Imperial College London

Anisotropic Mesh Adaptation and Structural Optimisation

- optimising the optimisation

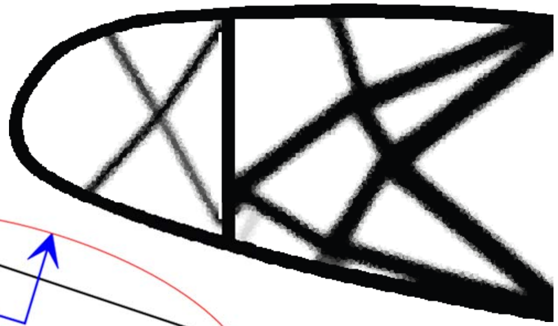
Dr. Kristian Ejlebjerg Jensen

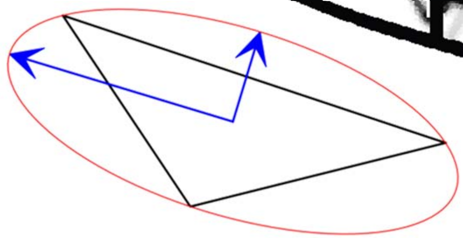
Dr. Gerard Gorman

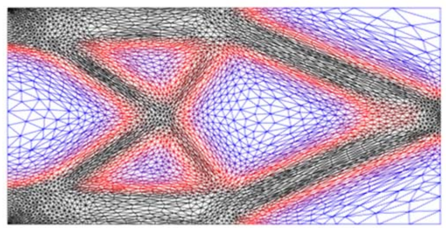
https://github.com/ggorman/pragmatic

Outline

- Introduction
  - Topology Optimisation
  - Anisotropic Mesh Adaptation
- Structural topology optimization
  - Minimum compliance
  - Stress constraints
- Parallelisation
- Future work
- Summary



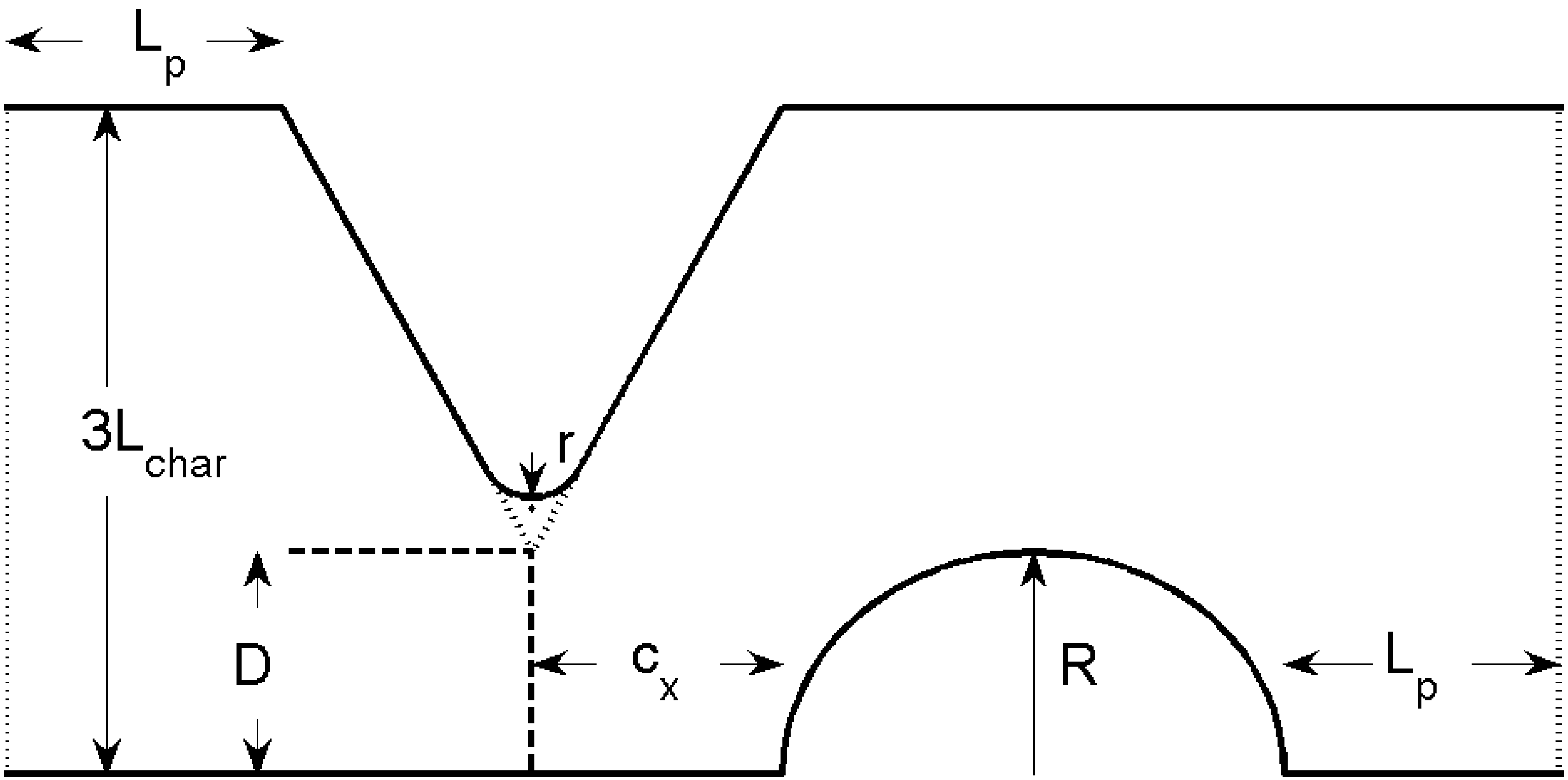




Imperial College London

Basic concept
shape versus topology optimization

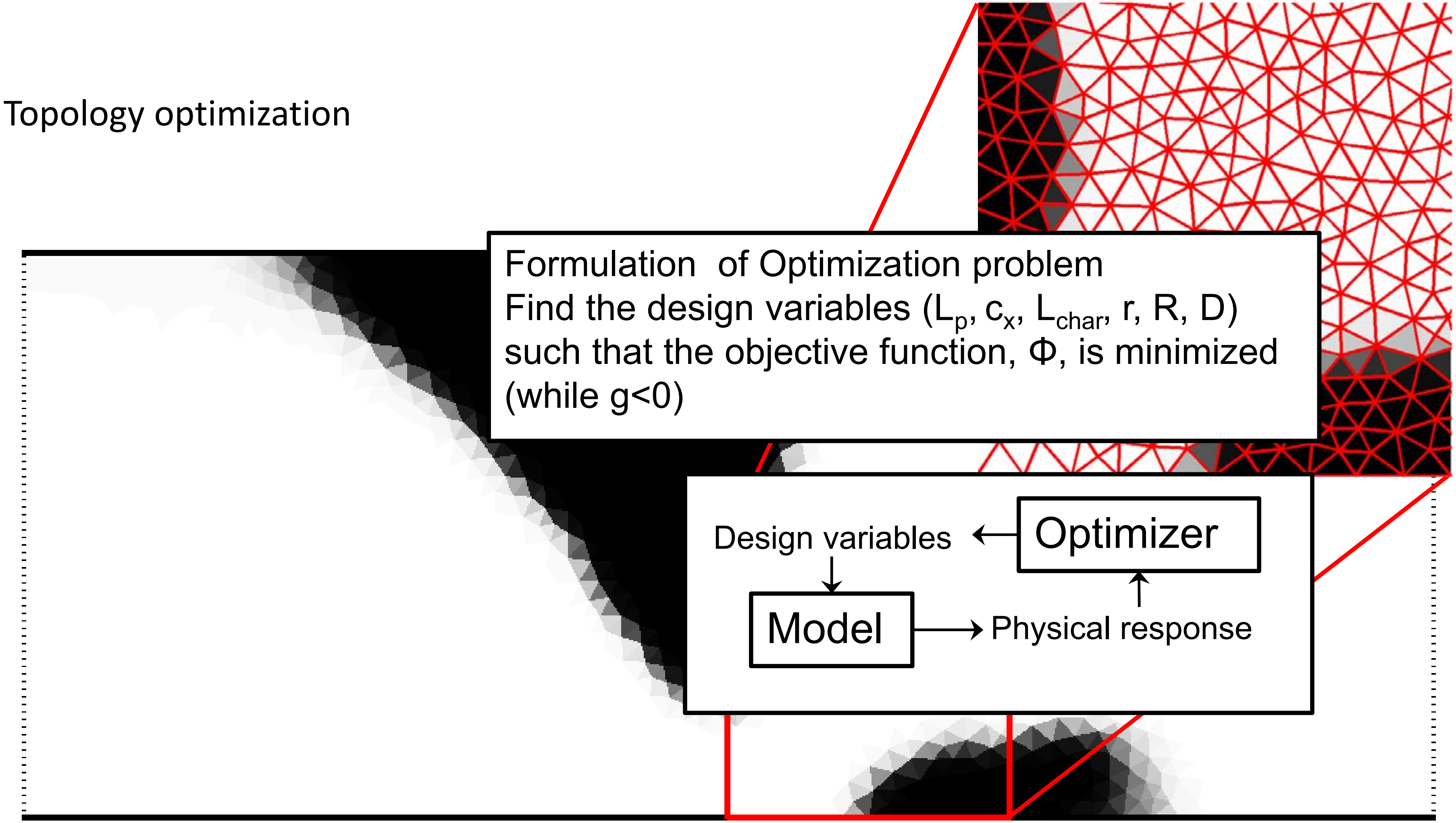
Shape optimization



Basic concept

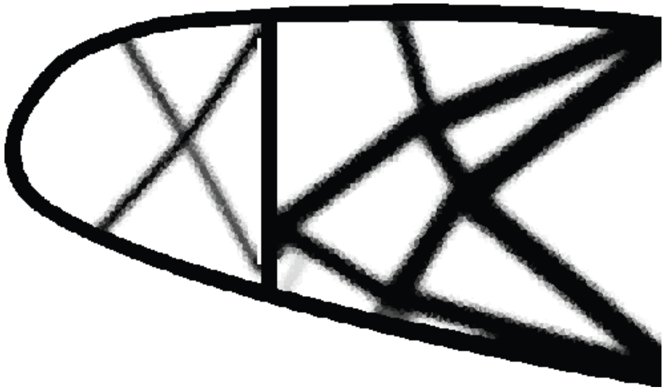
shape versus topology optimization

Topology optimization

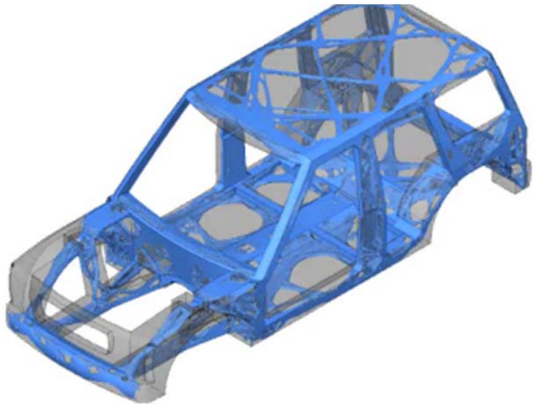


Formulation of Optimization problem
Find the design variables (L_p, c_x, L_{char}, r, R, D) such that the objective function, Φ, is minimized (while g<0)

Aerospace/automotive



DTU TOPOPT(www.topopt.dtu.dk)

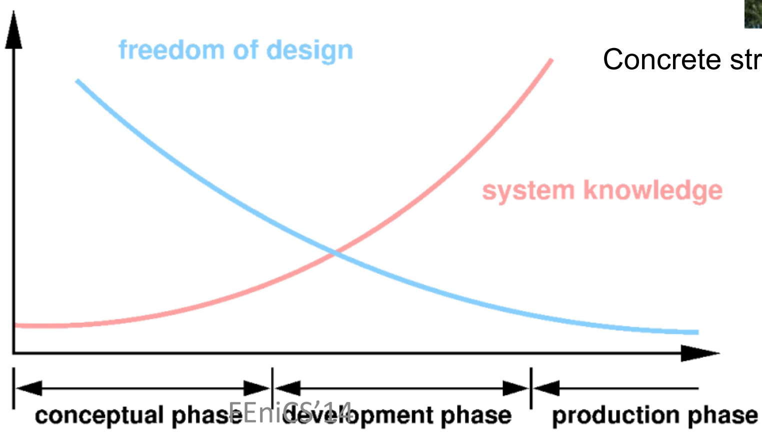


Vehicle optimization (www.youtube.com/watch?v=yztnDAexTHE)

Architecture/design

Bone Chair (designgallerist.com)

Concrete structure (www.digitalcrafting.dk)

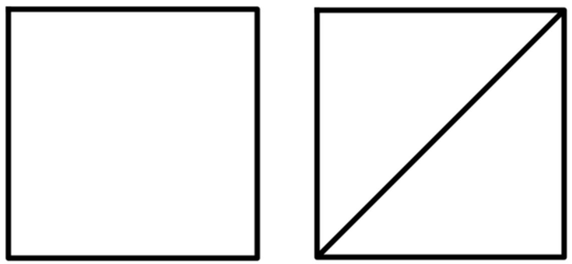


Imperial College London

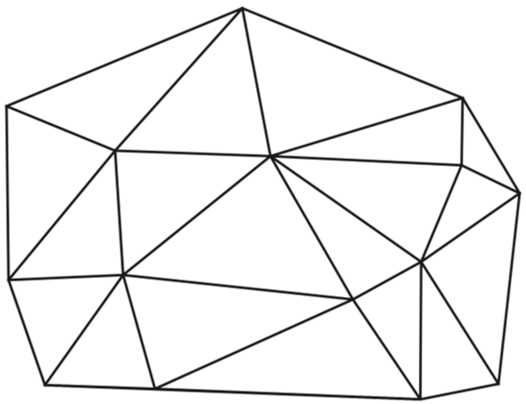
Anisotropic Mesh Generation

Continuous mesh framework

- Structured meshes

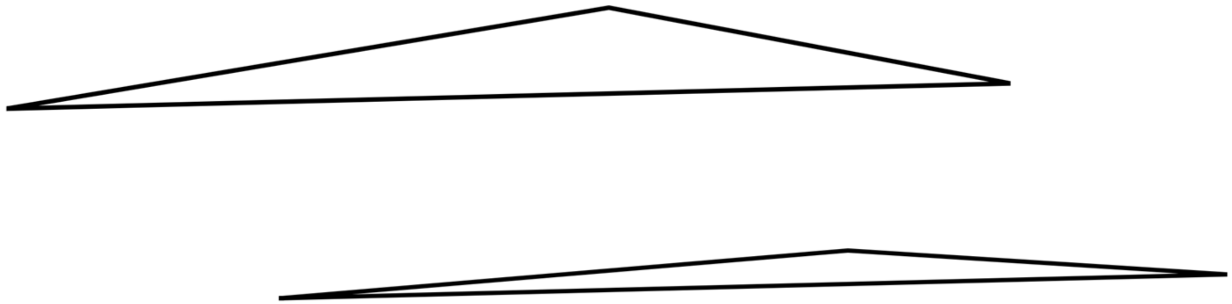


- Unstructured mesh



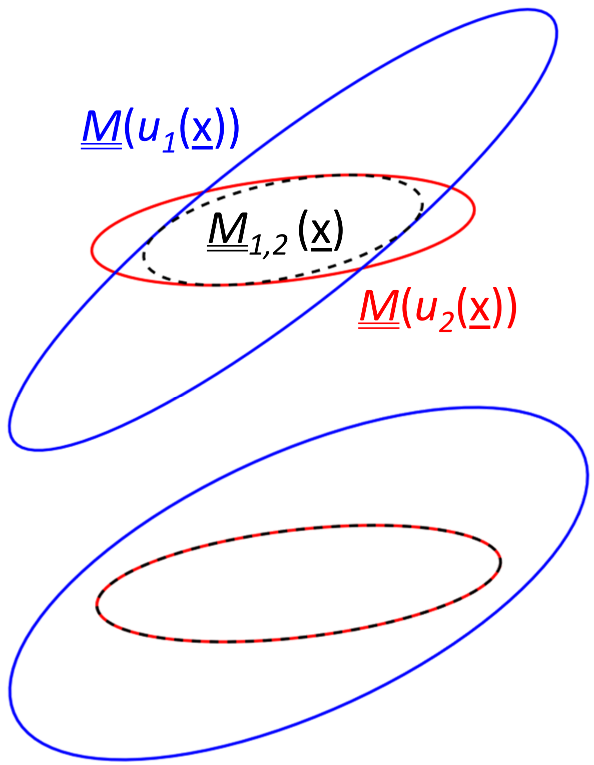
- Isotropic continuous framework: $M(\underline{x})$
- Anisotropic continous framework $\underline{\underline{M}}(\underline{x})$ (SPD)

Optimal 3D Highly Anisotropic Mesh Adaptation Based on the Continuous Mesh Framework, **Adrien Loseille, Frédéric Alauze, Proceedings of the 18th International Meshing Roundtable 2009, 575-594**



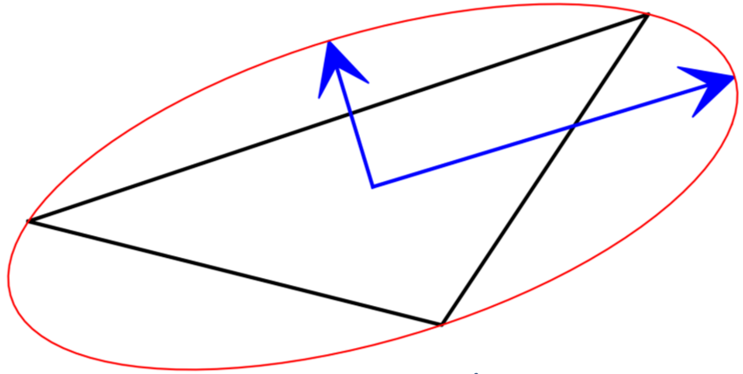
Continuous mesh framework

Inner ellipse



- Isotropic continuous framework: $M(\underline{x})$
- Anisotropic continous framework $\underline{\underline{M}}(\underline{x})$ (SPD)

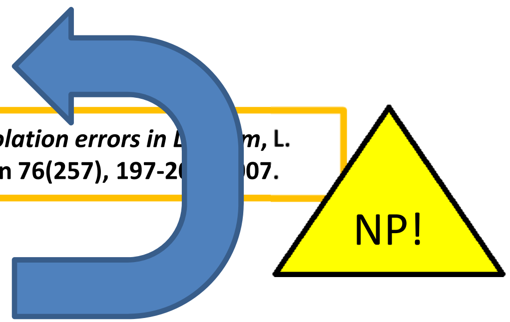
Optimal 3D Highly Anisotropic Mesh Adaptation Based on the Continuous Mesh Framework, **Adrien Loseille, Frédéric Alauze, Proceedings of the 18th International Meshing Roundtable 2009, 575-594**



- $u(\underline{x})$ -> $\underline{\underline{H}}(u(\underline{x}))$ -> $\underline{\underline{M}}(\underline{x})$

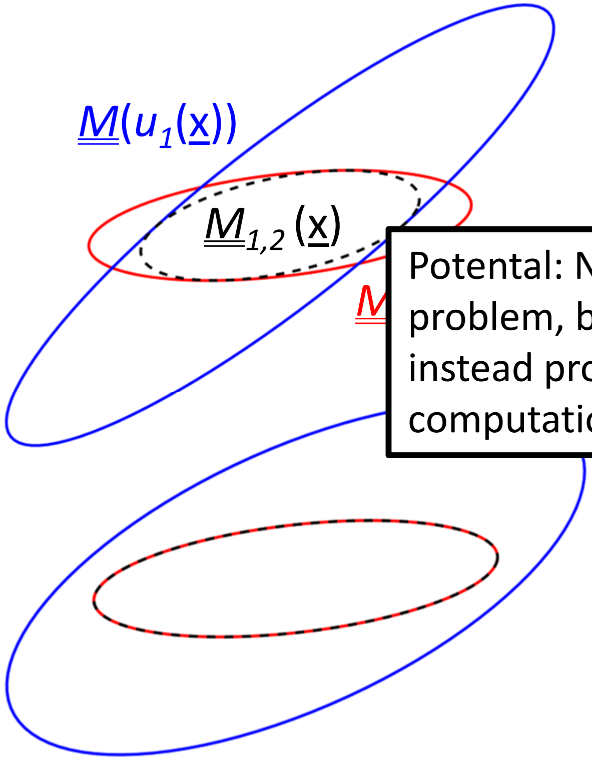
Optimal anisotropic meshes for minimizing interpolation errors in L **m, L. Chen, P. Sun, J. Xu, P, Mathematics of Computation 76(257), 197-2 07.**

- mesh -> $\underline{\underline{M}}_{mesh}(\underline{x})$



Continuous mesh framework

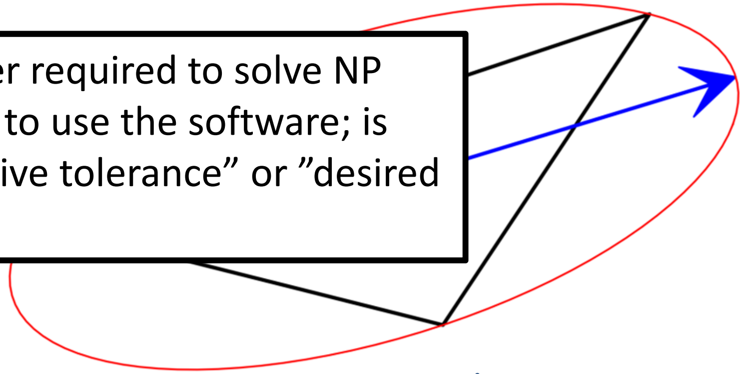
Inner ellipse



- Isotropic continuous framework: $M(\underline{x})$
- Anisotropic continous framework $\underline{\underline{M}}(\underline{x})$ (SPD)

***Optimal 3D Highly Anisotropic Mesh Adaptation Based on the Continuous Mesh Framework,* Adrien Loseille, Frédéric Alauze, Proceedings of the 18th International Meshing Roundtable 2009, 575-594**

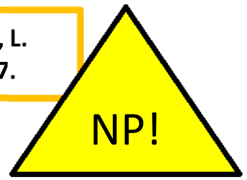
Potental: Novice is no longer required to solve NP problem, before being able to use the software; is instead prompted for ”relative tolerance” or ”desired computational time”



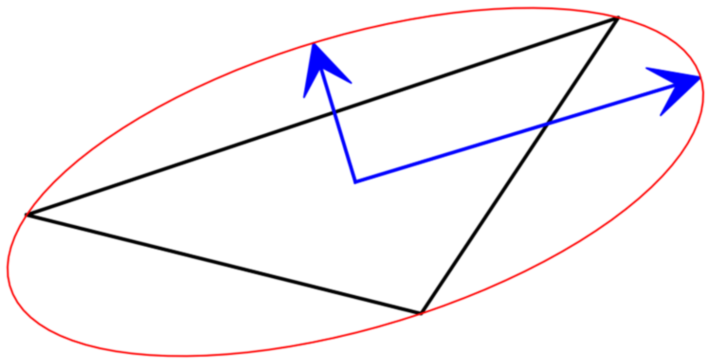
- $u(\underline{x})$ -> $\underline{\underline{H}}(u(\underline{x}))$ -> $\underline{\underline{M}}(\underline{x})$

Optimal anisotropic meshes for minimizing interpolation errors in L **m, L. Chen, P. Sun, J. Xu, P, Mathematics of Computation 76(257), 197-2 07.**

- mesh -> $\underline{\underline{M}}_{mesh}(\underline{x})$



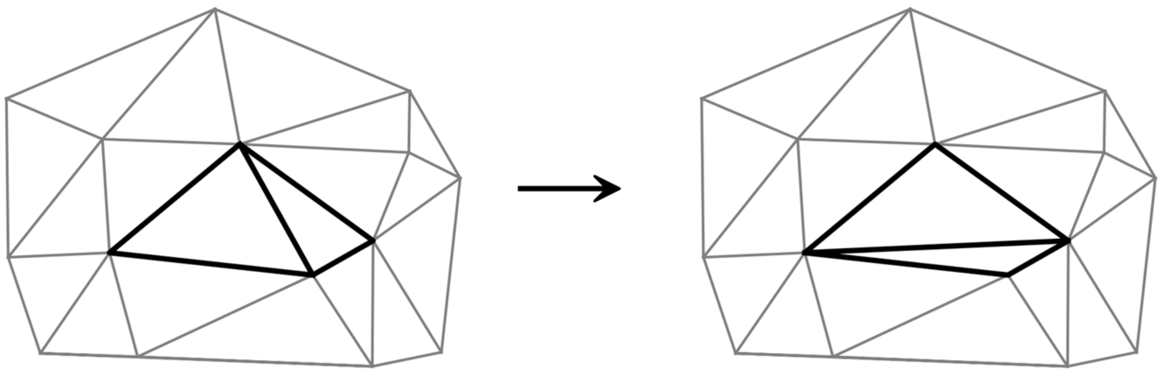
Local operations (2D)



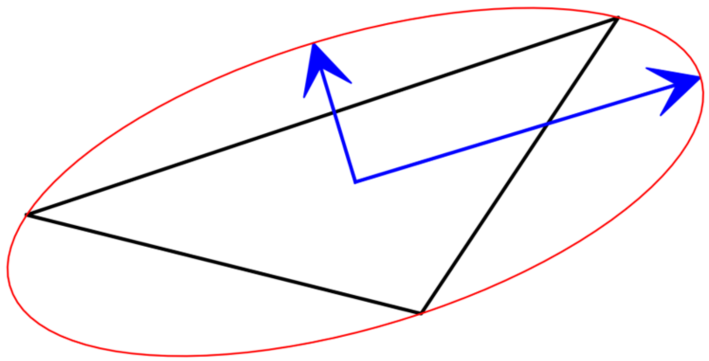
$$q = \frac{\Delta_A}{\sqrt{3}/4}\frac{3^2}{(\Delta_\partial)^2}F(\Delta_\partial/3)^3$$
$$F(x) = \min(x, x^{-1})\,(2 - \min(x, x^{-1}))$$

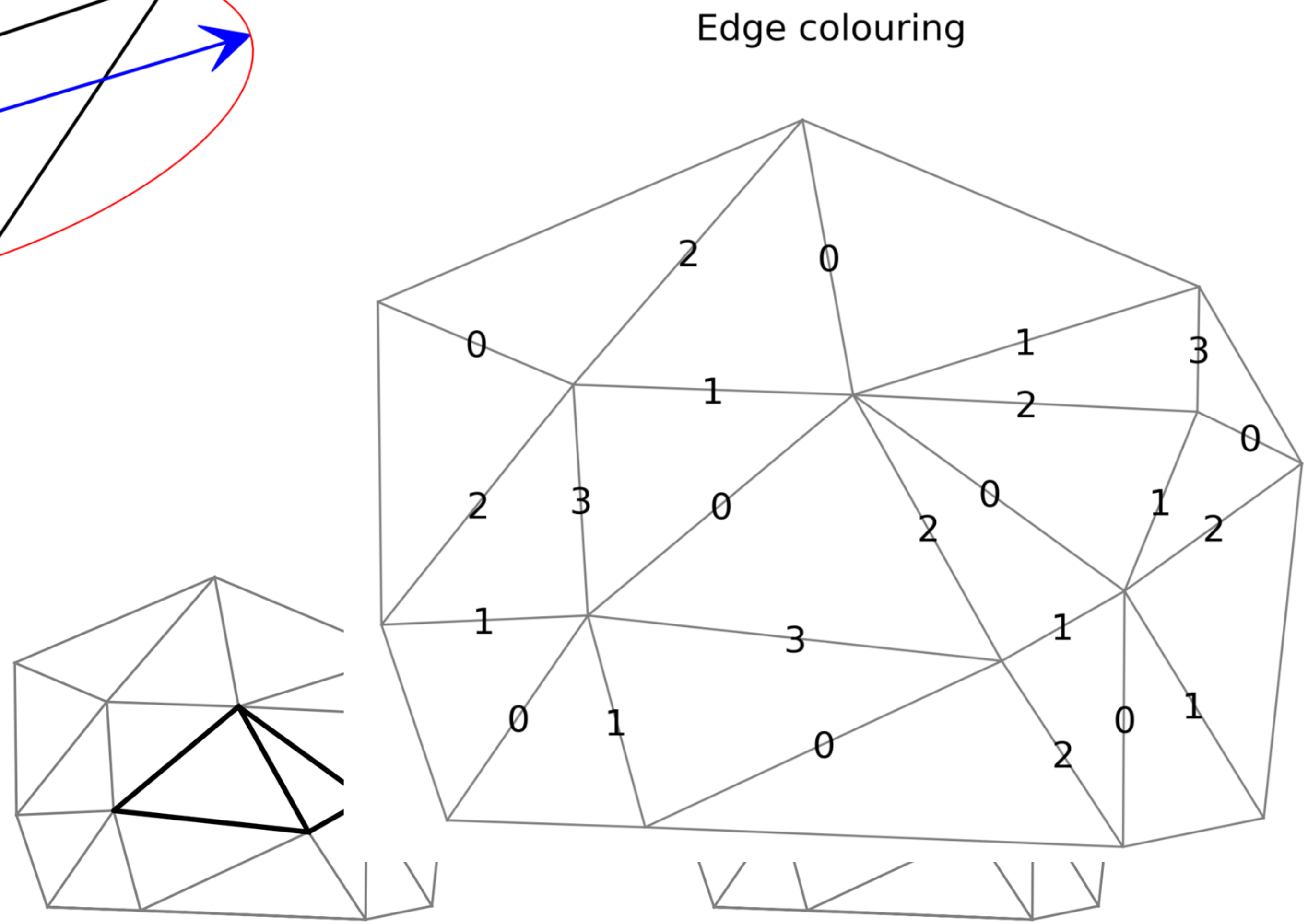
Error bounds for controllable adaptive algorithms based on a hessian recovery, **Yu V. Vasilevski, KN Lipnikov, Computational Mathematics and Mathematical Physics 45(8), 1374-1384, 2005**

Flipping



Local operations (2D)

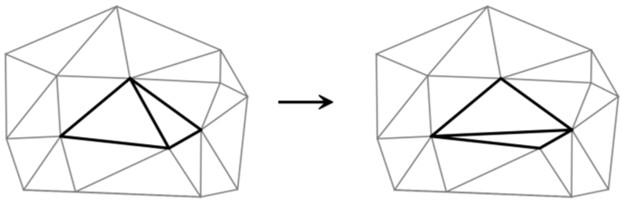




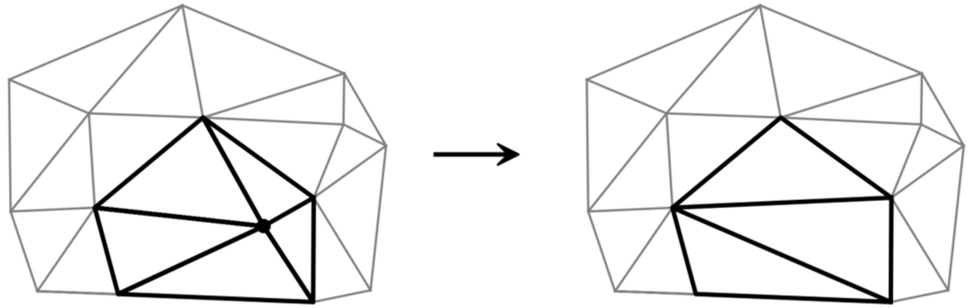
Imperial College London

Local operations (II)

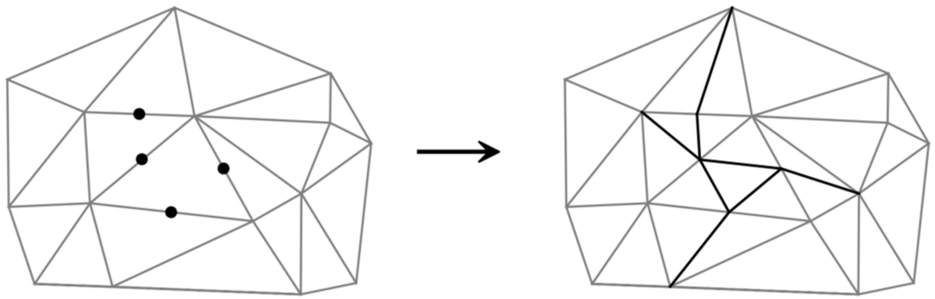
Flipping



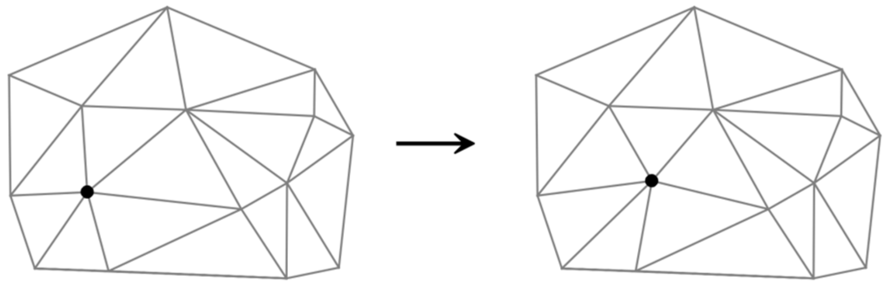
Coarsening



Refinement

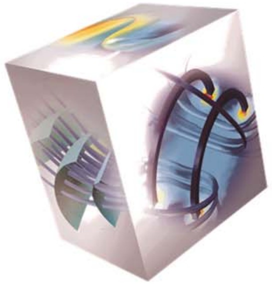


Smoothing



***A thread-parallel algorithm for anisotropic mesh adaptation*, G. Rokos, G.J. Gorman, J. Southern, P.H.J. Kelly, 2014.**

Isotropic benchmark

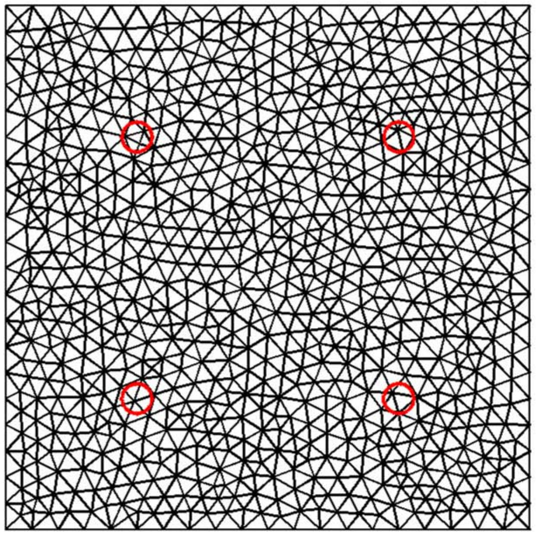


Speed: 10k nodes/second/core (up to ~10 cores)

~1-2% of total computation time (excluding metric computation)

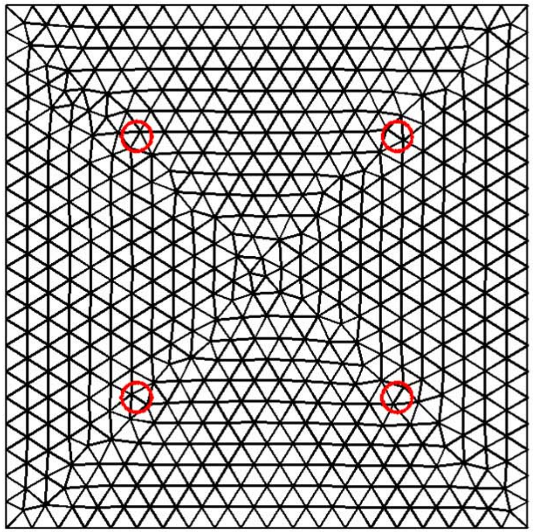
edges between 38% and 114% of target

average edge = 76% of target, std = 12%



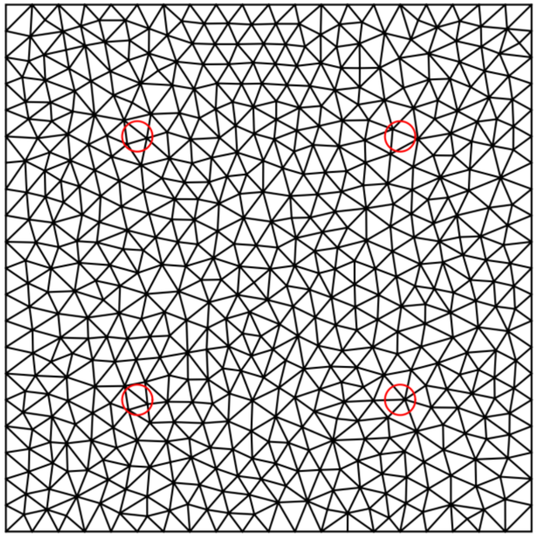
edges between 59% and 112% of target

average edge = 95% of target, std = 6%

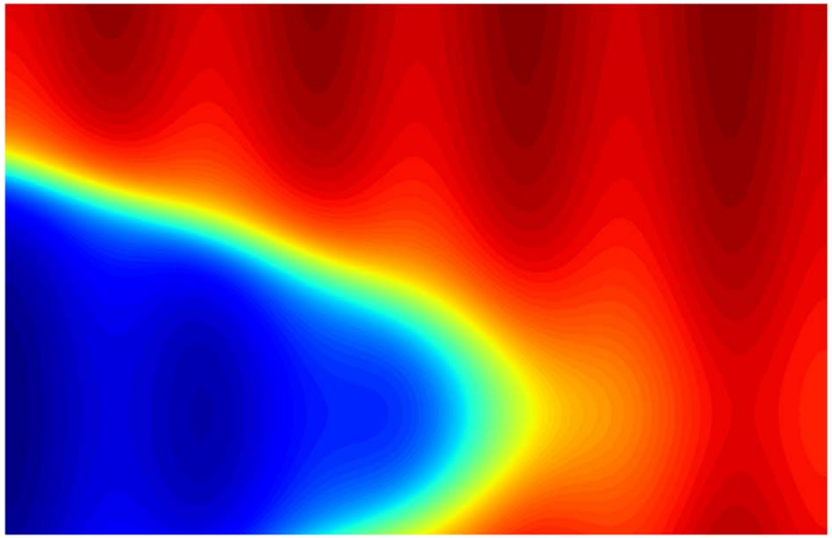


edges between 62% and 141% of the target

average edge = 95% of the target, std = 15%

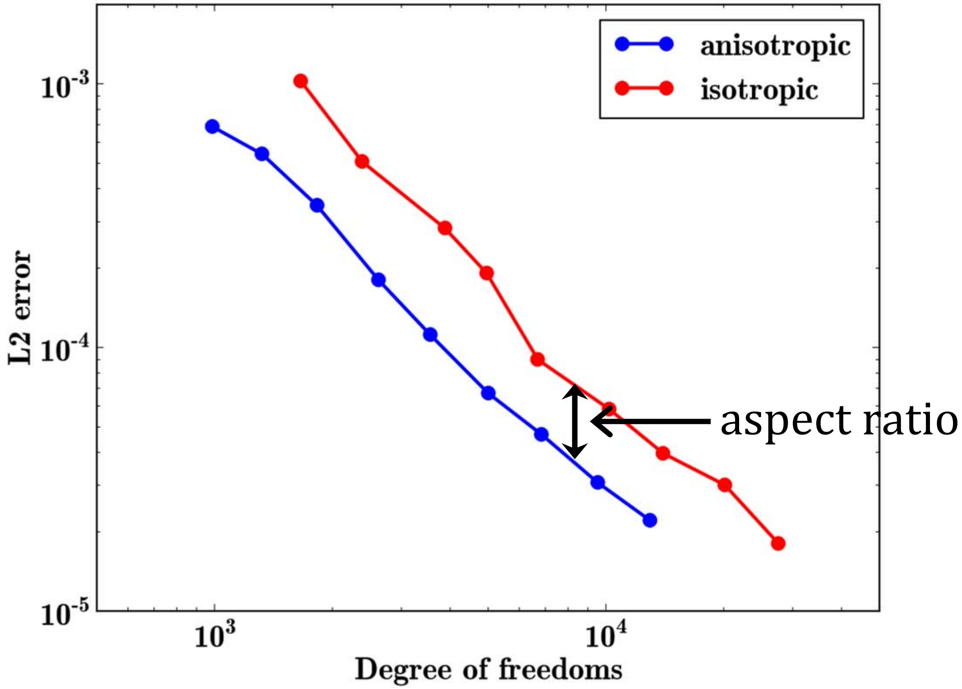


Example



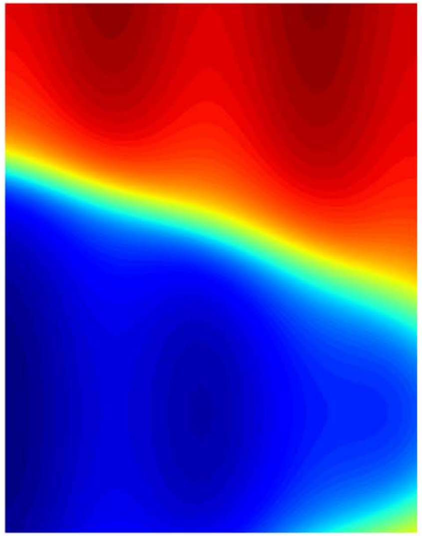
$$\nabla^2 u = f(x)$$

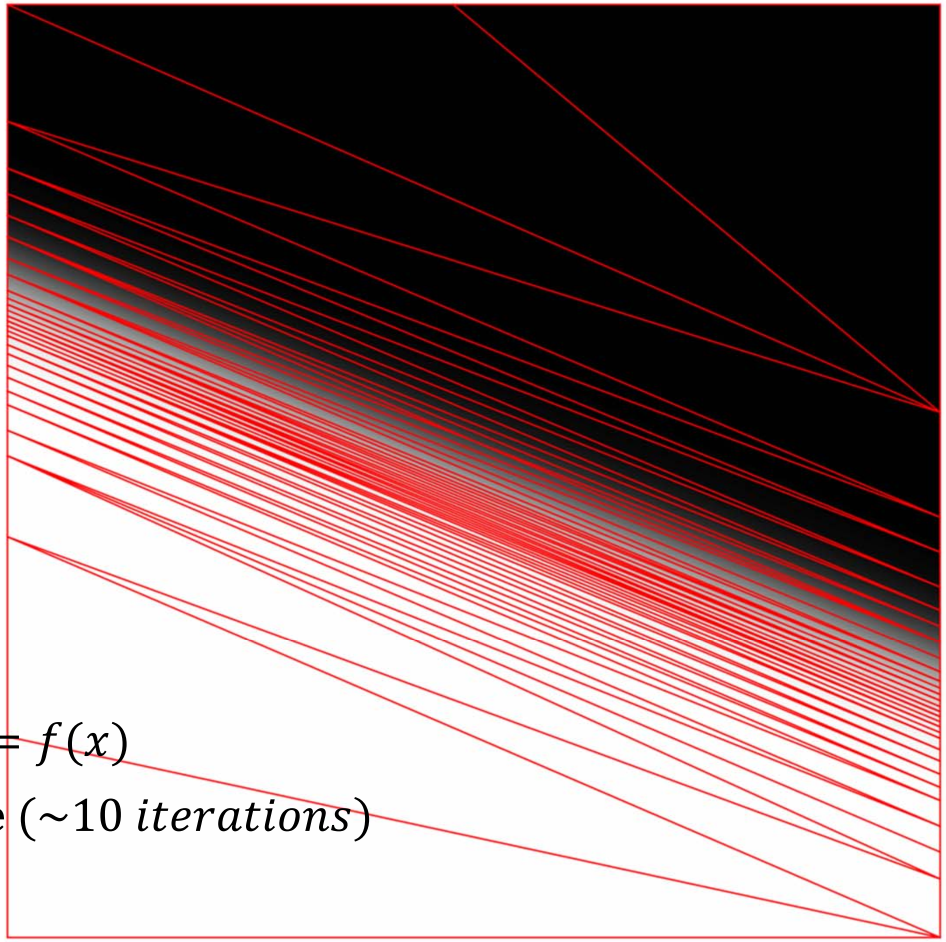
adapt <—> solve ($\sim$10 *iterations*)

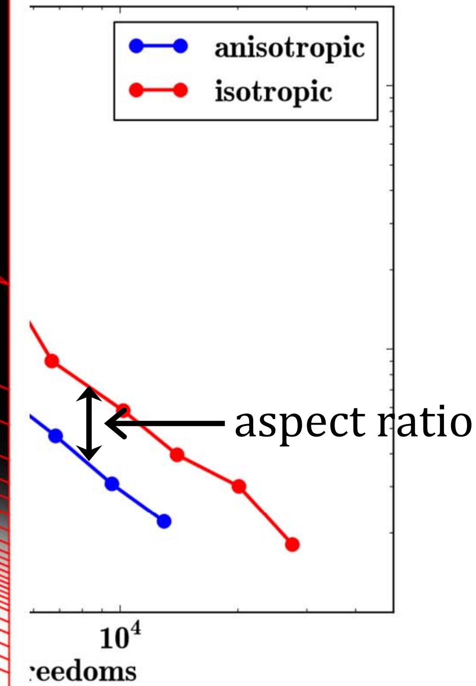


Imperial College London

Example







$\nabla^2 u = f(x)$

adapt <—> solve (~ 10 *iterations*)

Imperial College London

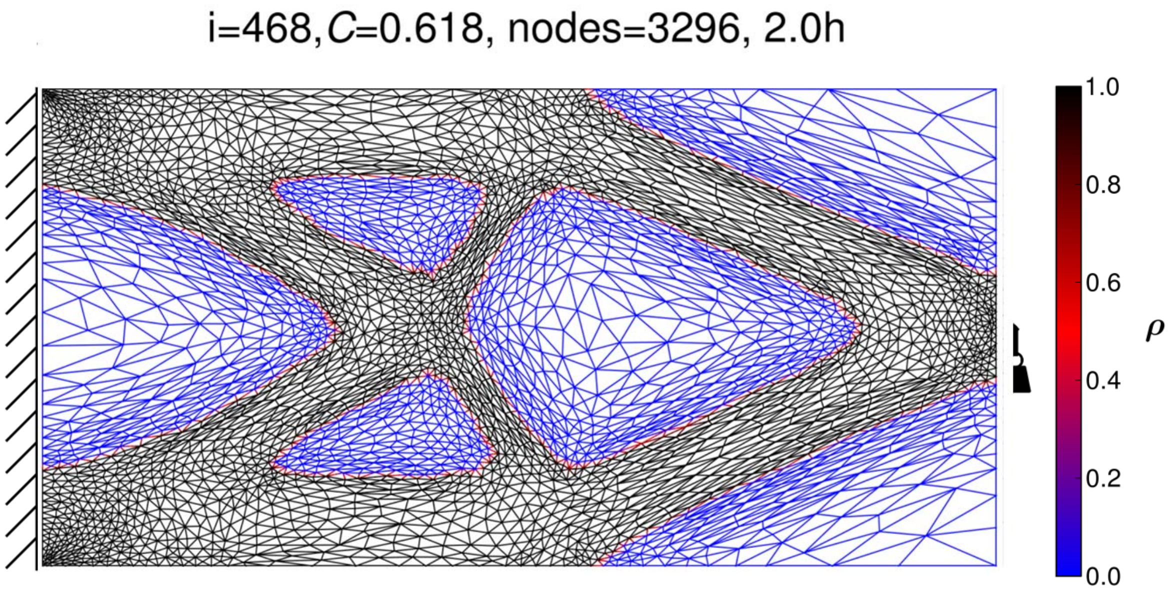
Examples

Imperial College London

Minimum Compliance, designs

$L_x = 2L_{char}$
$L_1 = 0.1L_{char}$
$\rho_{avg} = 0.5$
$L_{min} = 5 \cdot 10^{-2}L_{char}$
Linear elasticity, $E = \rho^P$

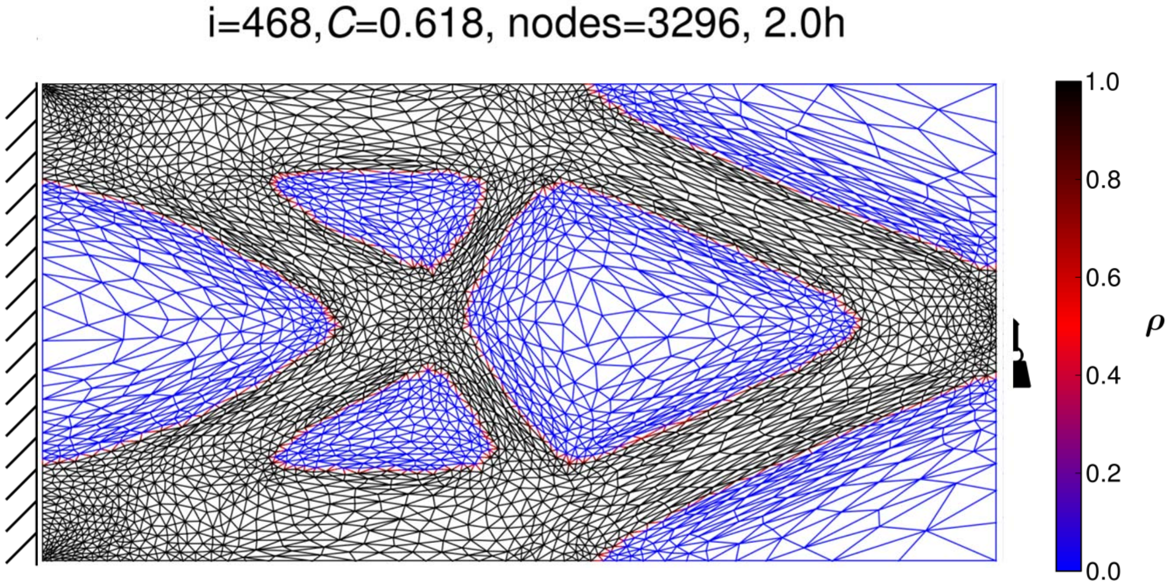
- Design and sensitivity drives adaptation
- Interpolation of internal optimiser variables

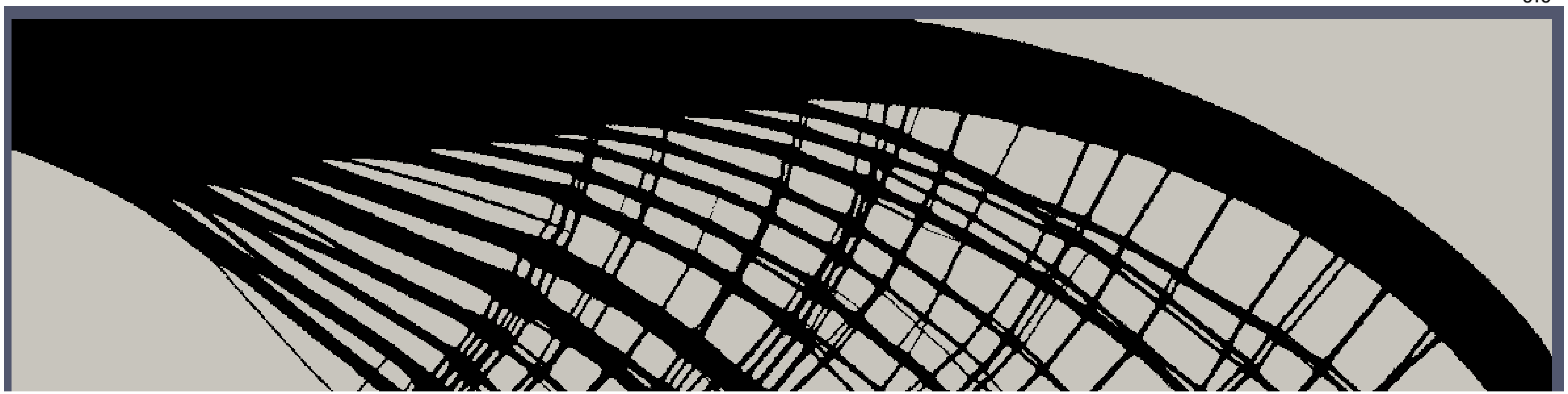


Imperial College London

Minimum Compliance, designs

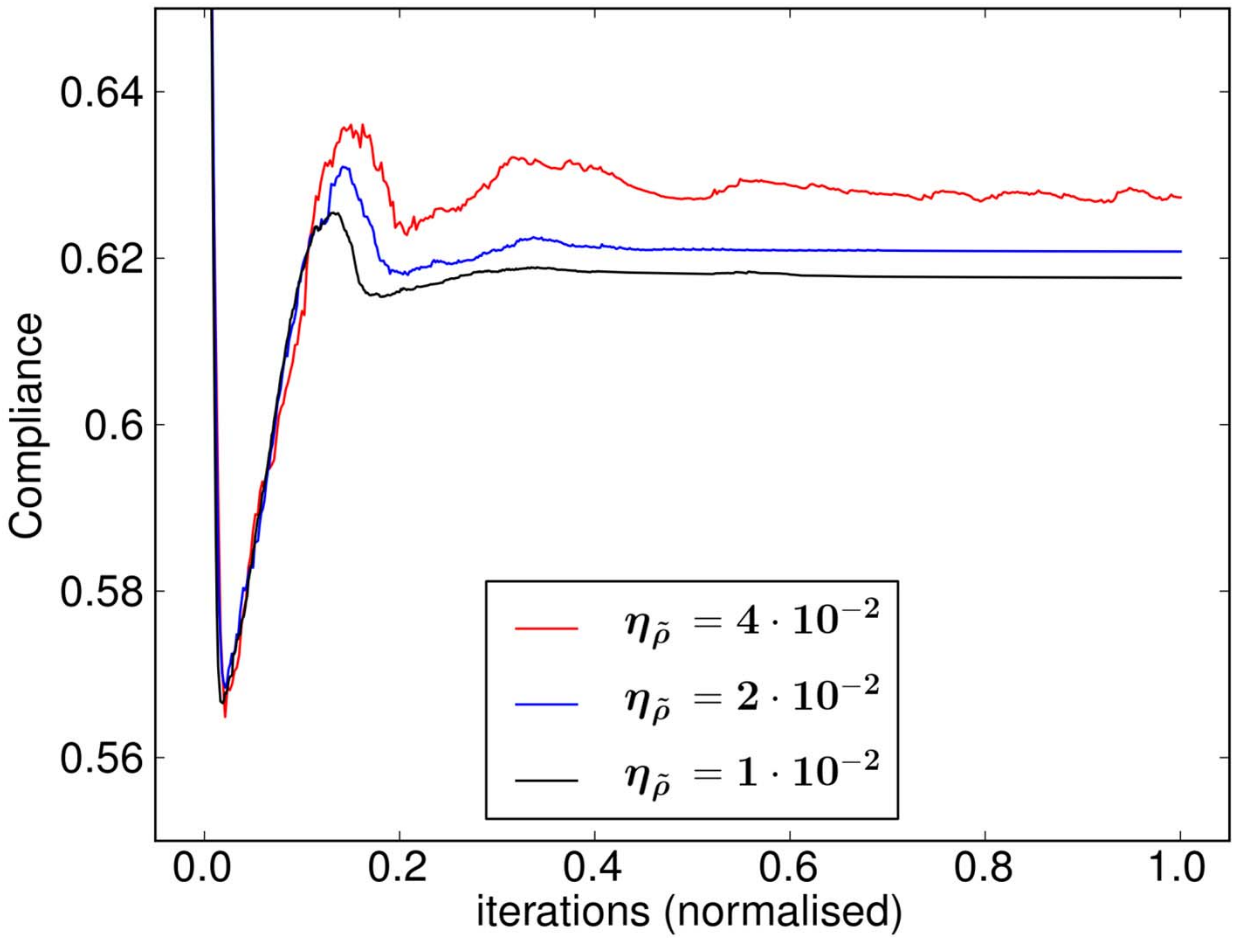
$L_x = 2L_{char}$
$L_1 = 0.1L_{char}$
$\rho_{avg} = 0.5$
$L_{min} = 5 \cdot 10^{-2} L_{char}$
Linear elasticity, $E = \rho^P$





Imperial College London

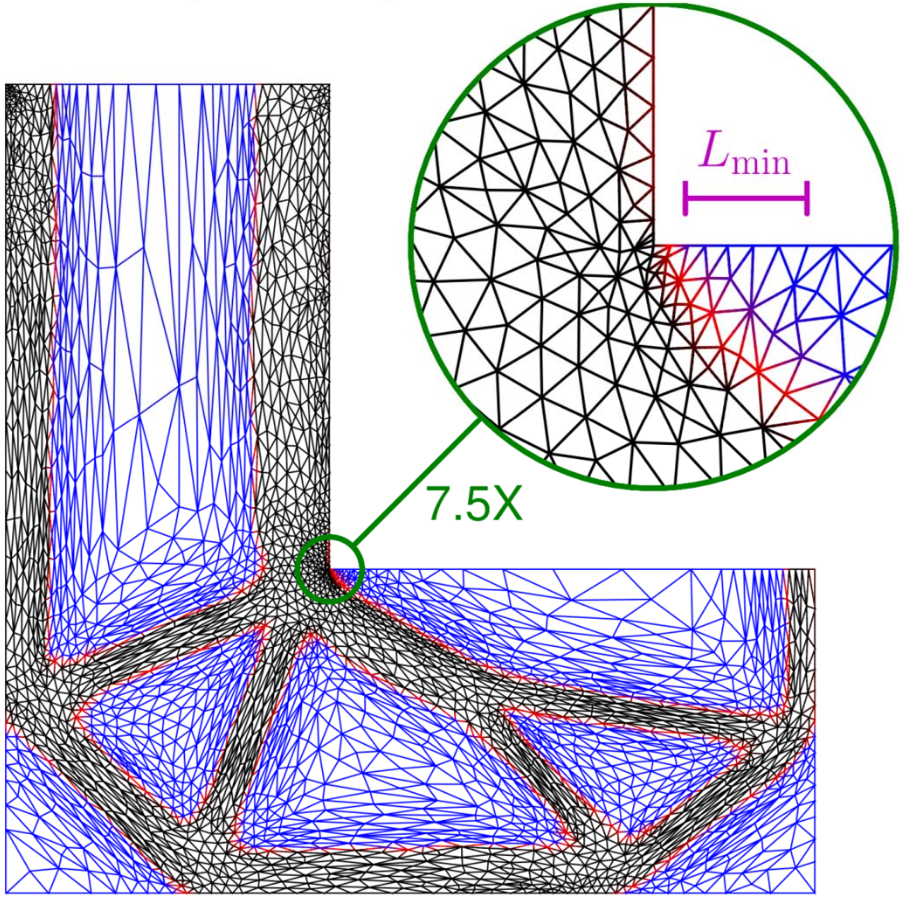
Minimum Compliance, convergence



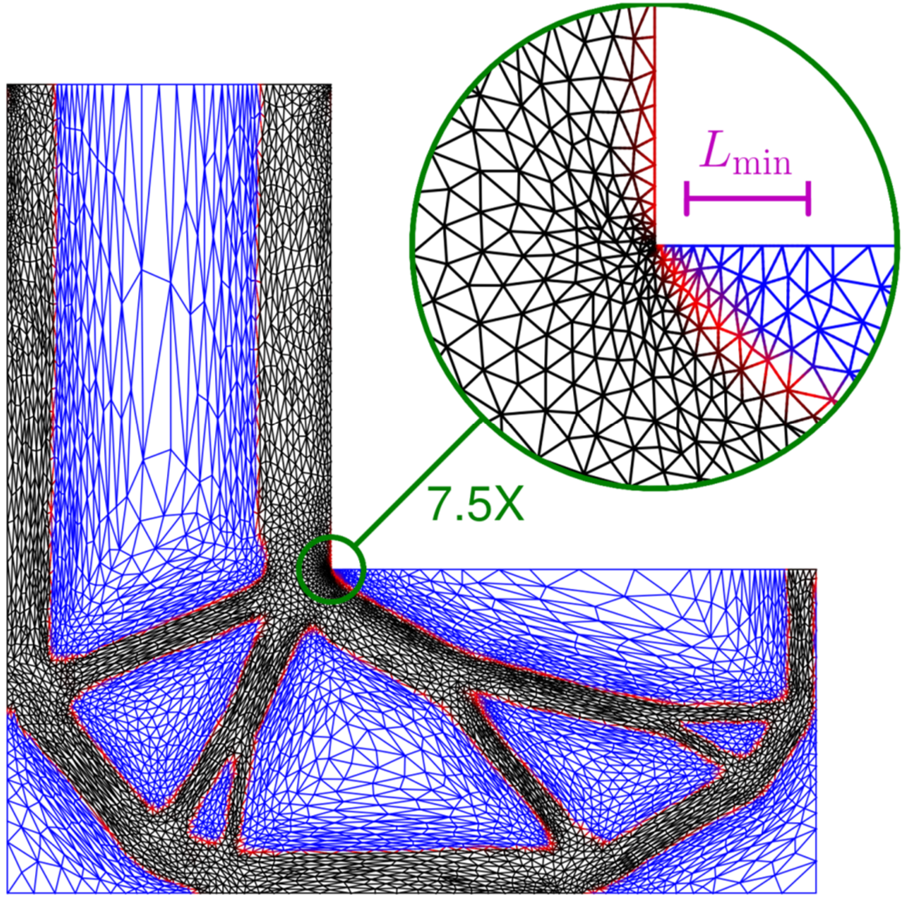
Imperial College London

Stress constraint, designs

i=388, V=0.374, nodes=2472, 3.2h

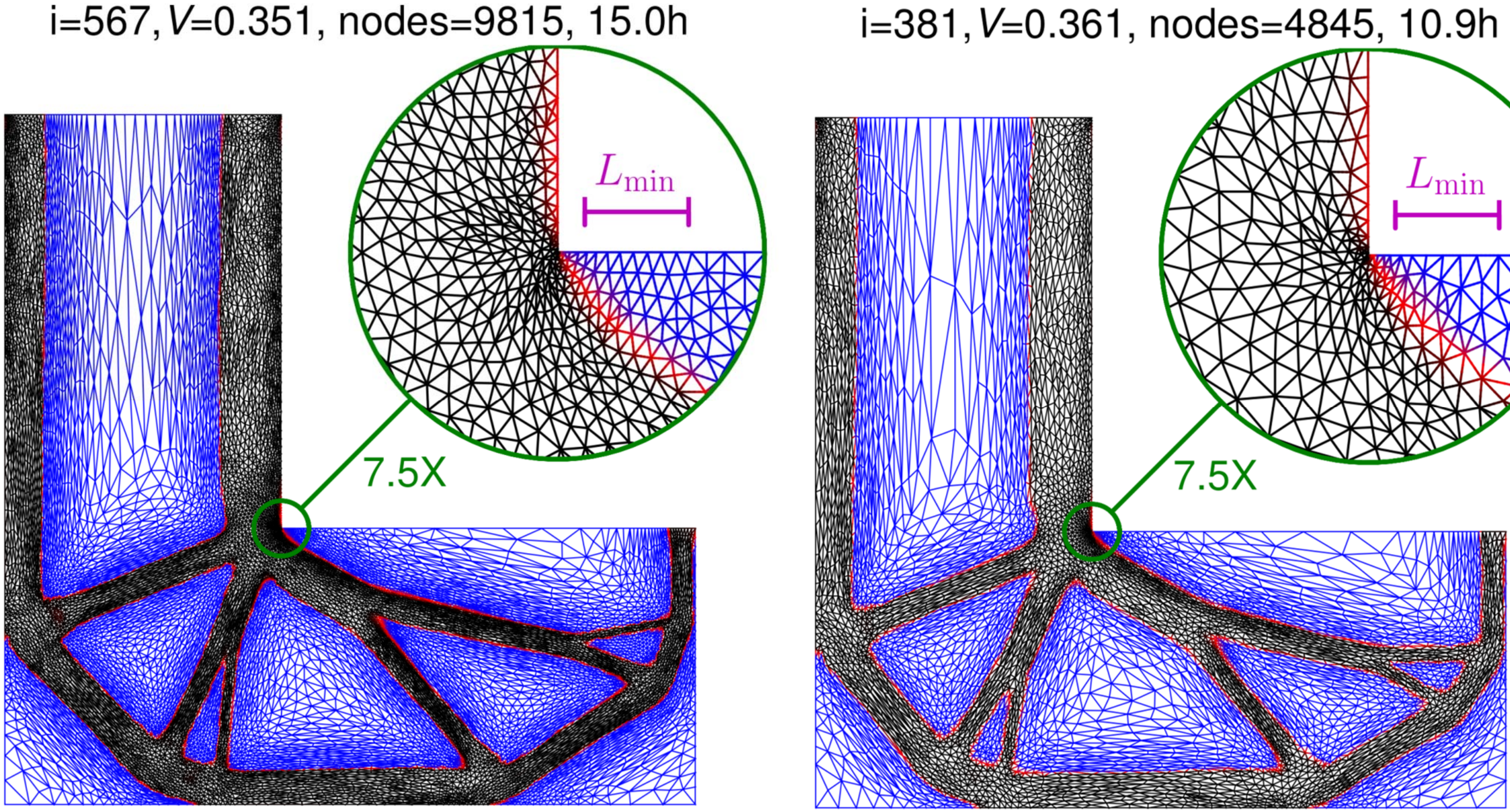


i=381, V=0.361, nodes=4845, 10.9h



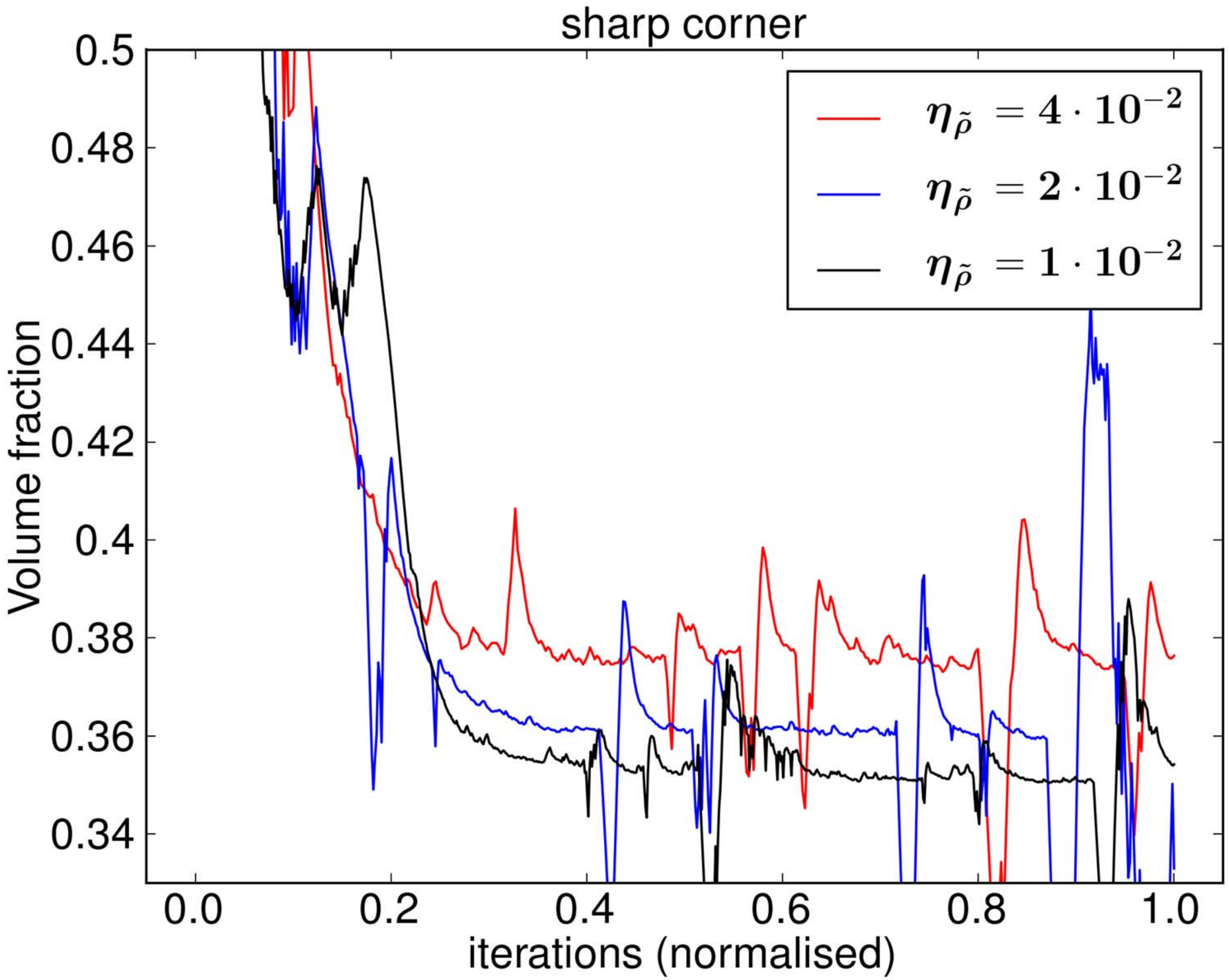
- Relaxed integral constraint $\left[\int_\Omega (\sigma_{\text{miss}})^q \partial\Omega\right]^{\frac{1}{q}}, < \sigma_{\max}$
- Volume is minimized

Stress constraint, designs



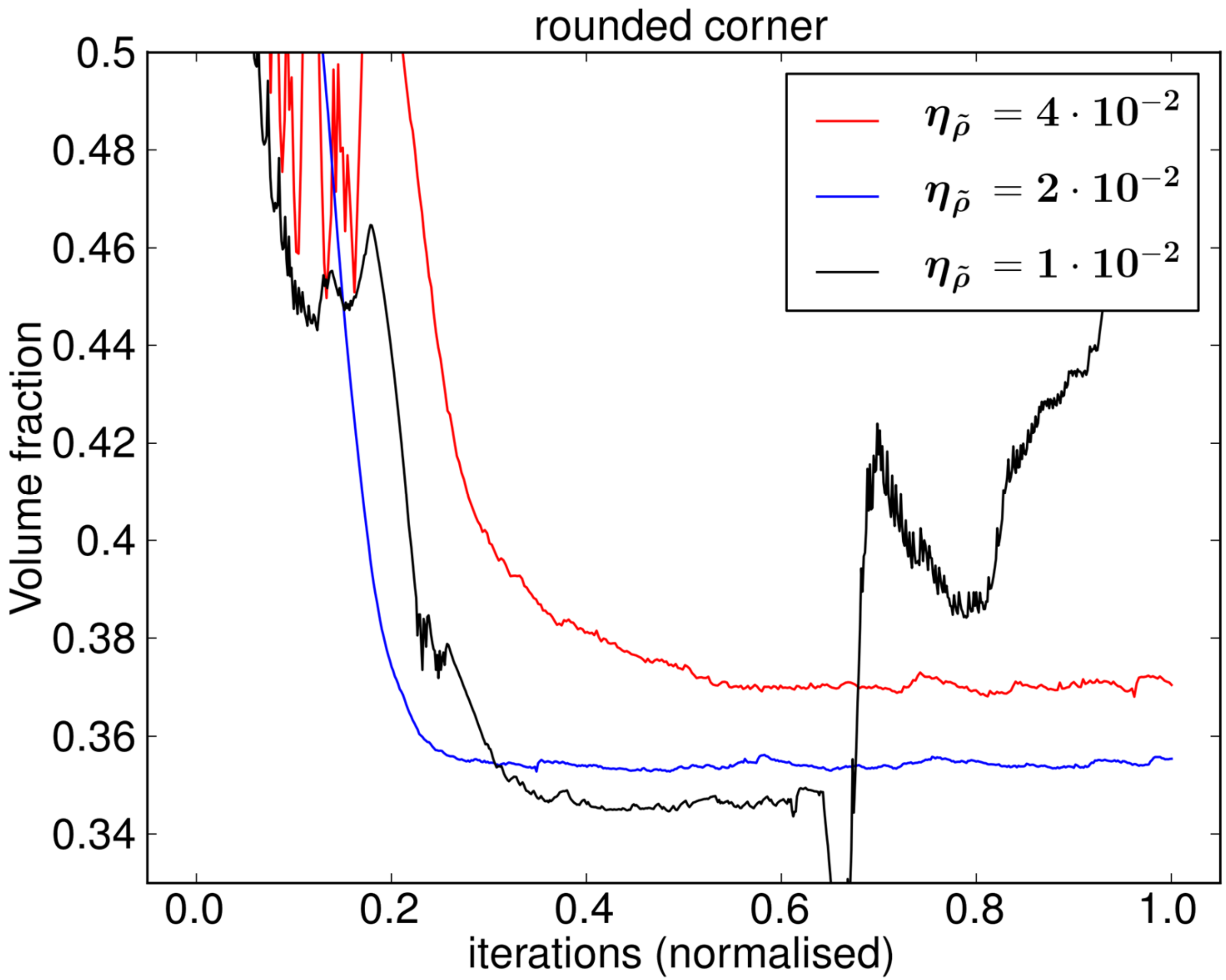
- Relaxed integral constraint $\left[\int_\Omega (\sigma_{\text{miss}})^q \partial\Omega\right]^{\frac{1}{q}}, < \sigma_{\max}$
- Volume is minimized

Stress constraint, convergence

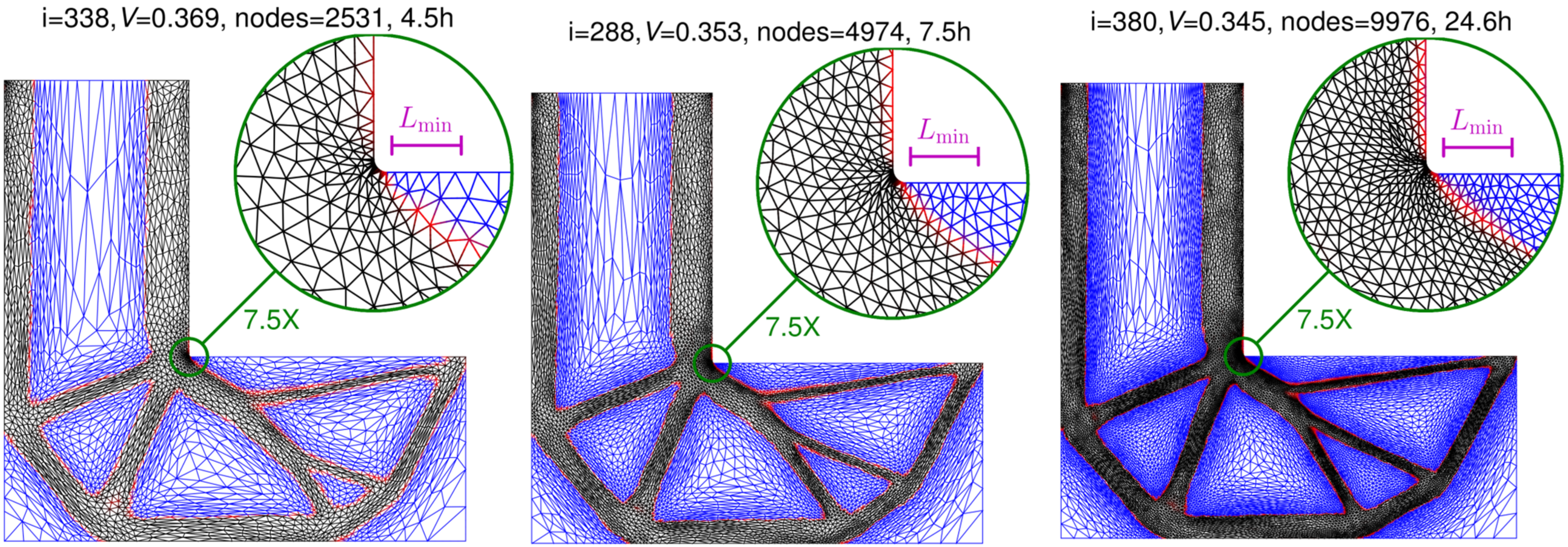


Imperial College London

Stress constraint, convergence (II)



Stress constraint, designs (II)



C^1 continuity at the corner might be important

Imperial College London

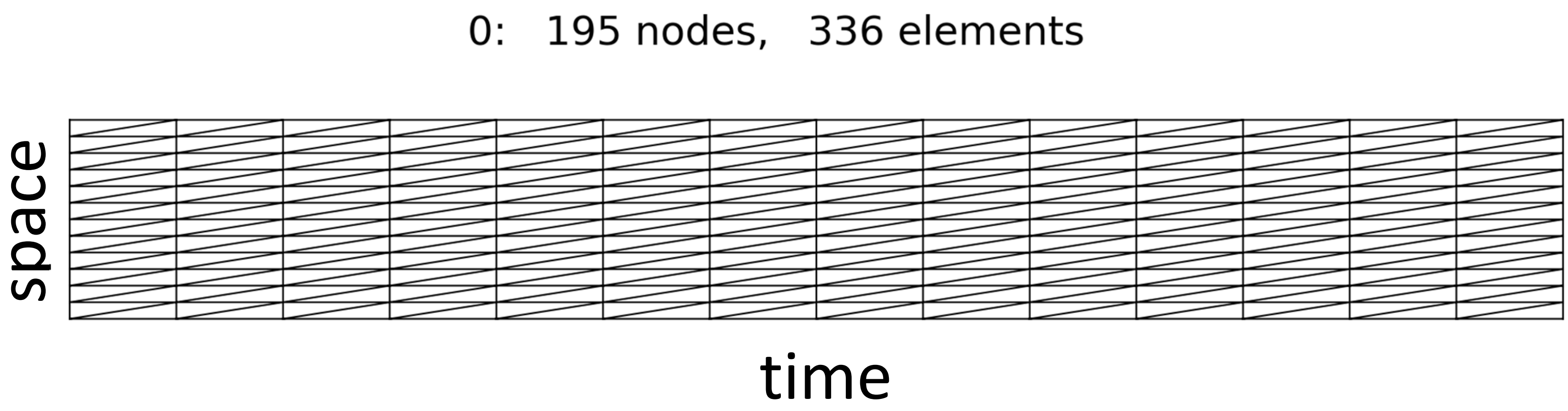
Parallelisation or adaptivity? No!

- time = money
  - time to solution is the relevant metric
    - thus, strong scaling is important
      - But this always top out
        - » So minimizing the number of degrees of freedom is always critical

- Domain decomposition is ignorant of physics
  - Stiff domains may appear
    - destroying load balancing and thus scaling

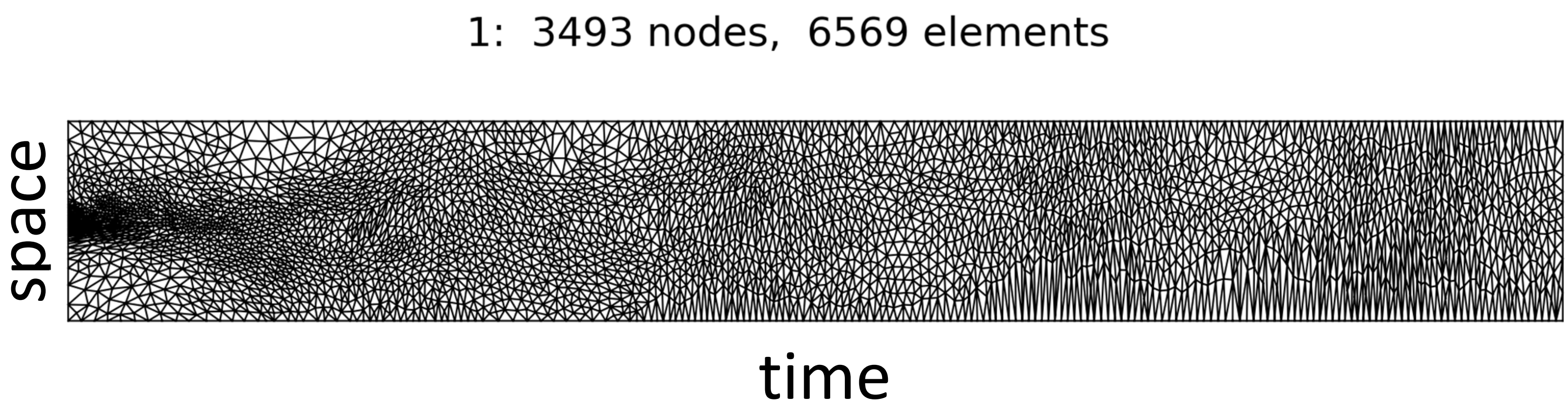
Future work: time-space elements

- The idea of using DOF optimally is incompatible with current time-stepping/optimisation algorithms.
- Using space-time elements for optimisation requires
  - Continuous adjoints
  - Continuous optimisation algorithm
  - 4D anisotropic mesh generator for 3D problems



Future work: time-space elements

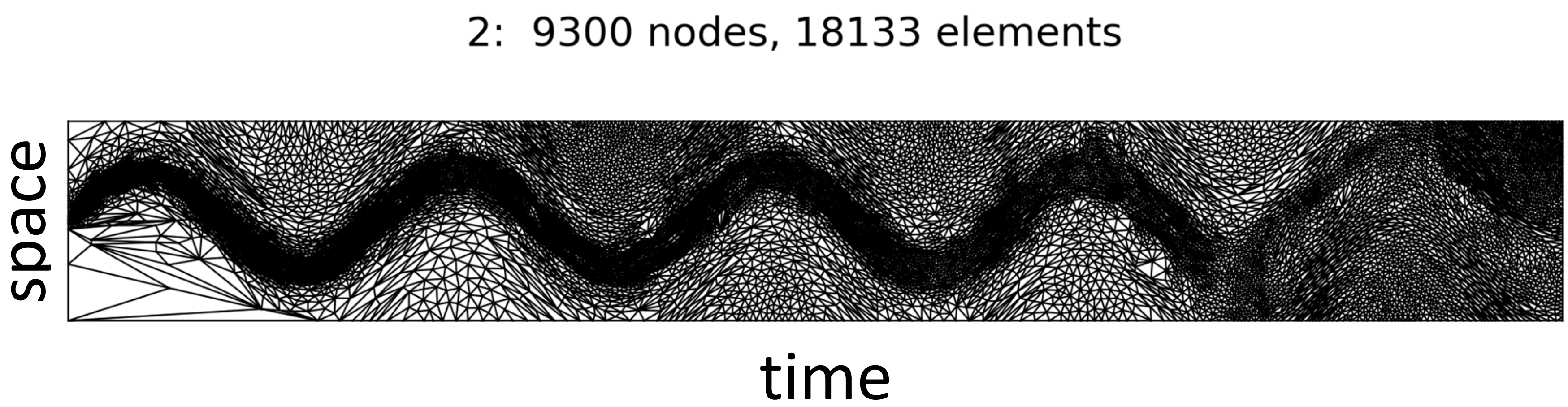
- The idea of using DOF optimally is incompatible with current time-stepping/optimisation algorithms.
- Using space-time elements for optimisation requires
  - Continuous adjoints
  - Continuous optimisation algorithm
  - 4D anisotropic mesh generator for 3D problems



Imperial College London

Future work: time-space elements

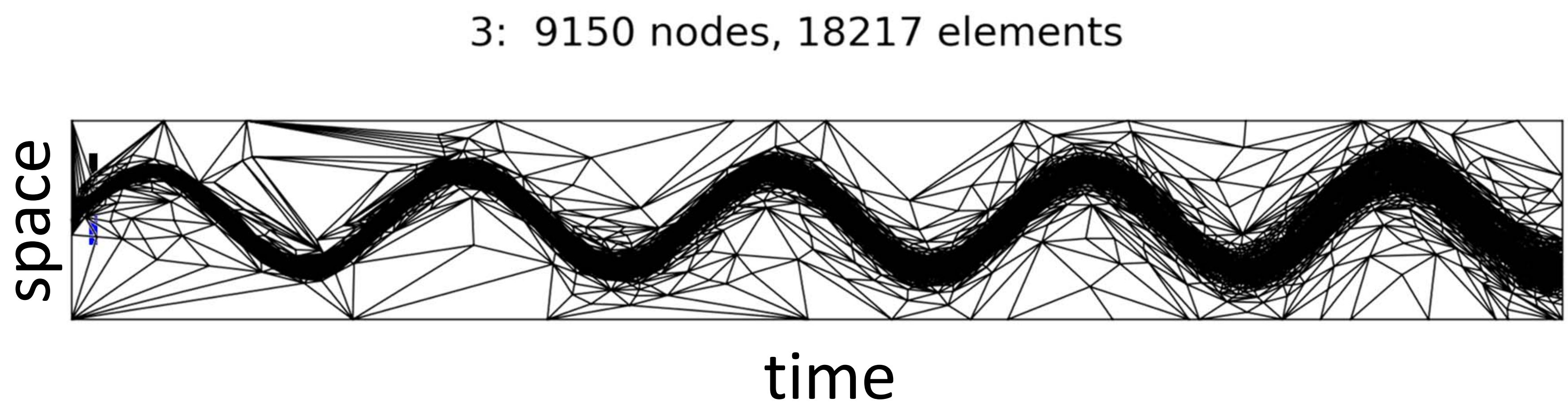
- The idea of using DOF optimally is incompatible with current time-stepping/optimisation algorithms.
- Using space-time elements for optimisation requires
  - Continuous adjoints
  - Continuous optimisation algorithm
  - 4D anisotropic mesh generator for 3D problems



Imperial College London

Future work: time-space elements

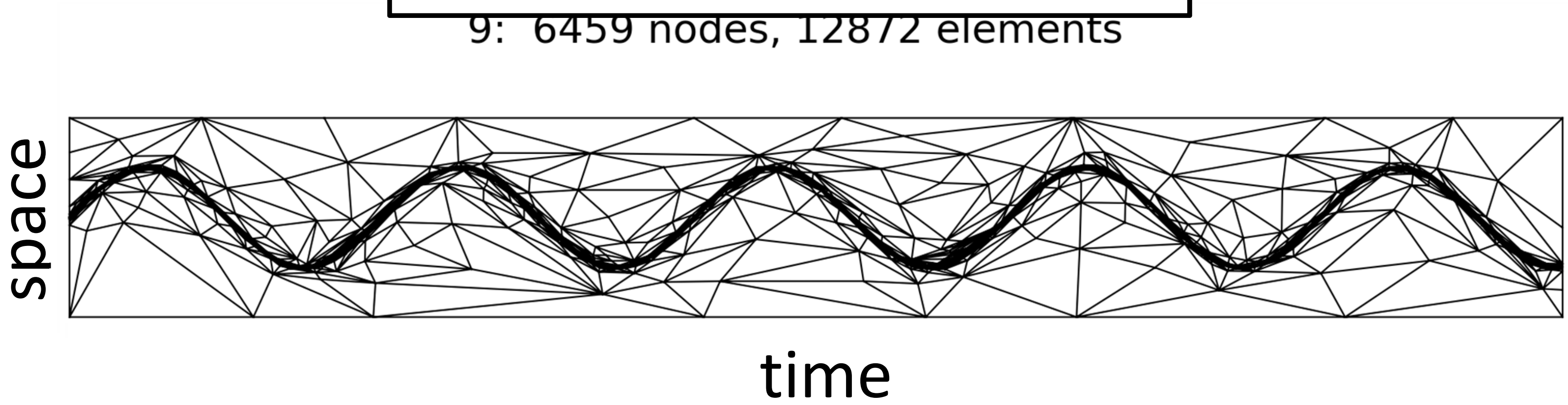
- The idea of using DOF optimally is incompatible with current time-stepping/optimisation algorithms.
- Using space-time elements for optimisation requires
  - Continuous adjoints
  - Continuous optimisation algorithm
  - 4D anisotropic mesh generator for 3D problems



Future work: time-space elements

- The idea of using DOF optimally is incompatible with current time-stepping/optimisation algorithms.
- Using space-time elements for optimisation requires
  - Continuous adjoints
  - Continuous optimisation algorithm
  - 4D anisotro

We want 4D FEniCS



Imperial College London

Syntax

Single scalar variable

```
from dolfin import *
from adaptivity import adapt
…
solve(a == L, u, bc)
mesh = adapt(u) #eta=0.01
```

Ellipse method

```
from dolfin import *
from adaptivity import adapt, metric_pnorm
from adaptivity import metric_ellipse
…
solve(a == L, u, bc1)
solve(b == M, v, bc2)
H1 = metric_pnorm(u, eta=0.02, p=3)
H2 = metric_pnorm(v) #eta=0.01, p=2
H = metric_ellipse(H1,H2)
mesh = adapt(H)
```

But: Simple boundary representation

```
from adaptivity import polygon_surfmesh
…
[bfaces,bfaces_IDs] = \
polyhedron_surfmesh(mesh)
…
mesh = adapt(u, bfaces=bfaces, \
bfaces_IDs=bfaces_IDs)
```

https://github.com/ggorman/pragmatic

Imperial College London

Syntax

Priorities:

1. Robustness
2. 3D

able

```
from dolfin import *
from adaptivity import adapt
…
solve(a == L, u, bc)
mesh = adapt(u) #eta=0.01
```

Ellipse method

```
from dolfin import *
from adaptivity import adapt, metric_pnorm
from adaptivity import metric_ellipse
…
solve(a == L, u, bc1)
solve(b == M, v, bc2)
H1 = metric_pnorm(u, eta=0.02, p=3)
H2 = metric_pnorm(v) #eta=0.01, p=2
H = metric_ellipse(H1,H2)
mesh = adapt(H)
```

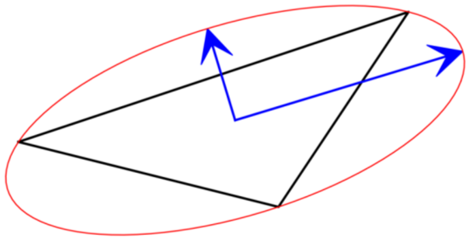
But: Simple boundary representation

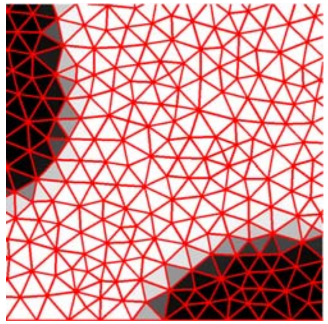
```
from adaptivity import polygon_surfmesh
…
[bfaces,bfaces_IDs] = \
polyhedron_surfmesh(mesh)
…
mesh = adapt(u, bfaces=bfaces, \
bfaces_IDs=bfaces_IDs)
```

https://github.com/ggorman/pragmatic

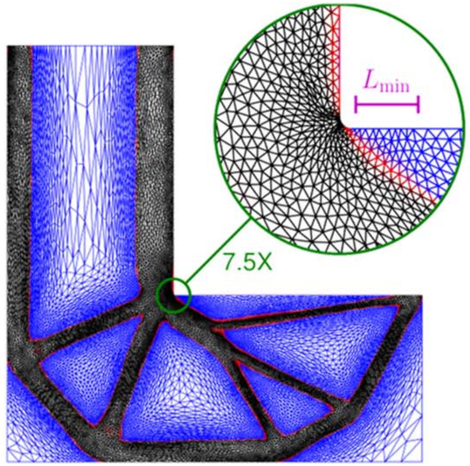
Summary

- Introduction
  - Anisotropic mesh adaptation
  - Topology optimisation

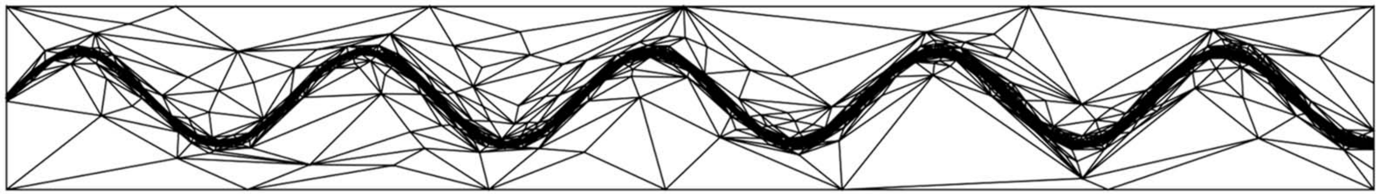




- The combination



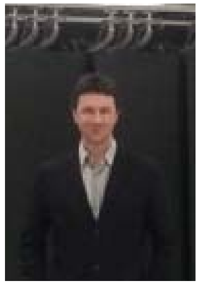
- Time-space elements



Imperial College London

Acknowledgements

- **Dr. Gerard Gorman**
- **Dr. Simon W. Funke**
- **Prof. Christopher Pain**
- **Prof. Matthew Jackson**



Imperial College London

Questions

?