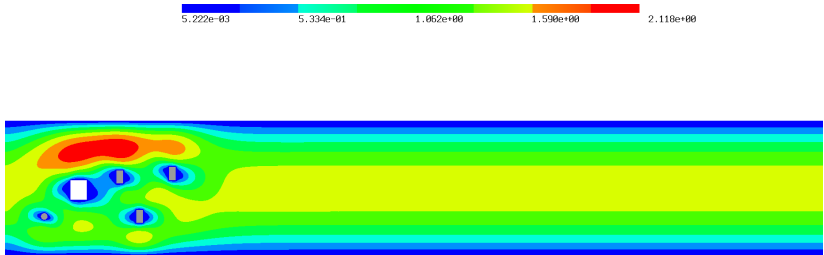
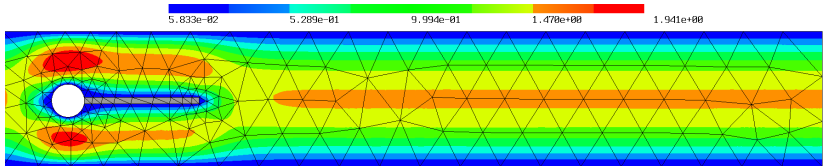


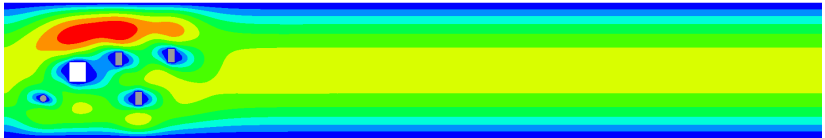
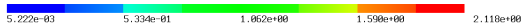
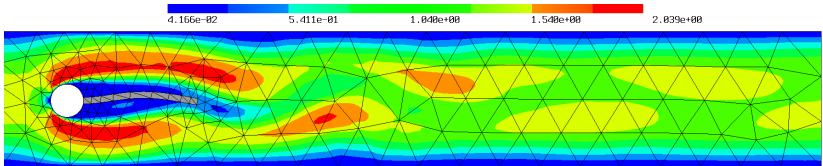
Moving meshes

Michael Neunteufel, Joachim Schöberl



3rd NGSolve Usermeeting, Vienna, July 1-3, 2019



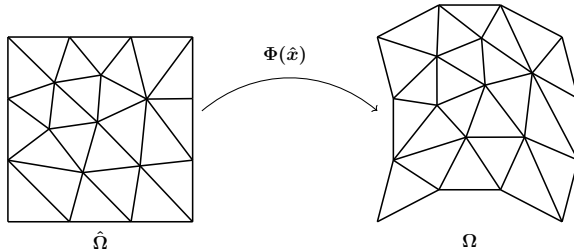


Arbitrary Lagrangian Eulerian

Deformation extension

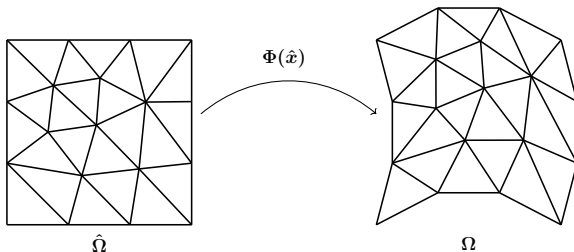
Optimizing mesh

Arbitrary Lagrangian Eulerian



$$\begin{aligned} -\Delta_x u &= f && \text{in } \Omega \\ u &= u_D && \text{on } \Gamma \end{aligned}$$

$$\int_{\Omega} \nabla_x u \nabla_x v \, dx = \int_{\Omega} f v \, dx$$

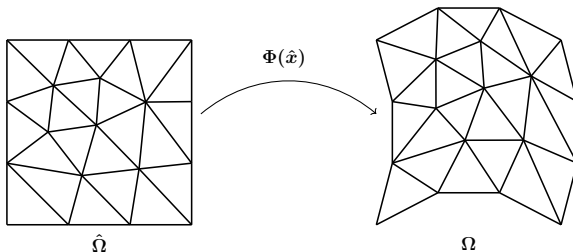


$$\begin{aligned} -\Delta_x u &= f && \text{in } \Omega \\ u &= u_D && \text{on } \Gamma \end{aligned}$$

$$u \circ \Phi = \hat{u}$$

$$\underbrace{\nabla_{\hat{x}} \Phi^T}_{=F^T} (\nabla_x u) \circ \Phi = \nabla_{\hat{x}} \hat{u}$$

$$\int_{\Omega} \nabla_x u \nabla_x v \, dx = \int_{\Omega} f v \, dx$$

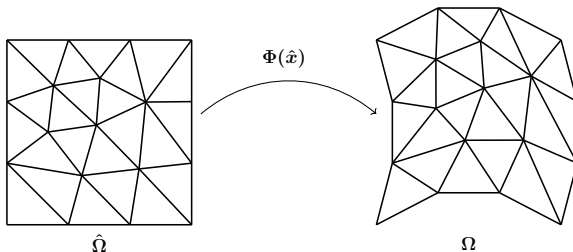


$$\begin{aligned} -\Delta_x u &= f && \text{in } \Omega \\ u &= u_D && \text{on } \Gamma \end{aligned}$$

$$u \circ \Phi = \hat{u}$$

$$\underbrace{\nabla_{\hat{x}} \Phi^T}_{=F^T} (\nabla_x u) \circ \Phi = \nabla_{\hat{x}} \hat{u}$$

$$\int_{\hat{\Omega}} J \nabla_x u \circ \Phi \nabla_x v \circ \Phi d\hat{x} = \int_{\hat{\Omega}} J f \circ \Phi v \circ \Phi d\hat{x} \quad J = \det(\mathbf{F})$$



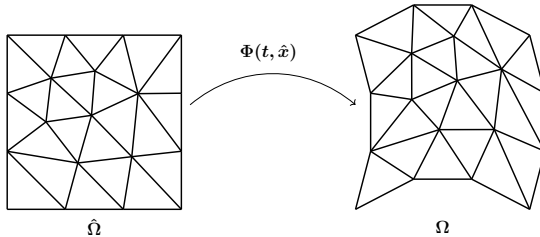
$$\begin{aligned} -\Delta_x u &= f && \text{in } \Omega \\ u &= u_D && \text{on } \Gamma \end{aligned}$$

$$u \circ \Phi = \hat{u}$$

$$\int_{\hat{\Omega}} J \mathbf{F}^{-T} \nabla_{\hat{x}} \hat{u} \mathbf{F}^{-T} \nabla_{\hat{x}} \hat{v} d\hat{x} = \int_{\hat{\Omega}} J \hat{f} \hat{v} d\hat{x}$$

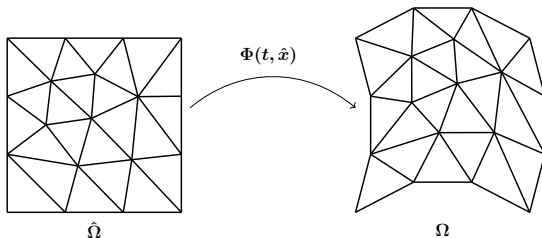
$$\underbrace{\nabla_{\hat{x}} \Phi^T}_{=\mathbf{F}^T} (\nabla_x u) \circ \Phi = \nabla_{\hat{x}} \hat{u}$$

$$J = \det(\mathbf{F})$$



$$\begin{aligned} \frac{\partial u}{\partial t} - \Delta_x u &= f && \text{in } \Omega \\ u &= u_D && \text{on } \Gamma \end{aligned}$$

$$\int_{\Omega} \frac{\partial u}{\partial t} v \, dx$$



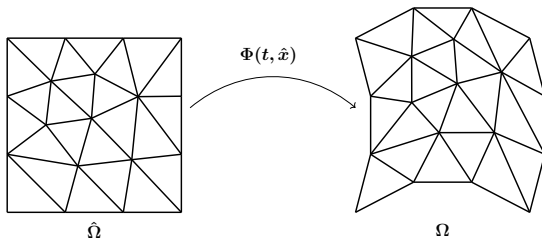
$$\frac{\partial u}{\partial t} - \Delta_x u = f \quad \text{in } \Omega$$

$$u = u_D \quad \text{on } \Gamma$$

$$\int_{\Omega} \frac{\partial u}{\partial t} v \, dx$$

$$u(t, \Phi(t, \hat{x})) = \hat{u}(t, \hat{x})$$

$$\frac{\partial u}{\partial t} + \nabla_x u \frac{\partial \Phi}{\partial t} = \frac{\partial \hat{u}}{\partial t}$$



$$\frac{\partial u}{\partial t} - \Delta_x u = f \quad \text{in } \Omega$$

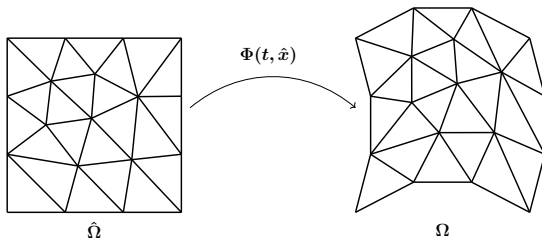
$$u = u_D \quad \text{on } \Gamma$$

$$\int_{\Omega} \frac{\partial u}{\partial t} v \, dx$$

$$u(t, \Phi(t, \hat{x})) = \hat{u}(t, \hat{x})$$

$$\frac{\partial u}{\partial t} + \nabla_x u \frac{\partial \Phi}{\partial t} = \frac{\partial \hat{u}}{\partial t}$$

$$\frac{\partial u}{\partial t} + \mathbf{F}^{-T} \nabla_{\hat{x}} \hat{u} \frac{\partial \Phi}{\partial t} = \frac{\partial \hat{u}}{\partial t}$$



$$\frac{\partial u}{\partial t} - \Delta_x u = f \quad \text{in } \Omega$$

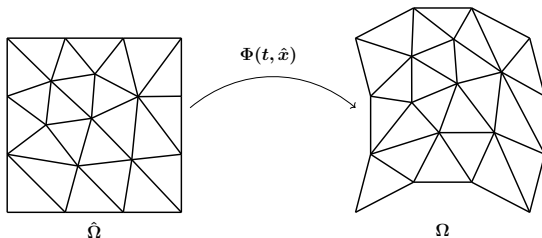
$$u = u_D \quad \text{on } \Gamma$$

$$\int_{\hat{\Omega}} J \frac{\partial u}{\partial t} \circ \Phi \, v \circ \Phi \, d\hat{x}$$

$$u(t, \Phi(t, \hat{x})) = \hat{u}(t, \hat{x})$$

$$\frac{\partial u}{\partial t} + \nabla_x u \frac{\partial \Phi}{\partial t} = \frac{\partial \hat{u}}{\partial t}$$

$$\frac{\partial u}{\partial t} + \mathbf{F}^{-T} \nabla_{\hat{x}} \hat{u} \frac{\partial \Phi}{\partial t} = \frac{\partial \hat{u}}{\partial t}$$



$$\frac{\partial u}{\partial t} - \Delta_x u = f \quad \text{in } \Omega$$

$$u = u_D \quad \text{on } \Gamma$$

$$\int_{\hat{\Omega}} J \left(\frac{\partial \hat{u}}{\partial t} - \mathbf{F}^{-T} \nabla_{\hat{x}} \hat{u} \frac{\partial \Phi}{\partial t} \right) \hat{v} d\hat{x}$$

$$u(t, \Phi(t, \hat{x})) = \hat{u}(t, \hat{x})$$

$$\frac{\partial u}{\partial t} + \nabla_x u \frac{\partial \Phi}{\partial t} = \frac{\partial \hat{u}}{\partial t}$$

$$\frac{\partial u}{\partial t} + \mathbf{F}^{-T} \nabla_{\hat{x}} \hat{u} \frac{\partial \Phi}{\partial t} = \frac{\partial \hat{u}}{\partial t}$$

$$\begin{aligned}\frac{\partial u}{\partial t} - \Delta_{\mathbf{x}} u &= f && \text{in } \Omega \\ u &= u_D && \text{on } \Gamma\end{aligned}$$

$$\int_{\hat{\Omega}} J \left(\frac{\partial \hat{u}}{\partial t} \hat{\mathbf{v}} - \mathbf{F}^{-T} \nabla_{\hat{\mathbf{x}}} \hat{u} \frac{\partial \Phi}{\partial t} \hat{\mathbf{v}} + \mathbf{F}^{-T} \nabla_{\hat{\mathbf{x}}} \hat{u} \mathbf{F}^{-T} \nabla_{\hat{\mathbf{x}}} \hat{\mathbf{v}} \right) d\hat{\mathbf{x}} = \int_{\hat{\Omega}} J \hat{f} \hat{\mathbf{v}} d\hat{\mathbf{x}}$$

$$\begin{aligned}\frac{\partial u}{\partial t} - \Delta_x u &= f && \text{in } \Omega \\ u &= u_D && \text{on } \Gamma\end{aligned}$$

$$\int_{\hat{\Omega}} J \left(\frac{\partial \hat{u}}{\partial t} \hat{v} - \mathbf{F}^{-T} \nabla_{\hat{x}} \hat{u} \frac{\partial \Phi}{\partial t} \hat{v} + \mathbf{F}^{-T} \nabla_{\hat{x}} \hat{u} \mathbf{F}^{-T} \nabla_{\hat{x}} \hat{v} \right) d\hat{x} = \int_{\hat{\Omega}} J \hat{f} \hat{v} d\hat{x}$$

- $\frac{\partial \Phi}{\partial t}$ mesh velocity

$$\begin{aligned}\frac{\partial u}{\partial t} - \Delta_x u &= f && \text{in } \Omega \\ u &= u_D && \text{on } \Gamma\end{aligned}$$

$$\int_{\hat{\Omega}} J \left(\frac{\partial \hat{u}}{\partial t} \hat{v} - \mathbf{F}^{-T} \nabla_{\hat{x}} \hat{u} \frac{\partial \Phi}{\partial t} \hat{v} + \mathbf{F}^{-T} \nabla_{\hat{x}} \hat{u} \mathbf{F}^{-T} \nabla_{\hat{x}} \hat{v} \right) d\hat{x} = \int_{\hat{\Omega}} J \hat{f} \hat{v} d\hat{x}$$

- $\frac{\partial \Phi}{\partial t}$ mesh velocity
- $\operatorname{div}_x \underline{u} = \operatorname{tr}(\nabla_{\hat{x}} \hat{u} \mathbf{F}^{-1})$

- Given mesh deformation Φ
- Assemble every time step

```
1 gfset      = GridFunction(feset)
2 gfsetold   = GridFunction(feset)
3
4 F = Grad(gfset) + Id(2)
5 J = Det(F)
6
7 diff = J*InnerProduct(Inv(F).trans*grad(u), Inv(F).
8   trans*grad(v))*dx
9 mass = J*u*v*dx
10 conv = -J*InnerProduct(Inv(F).trans*grad(u), 1/tau*(
11   gfset-gfsetold))*v*dx
```

- SetDeformation function/argument

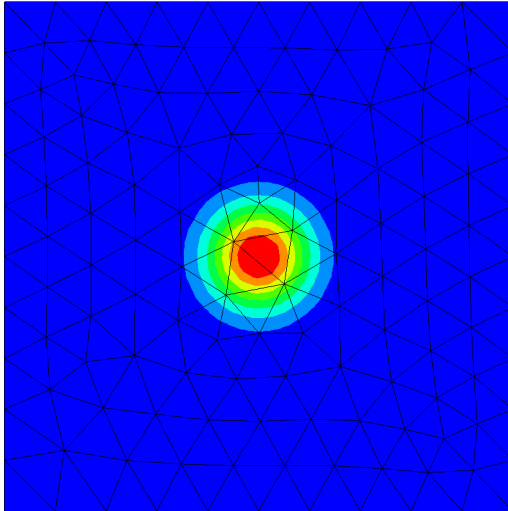
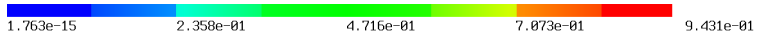
$$\int_{\hat{\Omega}} \left(\frac{\partial \hat{u}}{\partial t} \hat{v} - \nabla_{\hat{x}} \hat{u} \frac{\partial \Phi}{\partial t} \hat{v} + \nabla_{\hat{x}} \hat{u} \nabla_{\hat{x}} \hat{v} \right) d\hat{x} = \int_{\hat{\Omega}} \hat{f} \hat{v} d\hat{x}$$

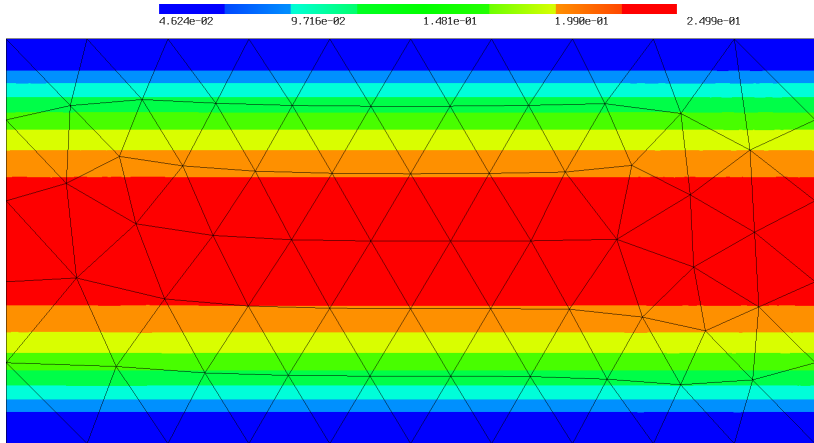
```
1 diff = InnerProduct( grad(u), grad(v))*dx
2 mass = u*v*dx
3 conv = -InnerProduct( grad(u), 1/tau*(gfset-gfsetold))*v
    *dx
4
5 mesh.SetDeformation(gfset)
6 a.Assemble()
7 m.Assemble()
8 b.Apply(gfu.vec,r)
9 f.Assemble()
10 mesh.UnsetDeformation()
```

- SetDeformation function/argument

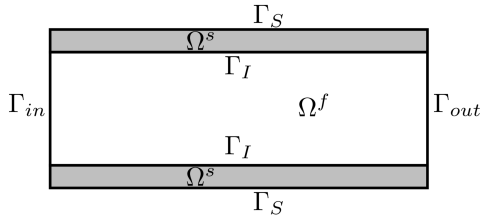
$$\int_{\hat{\Omega}} \left(\frac{\partial \hat{u}}{\partial t} \hat{v} - \nabla_{\hat{x}} \hat{u} \frac{\partial \Phi}{\partial t} \hat{v} + \nabla_{\hat{x}} \hat{u} \nabla_{\hat{x}} \hat{v} \right) d\hat{x} = \int_{\hat{\Omega}} \hat{f} \hat{v} d\hat{x}$$

```
1 diff = InnerProduct(grad(u), grad(v))*dx(deformation=  
    gfset)  
2 mass = u*v*dx(deformation=gfset)  
3 conv = -InnerProduct(grad(u), 1/tau*(gfset-gfsetold))*v  
    *dx(deformation=gfset)  
4  
5 a.Assemble()  
6 m.Assemble()  
7 b.Apply(gfu.vec, r)  
8 f.Assemble()
```



Deformation extension



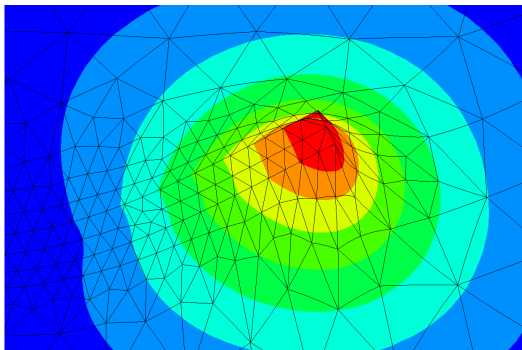
- Displacement variable in elastic part of problem
- Extend displacement to fluid domain

- Solve a simple Poisson problem for a given d^s

$$\begin{aligned} -\Delta d^f &= 0, & \text{in } \Omega^f \\ d^f &= 0, & \text{on } \partial\Omega \\ d^f &= d^s, & \text{on } \Gamma_I \end{aligned}$$

- Solve a simple Poisson problem for a given d^s

$$\begin{aligned} -\Delta d^f &= 0, & \text{in } \Omega^f \\ d^f &= 0, & \text{on } \partial\Omega \\ d^f &= d^s, & \text{on } \Gamma_I \end{aligned}$$



- Elements get pressed through

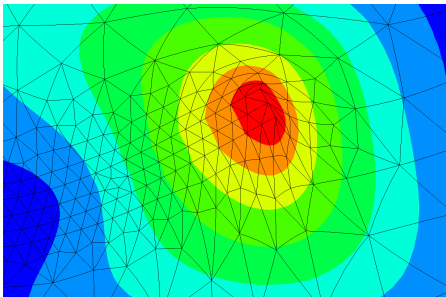
- Penalize compressed elements by Neo-Hooke material law
- Important for stability

$$\mathbf{C} = \mathbf{F}^T \mathbf{F}, \quad \mathbf{E} = 0.5(\mathbf{C} - \mathbf{I})$$
$$N(d) = h(\cdot) \mu \left(\mathbf{E} + \frac{\mu}{\lambda} \det(\mathbf{C})^{-\frac{\lambda}{2\mu}} - 1 \right)$$

- Penalize compressed elements by Neo–Hooke material law
- Important for stability

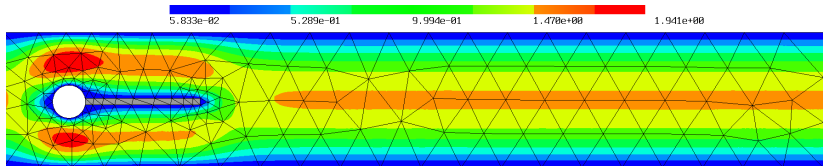
$$\mathbf{C} = \mathbf{F}^T \mathbf{F}, \quad \mathbf{E} = 0.5(\mathbf{C} - \mathbf{I})$$
$$N(d) = h(\cdot)(\mathbf{E} + \det(\mathbf{C})^{-\frac{1}{2}} - 1)$$

- Penalize compressed elements by Neo-Hooke material law
- Important for stability

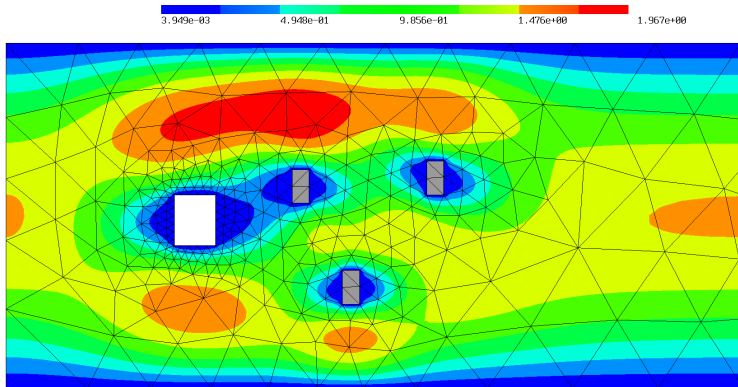


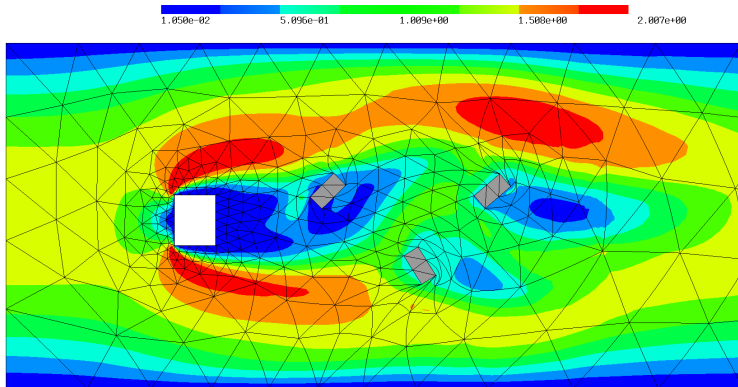
$$\mathbf{C} = \mathbf{F}^T \mathbf{F}, \quad \mathbf{E} = 0.5(\mathbf{C} - \mathbf{I})$$

$$N(d) = h(\cdot)(\mathbf{E} + \det(\mathbf{C})^{-\frac{1}{2}} - 1)$$

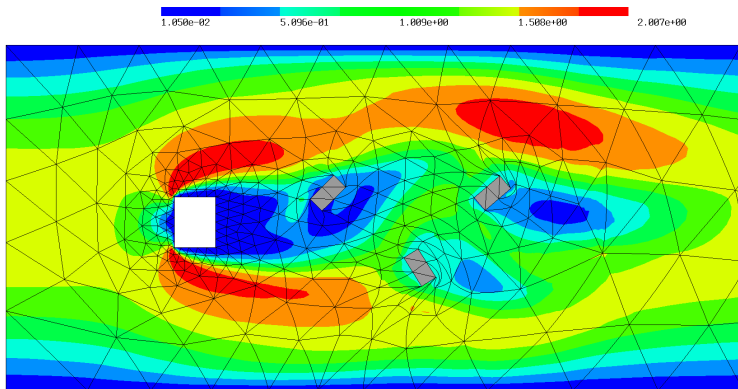


Optimizing mesh

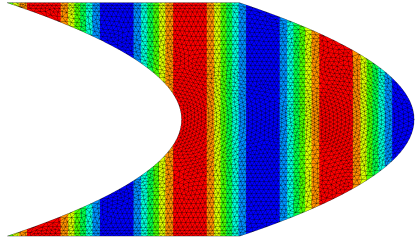
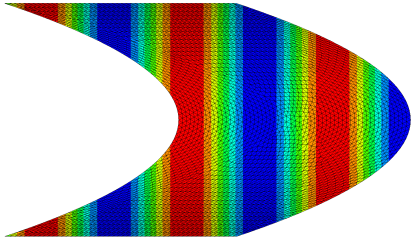


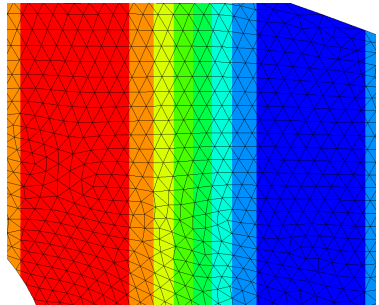
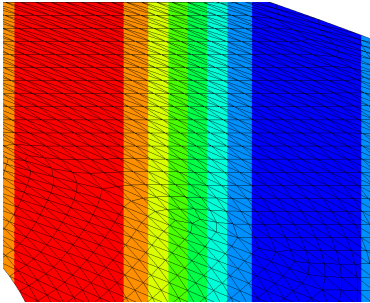


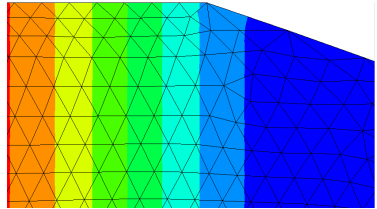
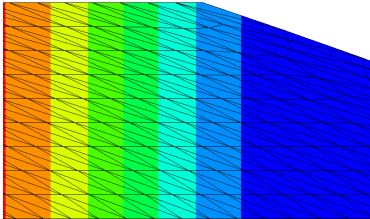
- XFEM (level set function)

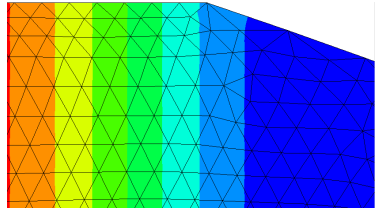
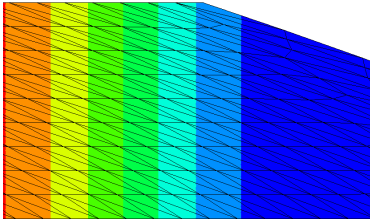


- XFEM (level set function)
- Optimize mesh when quality gets critical

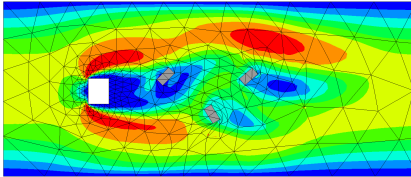


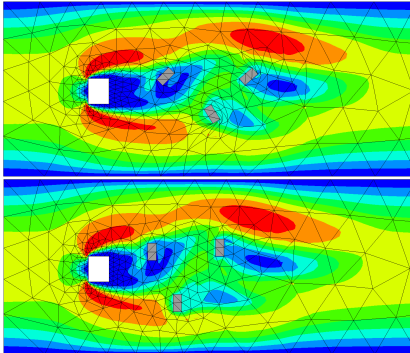


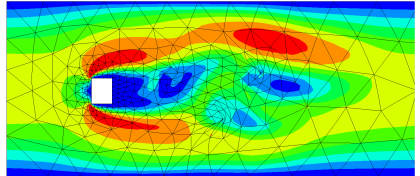
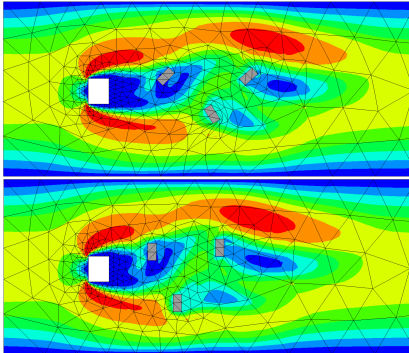




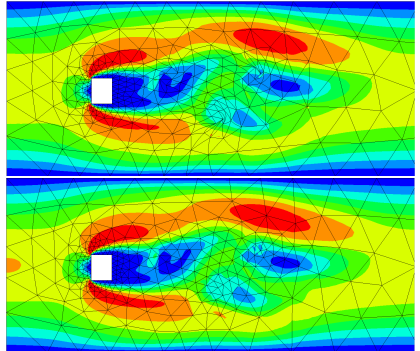
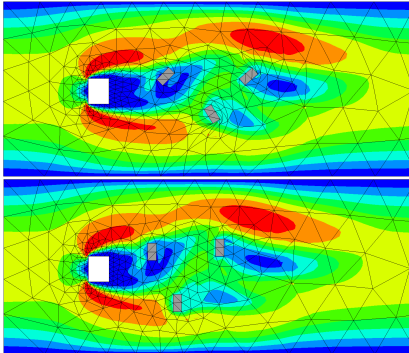
```
1 gfu = GridFunction(H1(mesh, order=3))  
2 gfu.Set(sin(10*x))  
3  
4 mesh2 = Mesh(mesh.ngmesh.Copy())  
5 mesh2.ngmesh.OptimizeMesh2d()  
6  
7 gfu2 = GridFunction(H1(mesh2, order=3))  
8 gfu2.Set(gfu)
```







```
1 tmpmesh = Mesh(mesh.ngmesh.Copy())
2 for p in tmpmesh.ngmesh.Points():
3     p += displacement(p)
4
5 gftmp = GridFunction(tmpfes)
6 gftmp.vec.data = gfu.vec
```



```
1 mesh2 = Mesh(tmpmesh.ngmesh.Copy())
2 ngmesh2.OptimizeMesh2d()
3
4 gfu2 = GridFunction(fes2)
5
6 gfu2.Set(gftmp)
```

