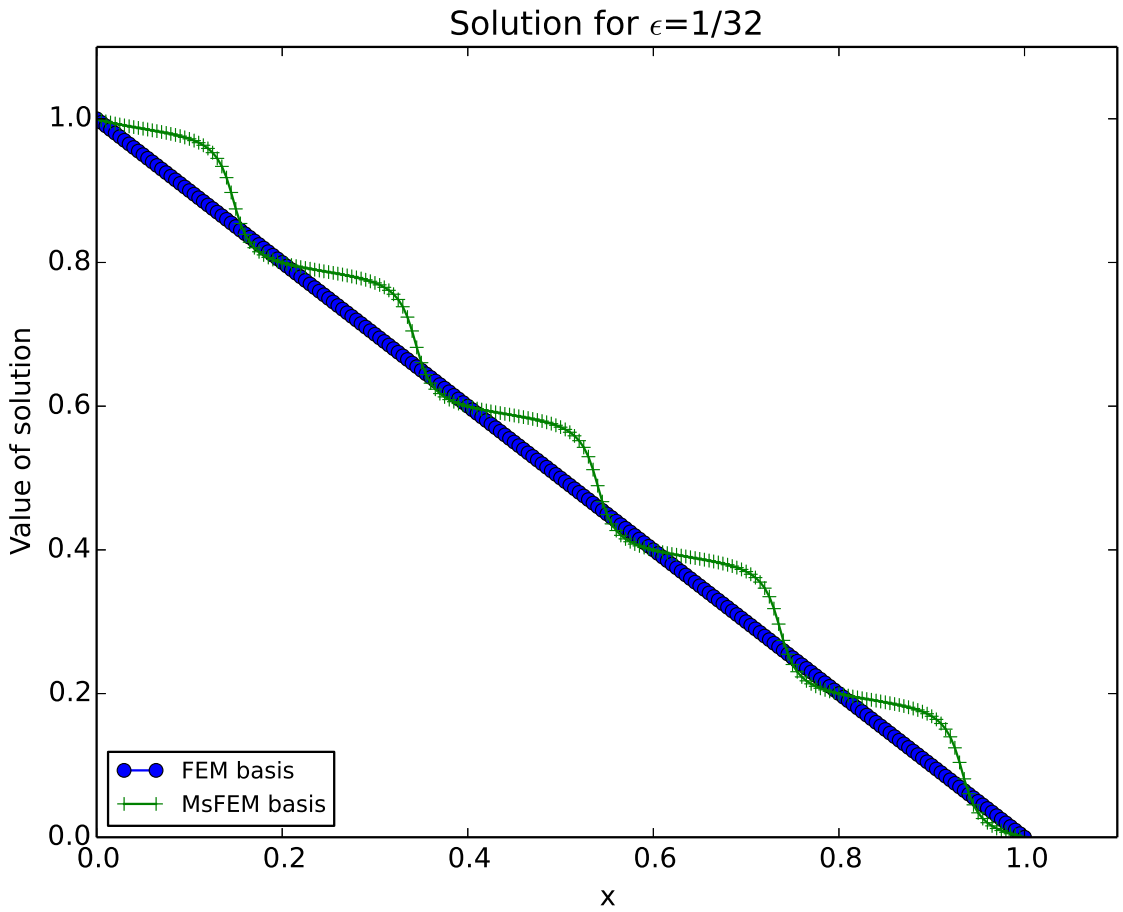
$$\|u_\varepsilon - u_{MsFEM}\|_{H^1(D)} \leq C\left(\sqrt{\varepsilon} + H + \sqrt{\frac{\varepsilon}{H}}\right)$$

Linear MsFEM - 2

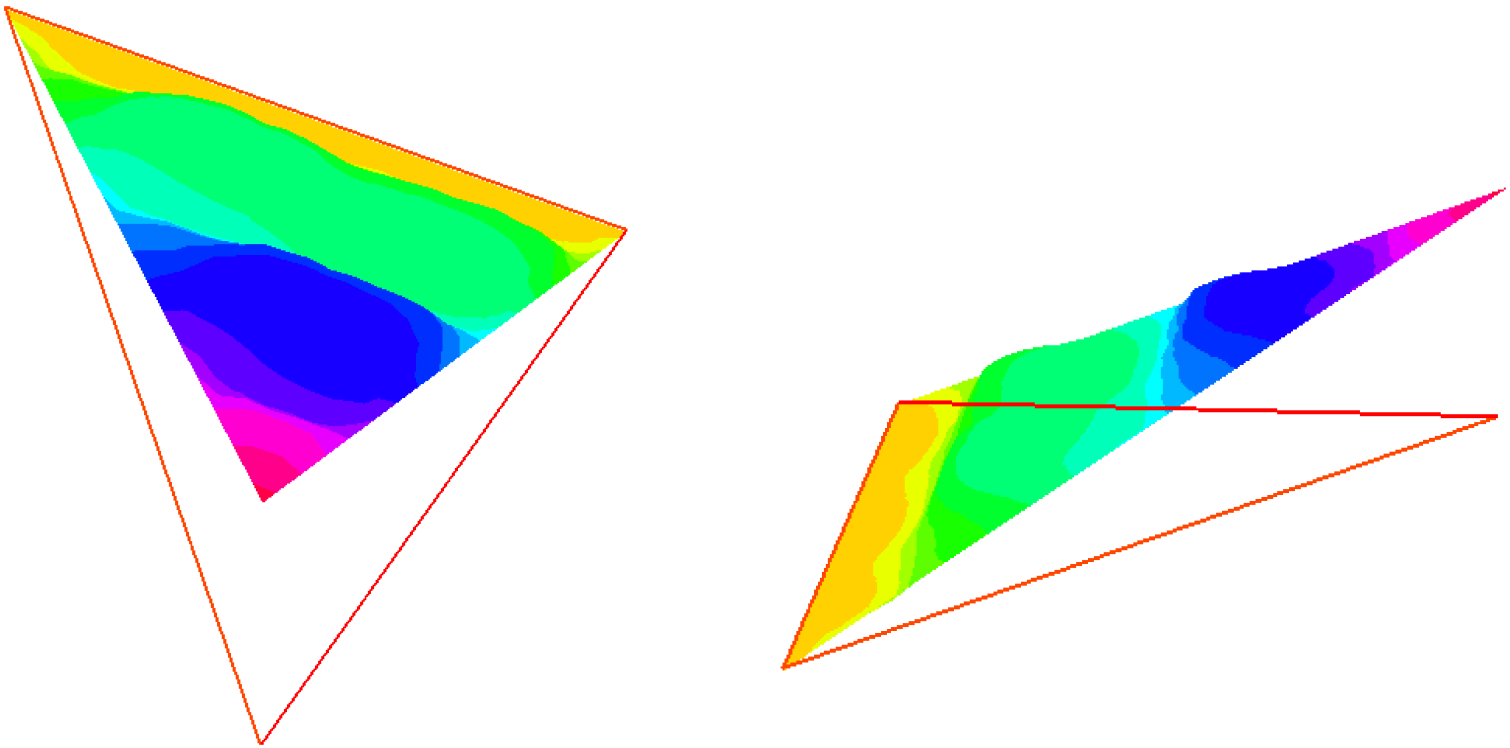
- The MsFEM basis ϕ_i is defined by local problems that can be solved in parallel (usually solved with FEM on a finer embedded grid)
- The offline stage is independent from the source term f, hence we can use the same basis for multiple f.
- The stiffness matrix $K_{i,j} = a_\varepsilon(\phi_i, \phi_j)$ is precomputed (in the offline stage) to speed up the online phase.
- Cost of the online phase: assemble the right-hand side term $B_i = b(\phi_i)$ and solve the coarse linear system $KU = B$.

Example of Linear MsFEM basis in 1D

Finite Element basis function and MsFEM basis function for $a_\varepsilon(x) = 1.1 + \sin(\frac{x}{\varepsilon})$ of period $\simeq 1/5$



Example of Linear MsFEM basis function



Discretization error

- On the edges of the coarse mesh, the MsFEM approximation is linear but the exact solution is oscillating
- Moreover, when H decreases to ε, the error may increase.

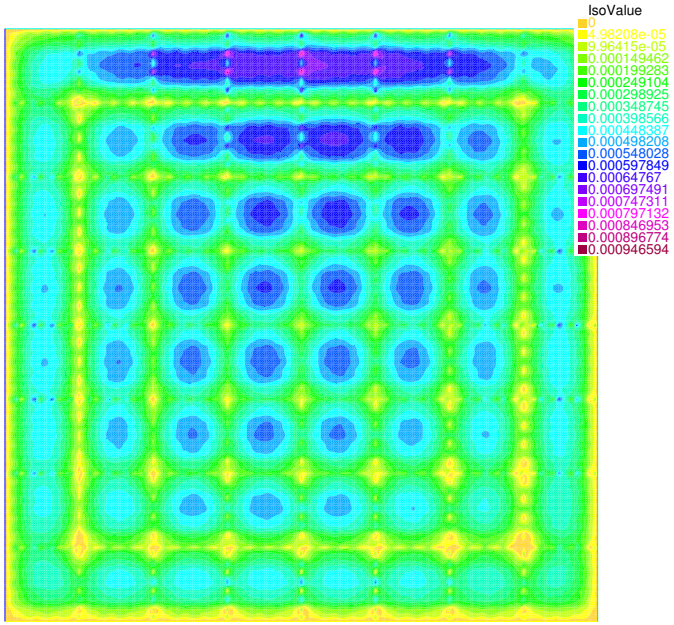


Figure : MsFEM error $H = 4\varepsilon = 1/8$ (Relative H^1 error =19%)

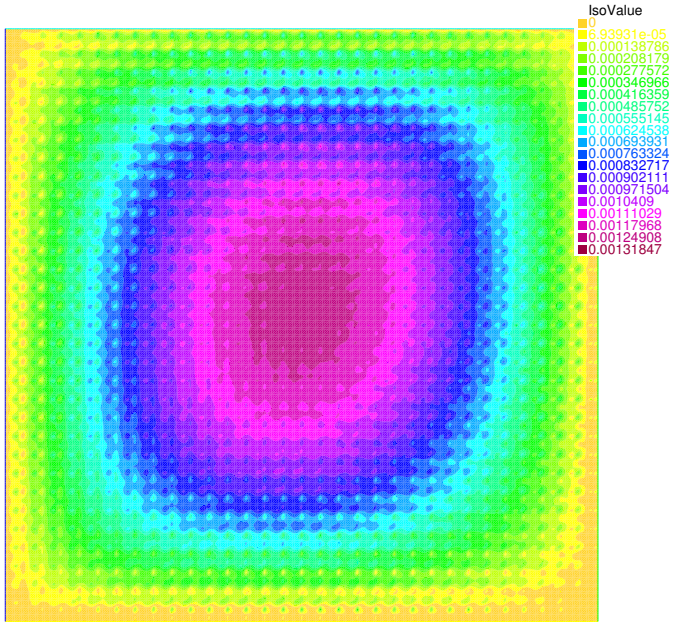


Figure : MsFEM error $H = \varepsilon = 1/32$ (Relative H^1 error =25%)

Advantages of the MsFEM

- Offline step can be parallelized as only local problems are considered. They are independent from each other.
- Better results for coarse discretization compared to FEM
- Useful when computation have to be repeated for multiple source term f (Multi-query context)
- In contrast to standard FE approaches, the MsFEM converges when $\varepsilon \to 0$, then NEXT $H \to 0$

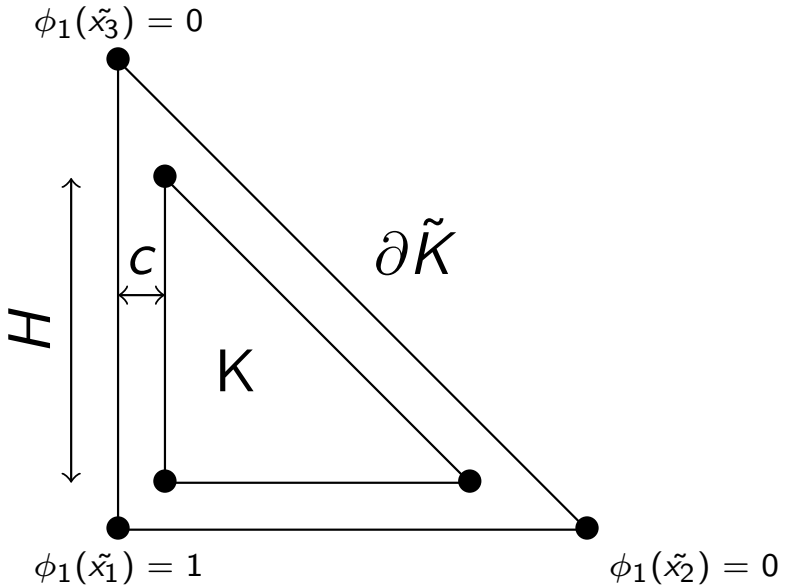
Alternative: MsFEM oversampling - 1

Let us consider a coarse mesh (mesh size H) of D

- Offline step:

$$\begin{cases} -\mathrm{div}(A_\varepsilon \nabla \psi_i) = 0 \quad \text{in } \widetilde{K} \supset K \\ \psi_i \text{ is linear on the edges of } \widetilde{K} \supset K \\ \psi_i(\widetilde{x}_j) = \delta_{i,j} \end{cases}$$

and $\phi_i = \psi_i|_K$



- Online step: Solve the coarse Galerkine problem for any source term f: find $u_{MsFEM} \in V_{MsFEM}$ such that

$$a_\varepsilon^H(u_{MsFEM}, v) =< f, v >, \quad \forall v \in V_{MsFEM} = \mathrm{Span}(\{\phi_i\})$$

with $a_\varepsilon^H(u, v) = \sum_{K \in \mathcal{T}_H} \int_K A_\varepsilon \nabla u \cdot \nabla v$

Alternative: MsFEM oversampling - 2

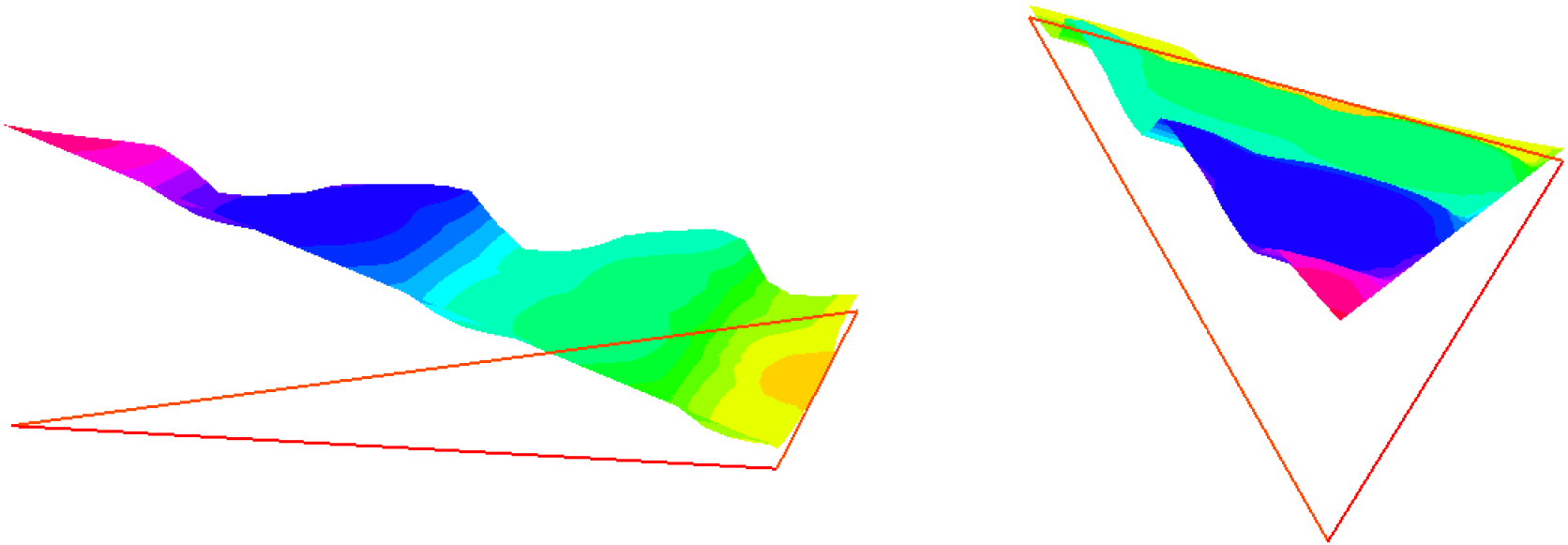
Pros:

- The MsFEM approximation is oscillating on the edges
- Numerical results show a significant improvement compared to linear MsFEM

Cons:

- The approximation is no longer conformal (discontinuous basis functions)
- The size of $\widetilde{K}$ should be chosen carefully

Example of MsFEM oversampling basis function



Comparison with linear MsFEM

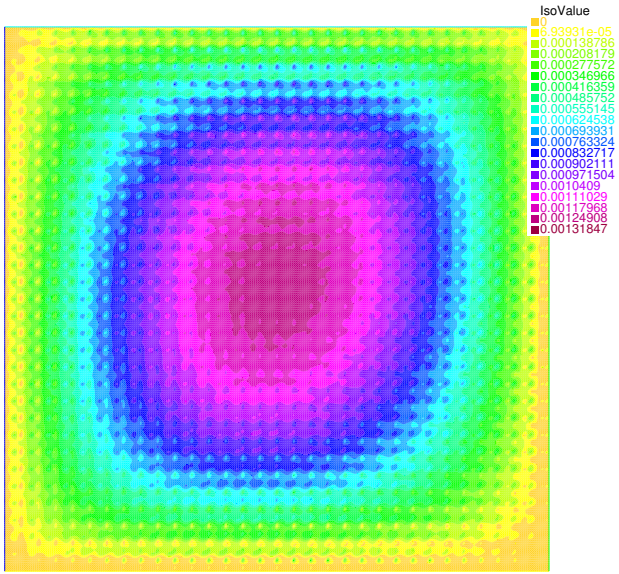


Figure : Error of Linear MsFEM $H = \varepsilon = 1/32$ (Relative H^1 error =25%)

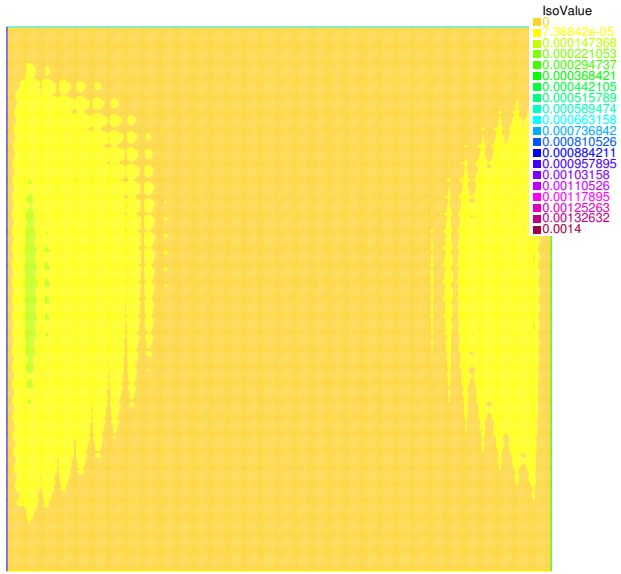


Figure : Error of Oversampling MsFEM $H = \varepsilon = 1/32$ (Relative H^1 error =4%)

Implementation in FreeFem++ - 1

After discussions with F. Hecht, we propose the following.

Providing a coarse mesh, a linear form b and an oscillating bilinear form a, the implementation is performed in two stages:

Offline stage (parallel step)

1. On each element of the coarse mesh, generate a fine mesh, compute the MsFEM basis functions and store them
2. Precompute the local stiffness matrices
3. Precompute RHS term of the form $\int_K \phi_i$

Online stage (for a given RHS f)

1. Assemble the RHS term and solve the coarse matrix system:
$$b(\phi_i) = \int_\Omega f\,\phi_i \simeq \sum_{K\in\mathcal{T}_H} f(x_K) \int_K \phi_i$$
2. Post-process of the numerical solution can be performed in parallel

Implementation in FreeFem++ - 2

```
// Mesh def and generation
int N=16; // Number of coarse mesh elements per direction
int nsplit=32; // ratio between coarse and fine mesh sizes

mesh TH=square(N,N,[x,y]); // Global coarse mesh
fespace P0Tri(TH,P0); // P0 on coarse mesh
fespace P1Tri(TH,P1); // P1 on coarse mesh
P0Tri ChiK=0;

// Generating fine submeshes
mesh[int] THK(NbCoarseElt), ThK(NbCoarseElt);
// Loop over the coarse elements #i
{
    ChiK[][i]=1; // P0 function used to mark the element i
    THK[i]=trunc(TH,ChiK>0.5,split=1); // Coarse mesh of coarse element
    ThK[i]=trunc(THK[i],1,split=nsplit); // Fine mesh of coarse element
    ChiK[][i]=0;
}

// Loop over the coarse elements #i
{
    // Load meshes
    mesh TK=THK[i];
    mesh TKh=ThK[i];
    fespace VKh(TKh,P1);
    fespace VK(TK,P1);
```

Implementation in FreeFem++ - 3

```
    // Loop over the MsFEM basis #j to be computed in the coarse element #i
    {
        // P1 coarse function to set the BC for the MsFEM basis function
        VK Test; Test[][j]=1; // P1 coarse fonction of vertex j

        varf vAK(u,v)=int2d(TKh)(Grad(u)*Grad(v)*aeps)+on(1,2,3,4,u=Test);
        // Bilinear form of local pb
        varf vBK(u,v)= on(1,2,3,4,u=Test); // Linear form of local pb
        real [int] bk=vBK(0,VKh); // RHS for local pb
        matrix AK=vAK(VKh,VKh); // Stiffness matrix for local pb
        uki[j][]=AK^-1*bk; // compute MsFEM basis function #j
    }


    // Double loop on basis functions to compute local stiffness matrix
    // Loop over the MsFEM basis #j
    {
        int J=P1Tri(i,j); // global index of vertex #j of element #i
        // Loop over the MsFEM basis #m
        {
            int M=P1Tri(i,m); // global index of vertex #m of element #i
            real KMsFEM = int2d(TKh)(Grad(uki[j])*Grad(uki[m])*aeps);
            // local stiffness contribution
            AMsFEM(J,M) = AMsFEM(J,M) + KMsFEM; // update global stiffness
        }

        int btestj=(TH[i][j].label>0); // test if dof is on partial Omega
        if (btestj) AMsFEM(J,J)=tgv; // penalization term for Dirichlet BC
    }
}

xx=AMsFEM^-1*Global_RHS;// Computation of the global MsFEM solution
```

Problems on perforated domains

Consider a material with small perforations and a coarse mesh:

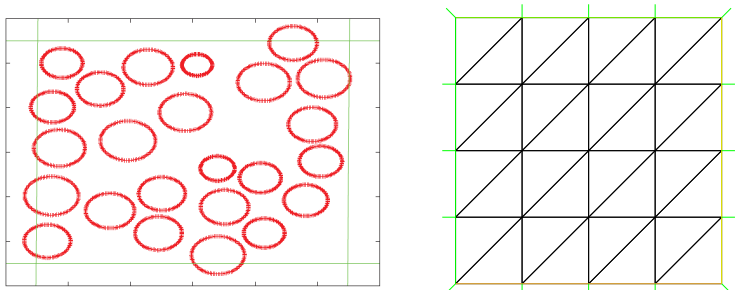


Figure : Left - Material with perforations, Right - Mesh used

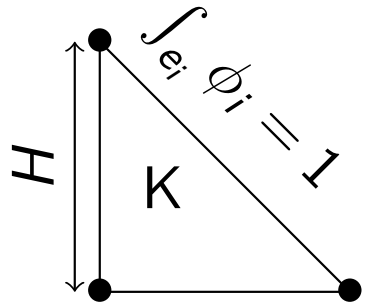
- Typical situation: the perforations intersect the edges of the mesh.
- We cannot apply the Linear MsFEM (or the MsFEM oversampling method): Linear boundary conditions would be enforced where the solution is supposed to vanish.
- An edge based MsFEM method has been designed: *MsFEM à la Crouzeix-Raviart* [Le Bris, Legoll and Lozinski, MMS 2014]

MsFEM Crouzeix-Raviart - 1

Let us consider a coarse mesh (mesh size H) of D

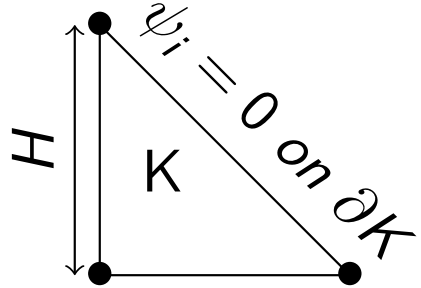
- Introduce edge-based basis functions

$$\begin{cases} -\mathrm{div}(A_\varepsilon \nabla \phi_i) = 0 \quad \text{in } K \\ \int_{e_i} \phi_i = 1 \\ \int_{e_j} \phi_i = 0, \ i \neq j \\ A_\varepsilon \nabla \phi_i \cdot n_j = \lambda_{i,j} \text{ on } e_j \end{cases}$$



- Introduce bubble-basis functions

$$\begin{cases} -\mathrm{div}(A_\varepsilon \nabla \psi_i) = 1 \quad \text{in } K \\ \psi_i = 0 \text{ on } \partial K \end{cases}$$



MsFEM Crouzeix-Raviart - 2

Basis functions are not continuous across the edges. However, they satisfy a weak continuity property, and hence

$$\text{for any edge } e \text{ of the mesh,} \quad \int_e [[u_{MsFEM}]] = 0$$

Solve the coarse Galerkine problem for any source term f:
find $u_{MsFEM} \in V_{MsFEM}$ such that

$$a_\varepsilon^H(u_{MsFEM}, v) =< f, v >, \quad \forall v \in V_{MsFEM} = \mathrm{Span}(\{\phi_i\}, \{\psi_i\})$$

with $a_\varepsilon^H(u, v) = \sum_{K \in \mathcal{T}_H} \int_K A_\varepsilon \nabla u \cdot \nabla v$

Implementation in FreeFem++

Main difficulties:

1. Find the appropriate structure for manipulating DOF and defining stiffness matrix

```
1  mesh TH=square(N,N,[x,y]);// Coarse global mesh
2  fespace P0P0edge(TH,[P0,P0edge]);//FE space similar to MsFEM basis
3  varf VZERO(u,v)=int2d(TH)(0.*u*v);// Dummy expression
4  matrix AMsFEM=VZERO(P0P0edge,P0P0edge);// Stiffness matrix
5  real[int] RhsMsFEM(P0P0edge.ndof);// RHS vector
```

2. Enforce the boundary conditions for local problems

```
1  fespace VKP0edge(TK,P0edge); fespace VKhP1(TKh,P1);
2  VKP0edge ei; ei[][i]=1; // function with value 1 on edge i
3  VKhP1 usol;// edge based MsFEM basis phi_i
4
5  matrix CK=vCedge(VKP0edge,VKhP1)=int1d(Kh)(u*v);//Lagrange mat
6  matrix Asub=vAK(VKhP1,VKhP1)=int2d(Kh)(Aeps(u,v));//Stiffness mat
7  matrix AK=[ [Asub,CK],[(CK)^T,0]]; // Stiffness matrix
8
9  lagrangerhs=vBK(0,VKP0edge)=-int1d(TK)(v*ei);//Lagrange mult RHS
10 real [int] bsub(VKhP1.ndof); // RHS solution
11 real[int] bk=[bsub, lagrangerhs]; // RHS
12
13 [usol,lagrangemult]=AK^-1*bk;// compute MsFEM basis function
```

Conclusions and perspectives

We have designed a FreeFem++ template allowing one to easily implement several variants of MsFEM, including

- With linear Boundary Conditions (conforming, local problems posed on coarse elements)
- With Crouzeix-Raviart type Boundary Conditions (non-conforming, local problems posed on coarse elements)
- Oversampling variant (non-conforming, local problems posed on enlarged elements)