Displacement based polytopal elements

a strain smoothing and scaled boundary approach

POEMS2019

Stéphane P.A. BORDAS & Sundararajan NATARAJAN
and many colleagues ;-)

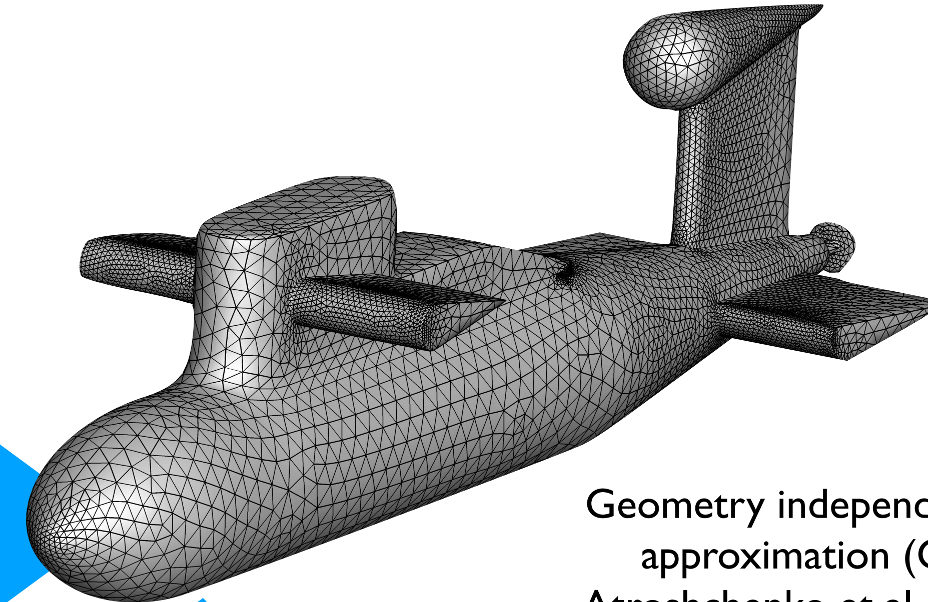
Slides can be downloaded here
https://orbilu.uni.lu/handle/10993/37921

Stéphane P.A. BORDAS, University of Luxembourg and Cardiff University
Sundararajan NATARAJAN, IIT Madras, Chennai, India
Marseille 20190503 https://conferences.cirm-math.fr/1954.html

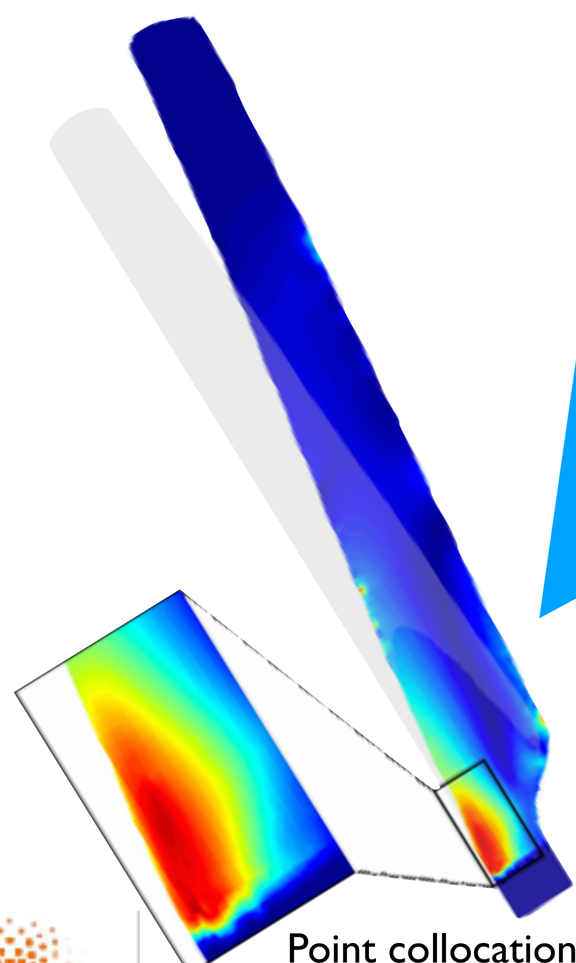
IntuiSIM

legato-team.eu

Intuitive modelling & SIMulation



Geometry independent field approximation (GIFT)
Atroshchenko et al. CMAME 2018



Point collocation methods
Jacquemin et al. 2019
(Archives)

Implicit boundaries for real time adaptive simulations (IEEE, IJNMBE, AMM, 2017-2019)

One machine, one minute, three billion tetrahedra

Célestin Marot* | Jeanne Pellerin | Jean-François Remacle

[1]Université catholique de Louvain, iMMC, Avenue Georges Lemaitre 4, bte L4.05.02, 1348 Louvain-la-Neuve, Belgium

Correspondence
*Corresponding author: Email: celestin.marot@uclouvain.be

Summary

This paper presents a new scalable parallelization scheme to generate the 3D Delaunay triangulation of a given set of points. Our first contribution is an efficient serial implementation of the incremental Delaunay insertion algorithm. A simple dedicated data structure, an efficient sorting of the points and the optimization of the insertion algorithm have permitted to accelerate reference implementations by a factor three. Our second contribution is a multi-threaded version of the Delaunay kernel that is able to concurrently insert vertices. Moore curve coordinates are used to partition the point set, avoiding heavy synchronization overheads. Conflicts are managed by modifying the partitions with a simple rescaling of the space-filling curve. The performances of our implementation have been measured on three different processors, an Intel core-i7, an Intel Xeon Phi and an AMD EPYC, on which we have been able to compute 3 billion tetrahedra in 53 seconds. This corresponds to a generation rate of over 55 million tetrahedra per second. We finally show how this very efficient parallel Delaunay triangulation can be integrated in a Delaunay refinement mesh generator which takes as input the triangulated surface boundary of the volume to mesh.

One machine, one minute, three billion tetrahedra

Célestin Marot* | Jeanne Pellerin | Jean-François Remacle

Truck tire

# threads	# tetrahedra	Timings (s)		
		BR	Refine	Total
1	123 640 429	75.9	259.7	364.7
2	123 593 913	74.5	166.8	267.1
4	123 625 696	74.2	107.4	203.6
8	123 452 318	74.2	95.5	190.0

One machine, one minute, three billion tetrahedra

Célestin Marot* | Jeanne Pellerin | Jean-François Remacle

Truck tire

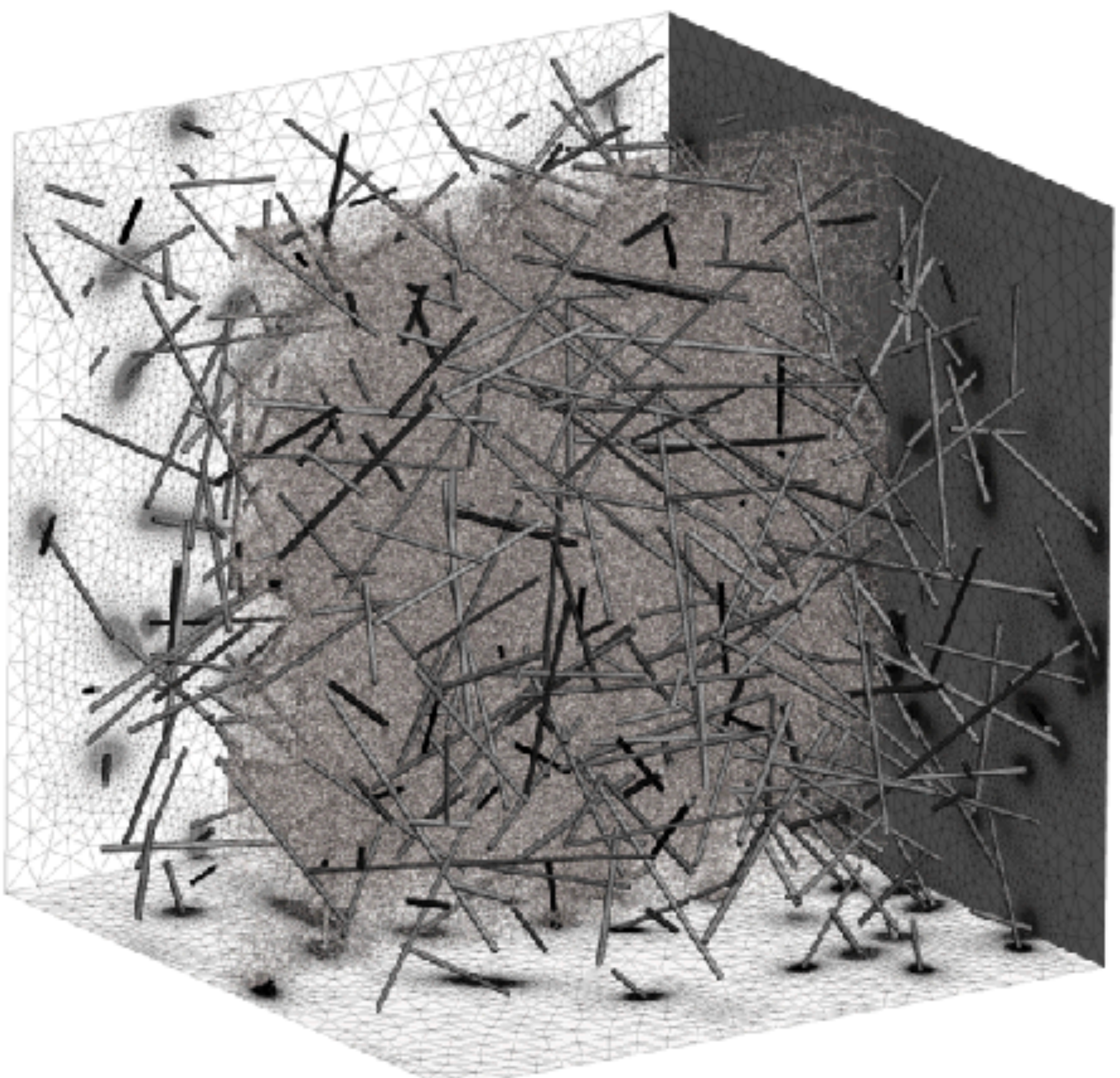
# threads	# tetrahedra	Timings (s) BR	Refine	Total
1	123 640 429	75.9	259.7	364.7
2	123 593 913	74.5	166.8	267.1
4	123 625 696	74.2	107.4	203.6
8	123 452 318	74.2	95.5	190.0

123 million tets

3 mn

One machine, one minute, three billion tetrahedra

Célestin Marot* | Jeanne Pellerin | Jean-François Remacle



100 thin fibers

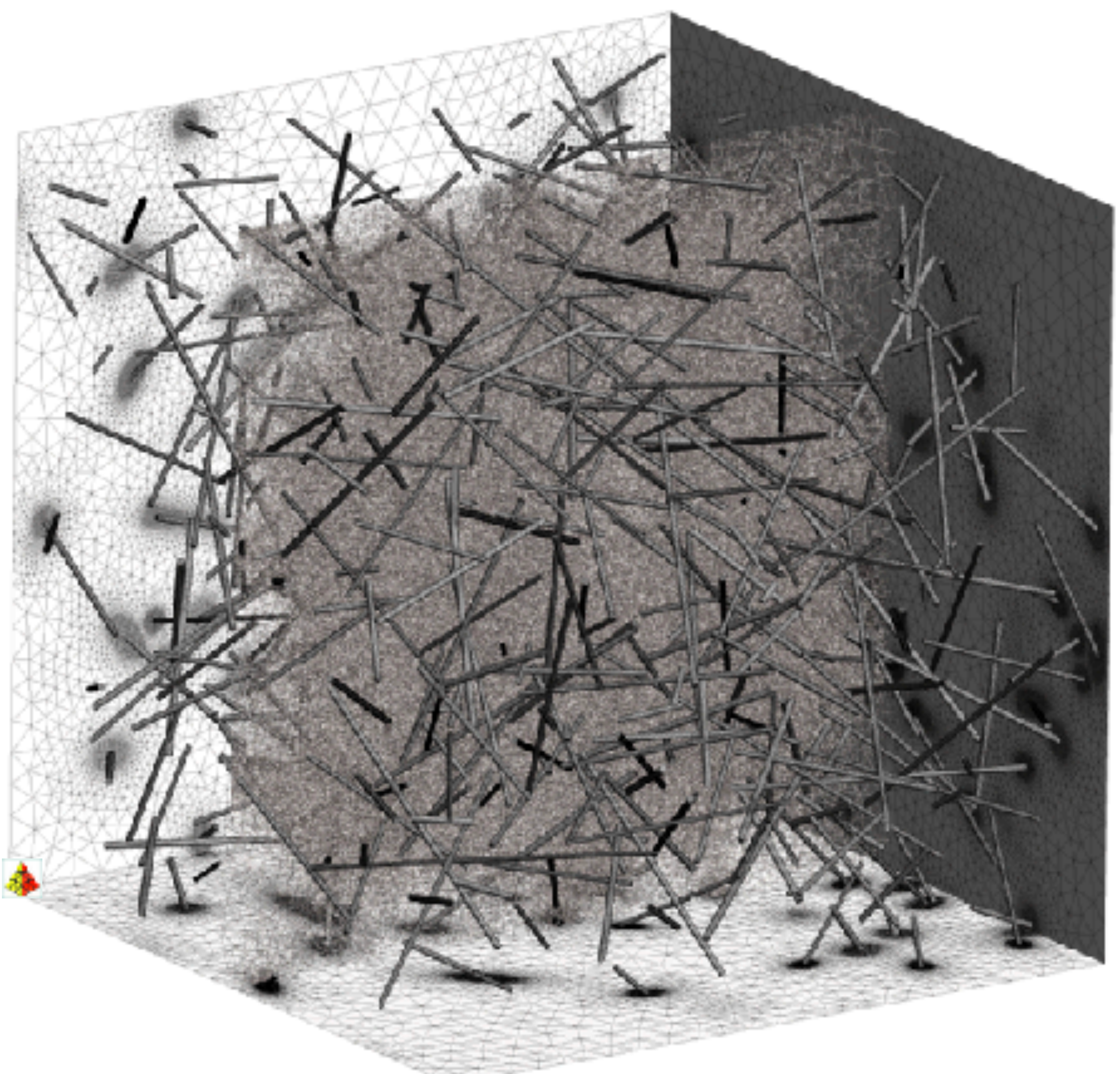
# threads	# tetrahedra	Timings (s) BR	Refine	Total
1	325 611 841	3.1	492.1	497.2
2	325 786 170	2.9	329.7	334.3
4	325 691 796	2.8	229.5	233.9
8	325 211 989	2.7	154.6	158.7
16	324 897 471	2.8	96.8	100.9
32	325 221 244	2.7	71.7	75.8
64	324 701 883	2.8	55.8	60.1
127	324 190 447	2.9	47.6	52.0

500 thin fibers

# threads	# tetrahedra	Timings (s) BR	Refine	Total
1	723 208 595	18.9	1205.8	1234.4
2	723 098 577	16.0	780.3	804.8
4	722 664 991	86.6	567.1	659.8
8	722 329 174	15.8	349.1	370.1
16	723 093 143	15.6	216.2	236.5
32	722 013 476	15.6	149.7	169.8
64	721 572 235	15.9	119.7	140.4
127	721 591 846	15.9	114.2	135.2

One machine, one minute, three billion tetrahedra

Célestin Marot* | Jeanne Pellerin | Jean-François Remacle



324 million tets 52s

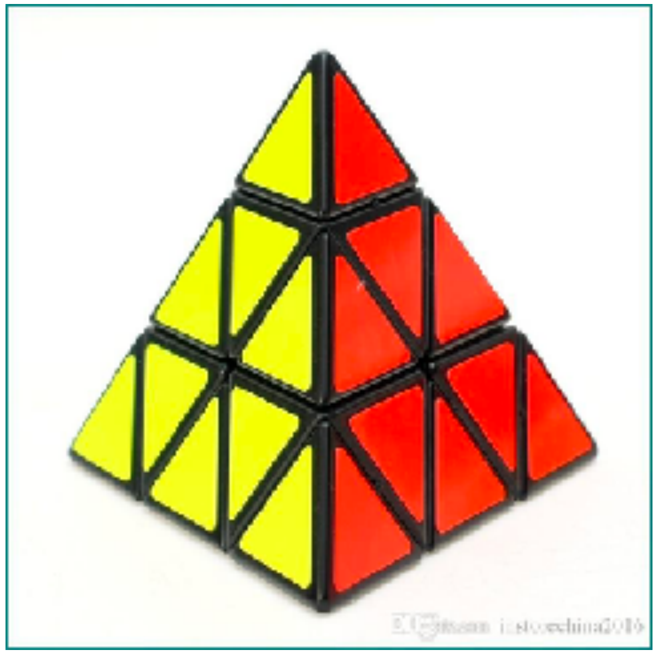
721 million tets 2mn 15s

100 thin fibers

# threads	# tetrahedra	Timings (s) BR	Refine	Total
1	325 611 841	3.1	492.1	497.2
2	325 786 170	2.9	329.7	334.3
4	325 691 796	2.8	229.5	233.9
8	325 211 989	2.7	154.6	158.7
16	324 897 471	2.8	96.8	100.9
32	325 221 244	2.7	71.7	75.8
64	324 701 883	2.8	55.8	60.1
127	324 190 447	2.9	47.6	52.0

500 thin fibers

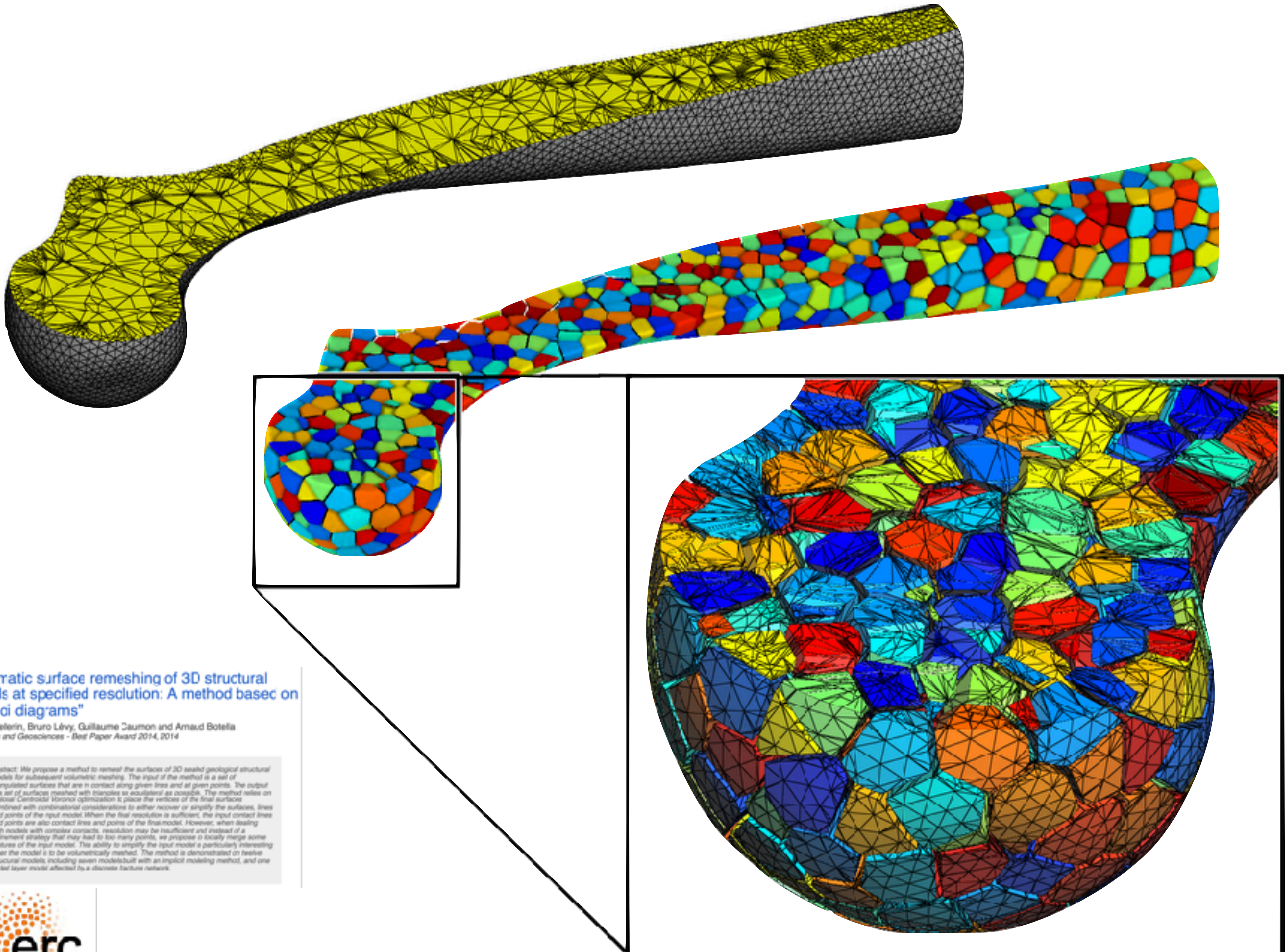
# threads	# tetrahedra	Timings (s) BR	Refine	Total
1	723 208 595	18.9	1205.8	1234.4
2	723 098 577	16.0	780.3	804.8
4	722 664 991	86.6	567.1	659.8
8	722 329 174	15.8	349.1	370.1
16	723 093 143	15.6	216.2	236.5
32	722 013 476	15.6	149.7	169.8
64	721 572 235	15.9	119.7	140.4
127	721 591 846	15.9	114.2	135.2



Linear tetrahedral elements are limited
-Stiff
-Locking
-...

Alternative element technologies have been developed

Alternative elements - polyhedral - virtual elements, HHO, SBFEM, smoothed FEM…



"Automatic surface remeshing of 3D structural models at specified resolution: A method based on Voronoi diagrams"

Jeanne Pellerin, Bruno Lévy, Guillaume Caumon and Arnaud Botella

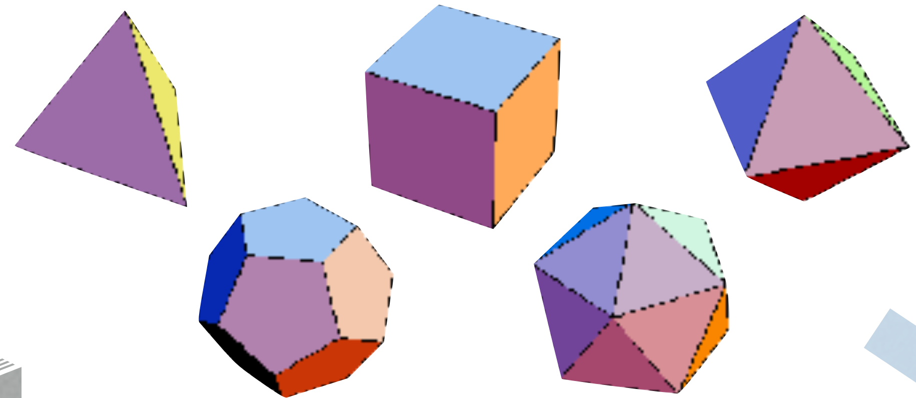
Computers and Geosciences - Best Paper Award 2014, 2014

Abstract: We propose a method to remesh the surfaces of 3D sealed geological structural models for subsequent volumetric meshing. The input of the method is a set of triangulated surfaces that are in contact along given lines and at given points. The output is a set of surfaces meshed with triangles as equilateral as possible. The method relies on a global Centroidal Voronoi optimization to place the vertices of the final surfaces combined with combinatorial considerations to either recover or simplify the surfaces, lines and points of the input model. When the final resolution is sufficient, the input contact lines and points are also contact lines and points of the final model. However, when dealing with models with complex contacts, resolution may be insufficient and instead of a refinement strategy that may lead to too many points, we propose to locally merge some features of the input model. This ability to simplify the input model is particularly interesting when the model is to be volumetrically meshed. The method is demonstrated on twelve structural models, including seven models built with an implicit modeling method, and one folded layer model affected by a discrete fracture network.

Stéphane Pierre Alain BORDAS, Department of Computational Engineering Sciences University of Luxembourg

IntuiSIM

Use polyhedra



Mesh generators...

Bruno Lévy


3D NOffset mixed-element mesh generator approach

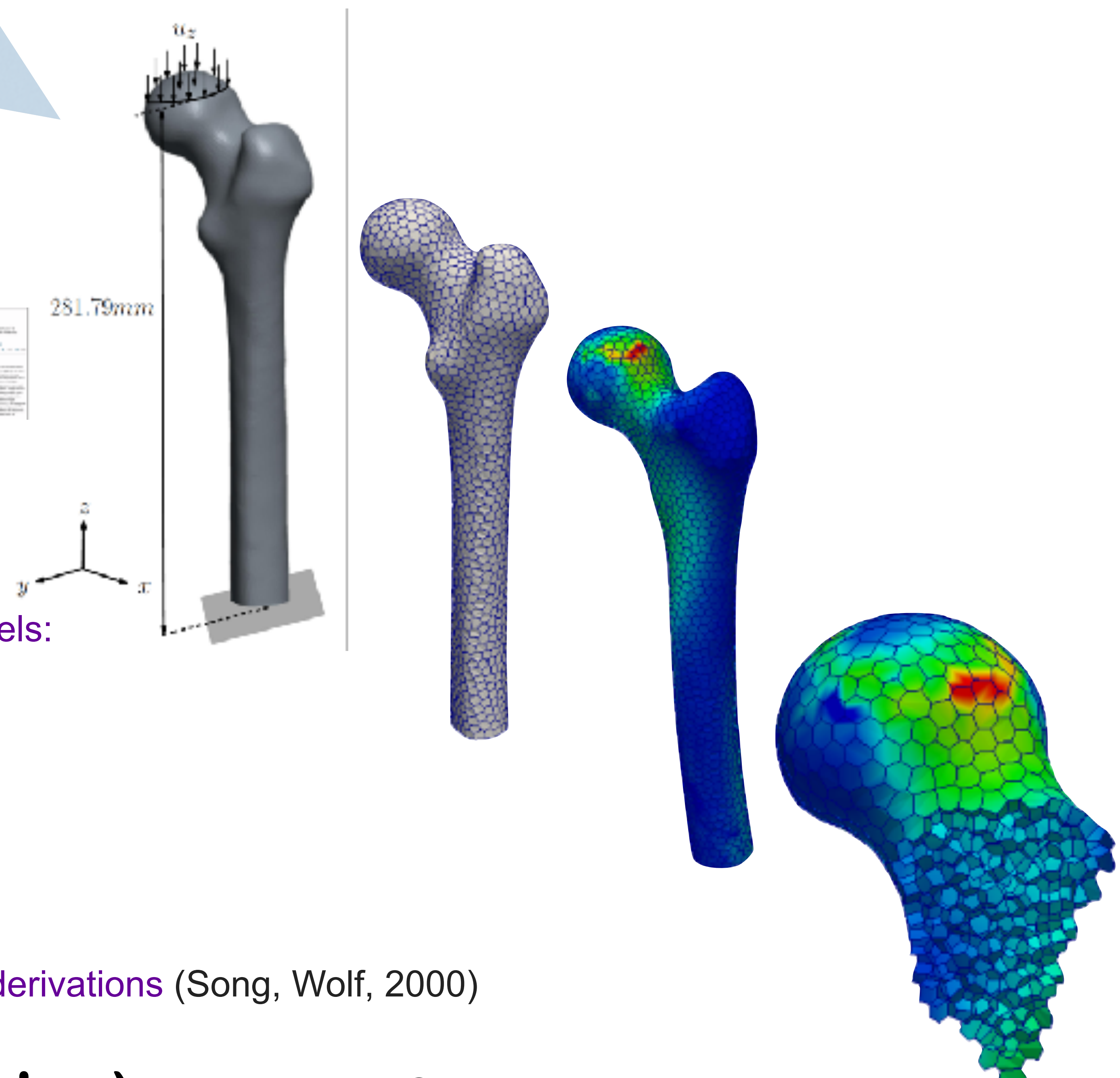
Claudio Lobos and Nancy Hitschfeld-Kahler

Claudio Lobos

Virtual elements

[The hitchhiker's guide to the **virtual element method**]

[Virtual and smoothed finite elements: A connection and its application to polygonal/polyhedral finite element methods] (Natarajan, Ooi, Bordas)



Smoothed FEM

A theoretical study on the **smoothed FEM** (S−**FEM**) models: Properties, accuracy and convergence rates (G.R. Liu, Nguyen et al)

On the approximation in the **smoothed** finite element method (SFEM) (Natarajan, Bordas)

Scaled boundary FEM

The **scaled boundary finite-element** method–a primer: derivations (Song, Wolf, 2000)

HHO (cf. F. Chouly and G. Delay)

Smoothed polyhedral FEMs
Francis, Natarajan, Lévy, Bordas, 2019

International Journal for Numerical Methods in Engineering

International Journal for Numerical Methods in Engineering homepage

Research Article

Virtual and smoothed finite elements: A connection and its application to polygonal/polyhedral finite element methods

Sundararajan Natarajan, Stéphane PA Bordas, Ean Tat Ooi

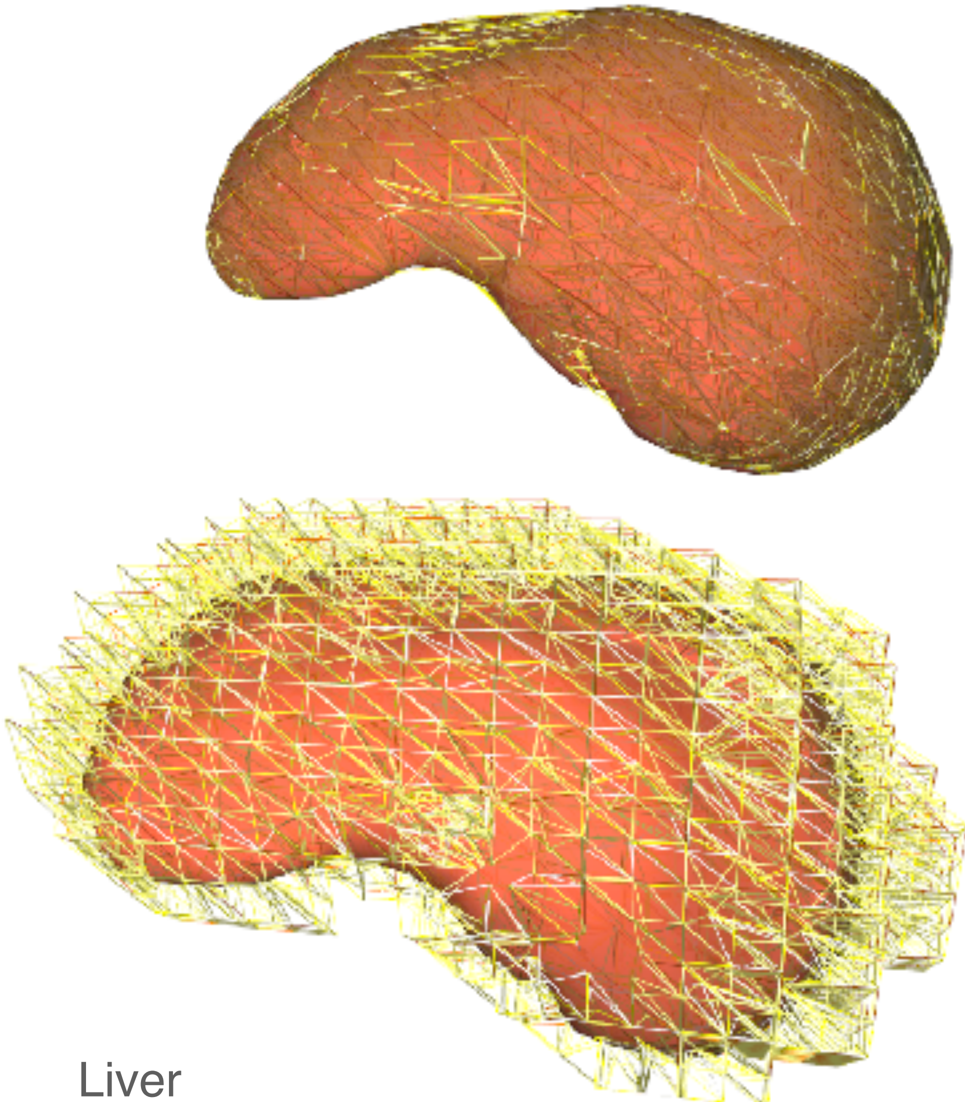
First published: 15 June 2015 | https://doi.org/10.1002/nme.4965 | Cited by: 22

Summary

We show both theoretically and numerically a connection between the smoothed finite element method (SFEM) and the virtual element method and use this approach to derive stable, cheap and optimally convergent polyhedral FEM. We show that the stiffness matrix computed with one subcell SFEM is identical to the consistency term of the virtual element method, irrespective of the topology of the element, as long as the shape functions vary linearly on the boundary. Using this connection, we propose a new stable approach to strain smoothing for polygonal/polyhedral elements where, instead of using sub-triangulations, we are able to use one single polygonal/polyhedral subcell for each element while maintaining stability. For a similar number of degrees of freedom, the proposed approach is more accurate than the conventional SFEM with triangular subcells. The time to compute the stiffness matrix scales with the $\mathcal{O}(dofs)^{1.1}$ in case of the conventional polygonal FEM, while it scales as $\mathcal{O}(dofs)^{0.7}$ in the proposed approach. The accuracy and the convergence properties of the SFEM are studied with a few benchmark problems in 2D and 3D linear elasticity.

Avoid meshing complex/evolving interfaces through unfitted methods

Implicit boundaries and error control for real time simulations



Liver

Deep brain stimulation simulation

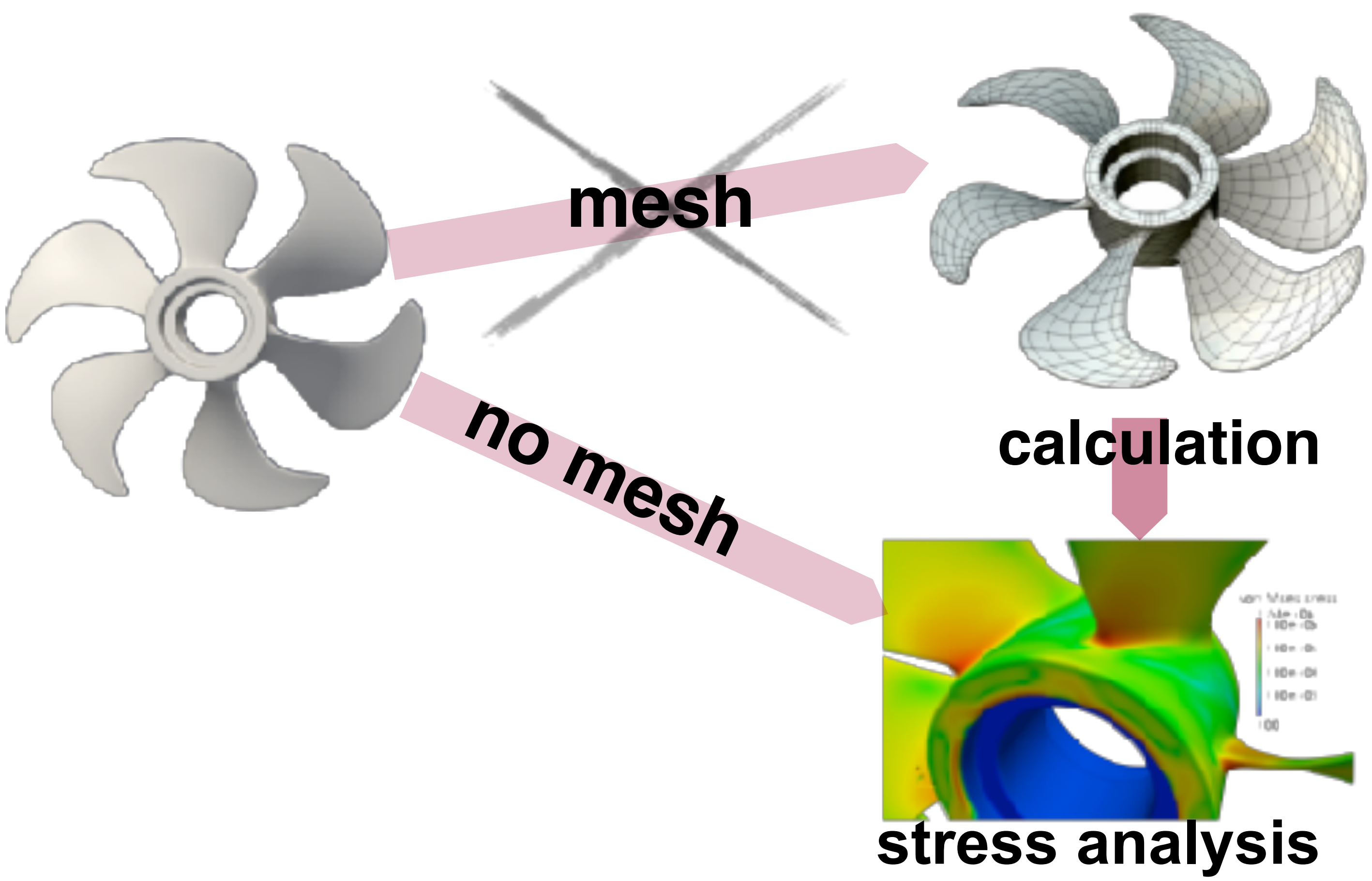
Real-time Error Control for Surgical Simulation, HP **Bui** et al, **IEEE Trans. Biomed. Eng., 2016.**
Controlling the Error on Target Motion through Real-time Mesh Adaptation: Applications to Deep Brain Stimulation, HP **Bui** et al, **Int J Numer Meth Bio, 2017.**

Corotational Cut Finite Element Method for real-time surgical simulation: application to needle insertion simulation, HP **Bui** et al, **arXiv**:1712.03052[cs.CE] **2018**.

erc
RealTCut

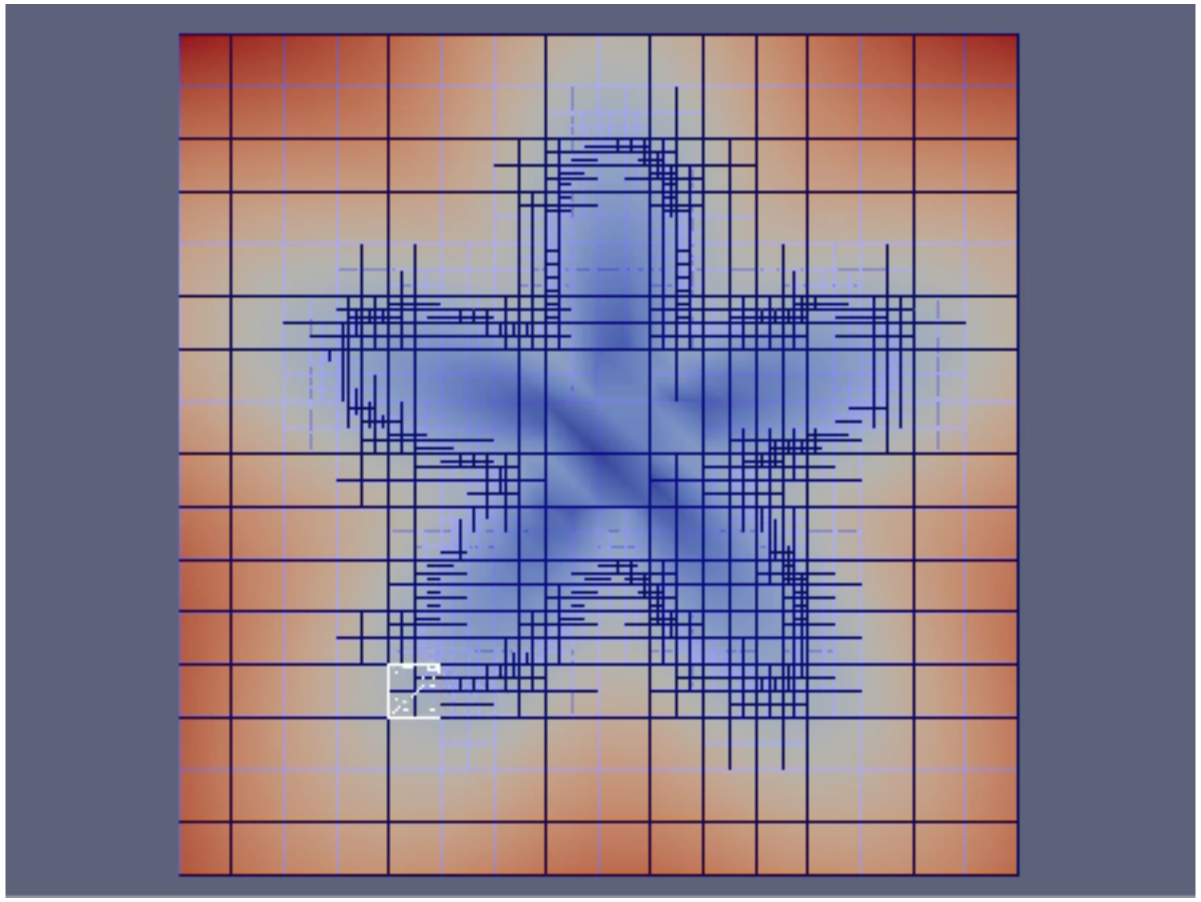
Handling interfaces numerically

Couple geometry & analysis



Isogeometric analysis

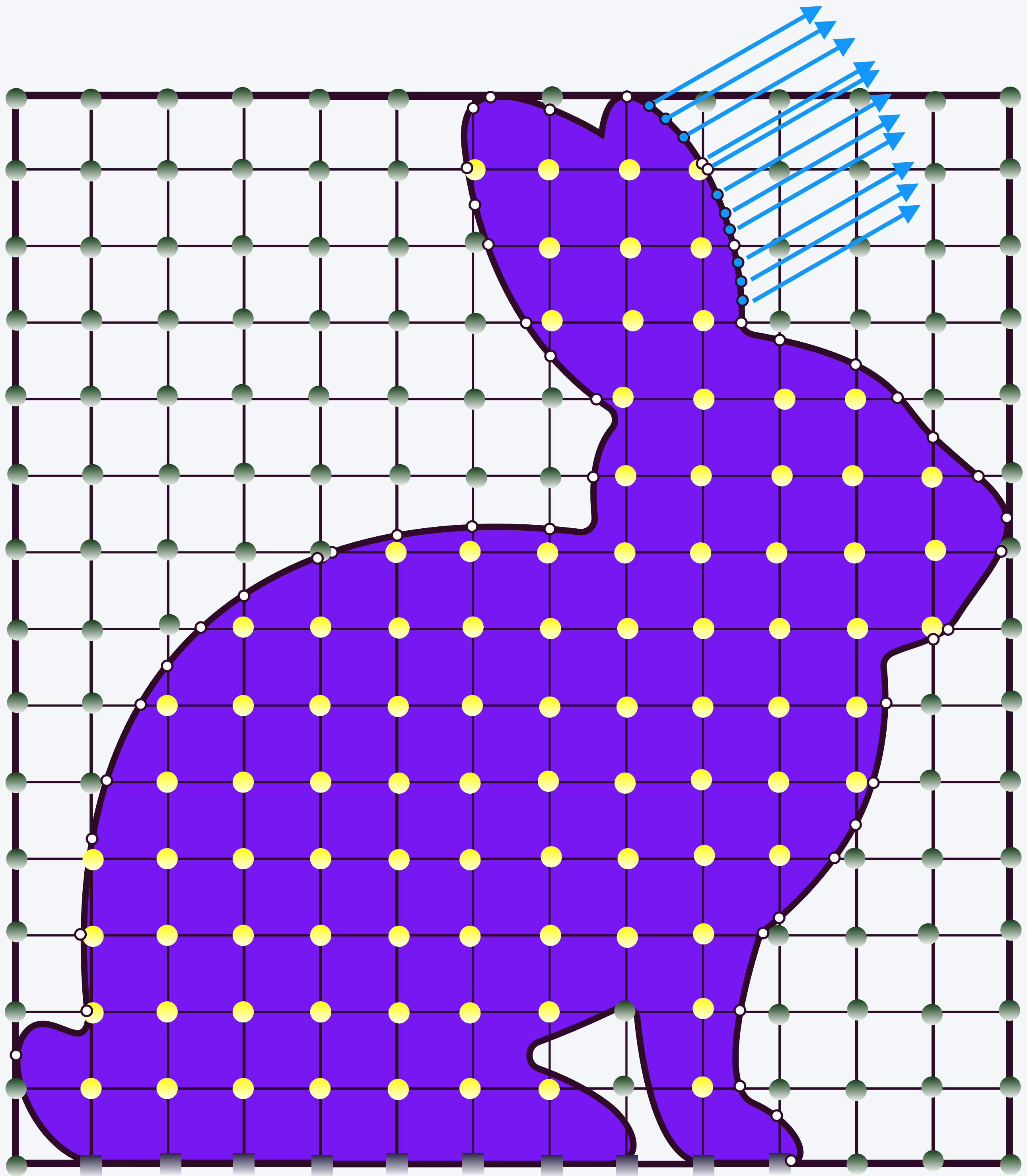
Decouple geometry from analysis



Implicit interfaces/unfitted

Generalisation: geometry independent field approximation (GIFT)
Atroshchenko et al, 2018, CMAME

Immersed collocation generalized FD



Formulation

Governing equations

$$\mathscr{L}u = f \quad \text{in} \quad \Omega$$
$$u(\mathbf{x}) = g(\mathbf{x}) \quad \text{on} \quad \partial\Omega$$

Test & Trial functions Space

$$\mathscr{U}^h \subset \mathscr{U} = \left\{ u \in H^1(\Omega) \quad \text{such that} \quad u|_{\partial\Omega} = g \right\}$$
$$\mathscr{V}^h \subset \mathscr{V} = \left\{ v \in H^1(\Omega) \quad \text{such that} \quad v|_{\partial\Omega} = 0 \right\}$$

Weak form

$$\text{find} \quad u^h \in \mathscr{U}^h : \forall v^h \in \mathscr{V}^h \quad a(u^h, v^h) = \ell(v^h)$$

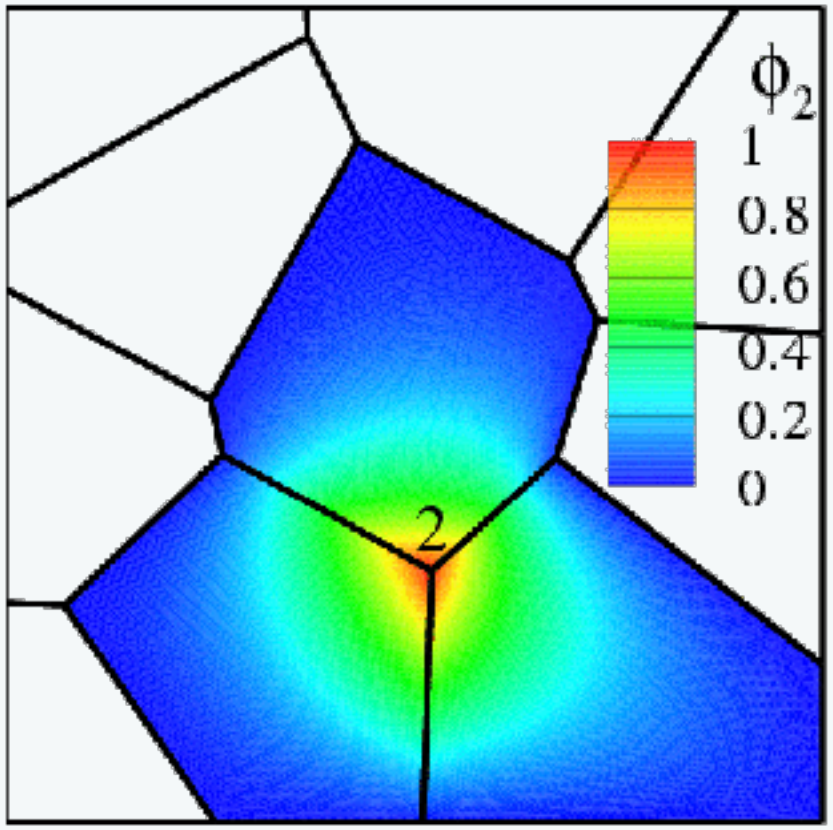
Approximate solutions

$$u^h = \sum_I \psi_I u_I \qquad v^h = \sum_I \psi_I v_I$$

FE implementation

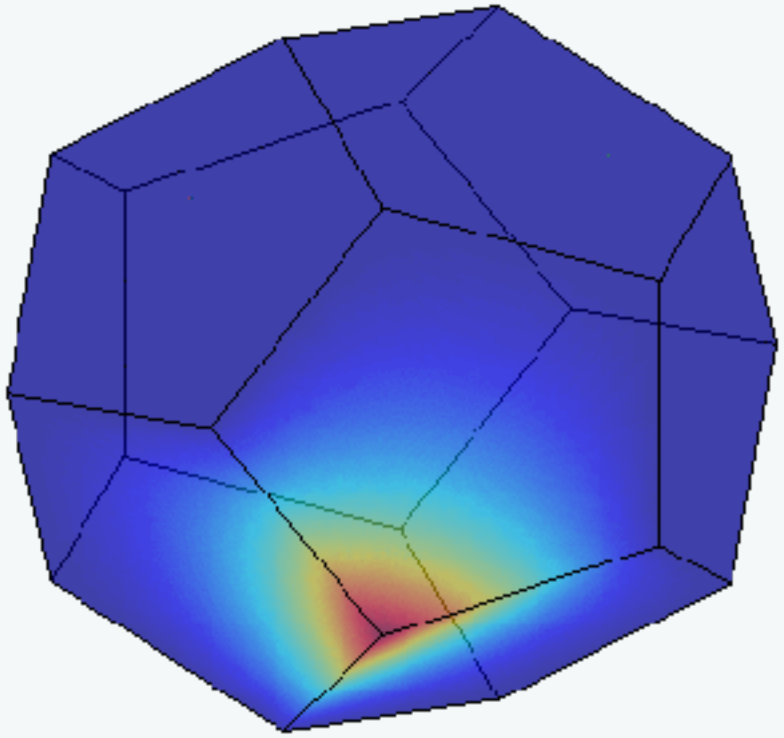
Shape functions

- Length and area measures:
  - Wachspress interpolants (1975)



- Natural Neighbour interpolants:
  - Sibson interpolant (1980),
  - Laplace interpolant

- Harmonic
  - Waren et al, (1996, 2007)



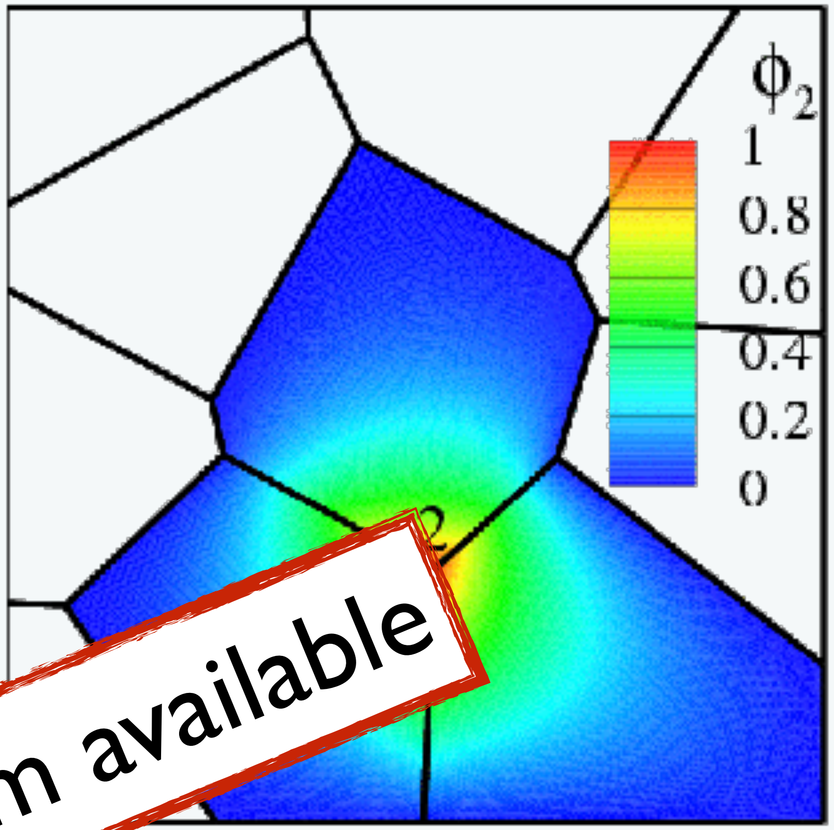
- Maximum entropy approximants (Sukumar 2013)

- Mean value coordinates Floater et al., (2003, 2005)

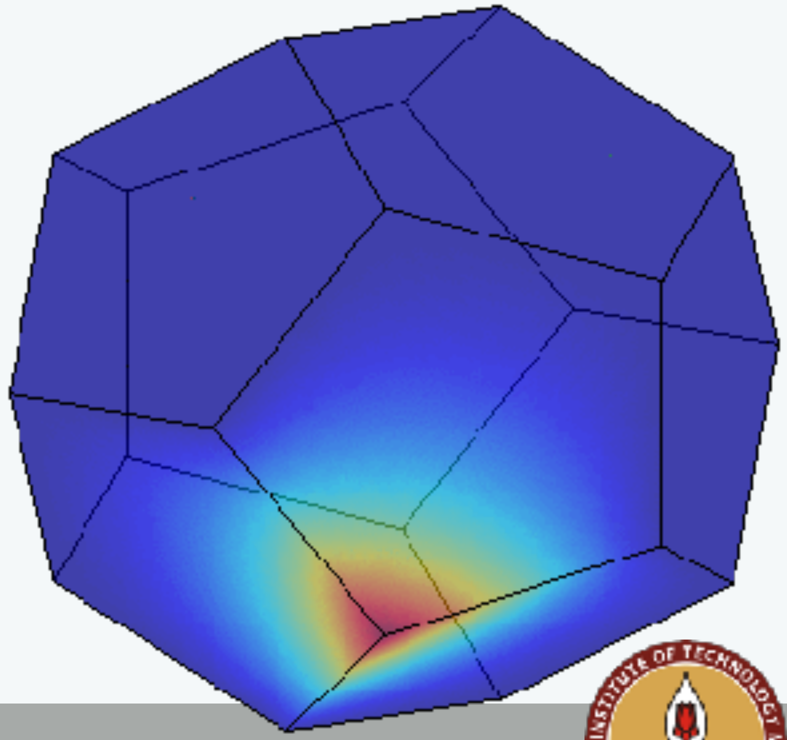
FE implementation

Shape functions

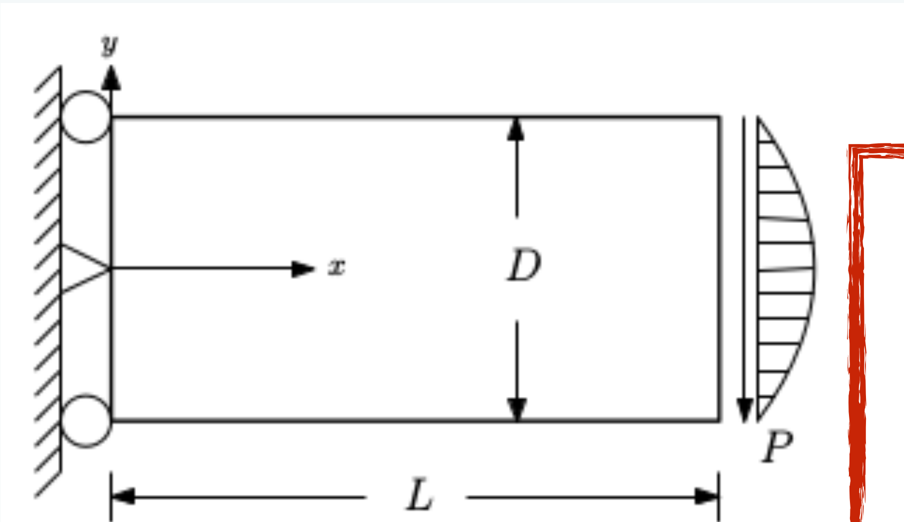
- Length and area measures:
  - Wachspress interpolants (1975)
- Natural Neighbour interpolants:
  - Sibson interpolant (1980),
  - Laplace interpolant
- Harmonic
  - Waren et al, (1996, 2007)
- Maximum entropy approximants (Sukumar 2013)
- Mean value coordinates Floater et al., (2003, 2005)

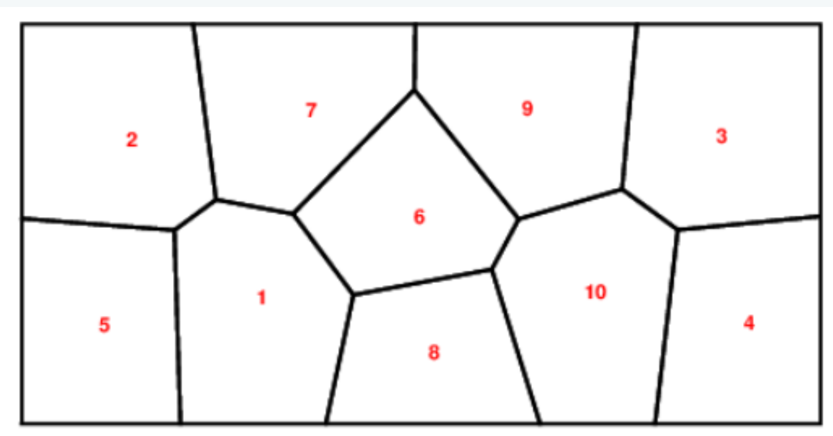


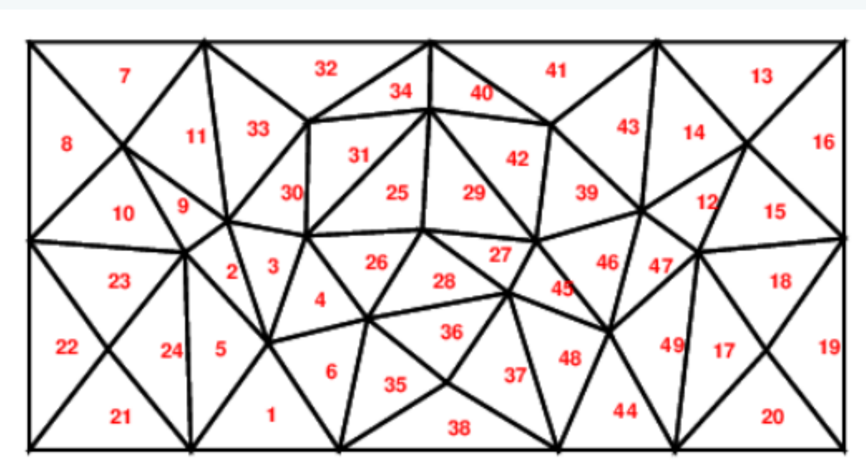
No explicit form available

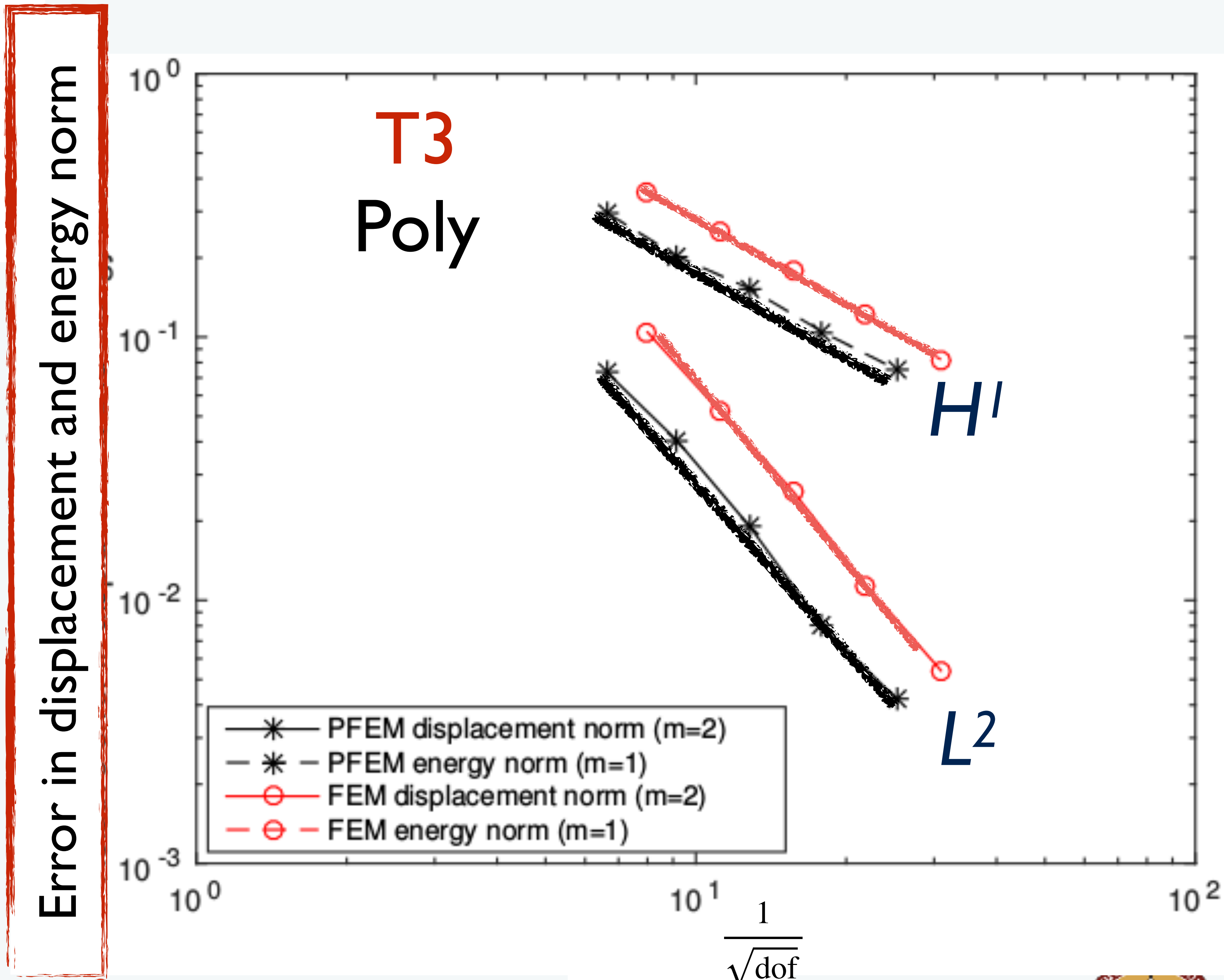


Similar accuracy & convergence to FEM

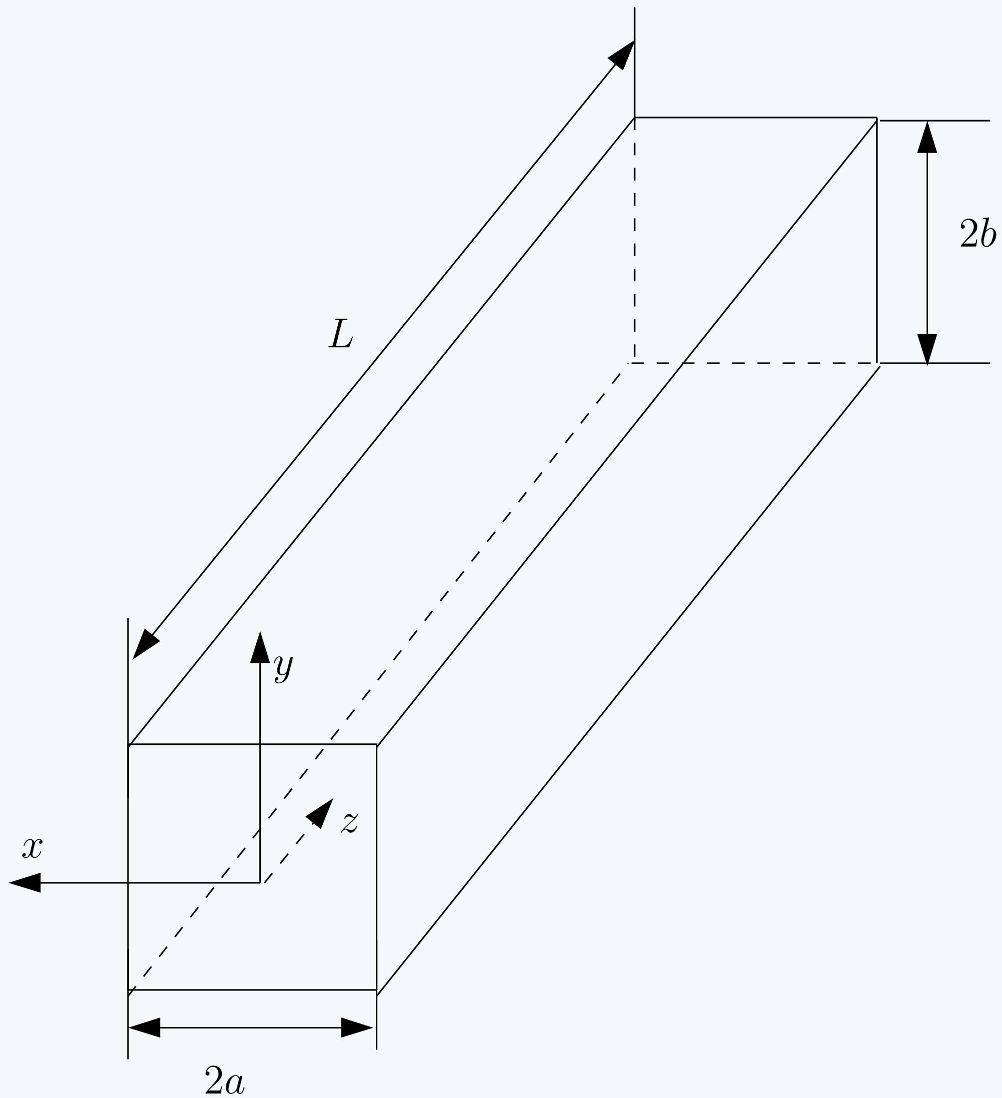


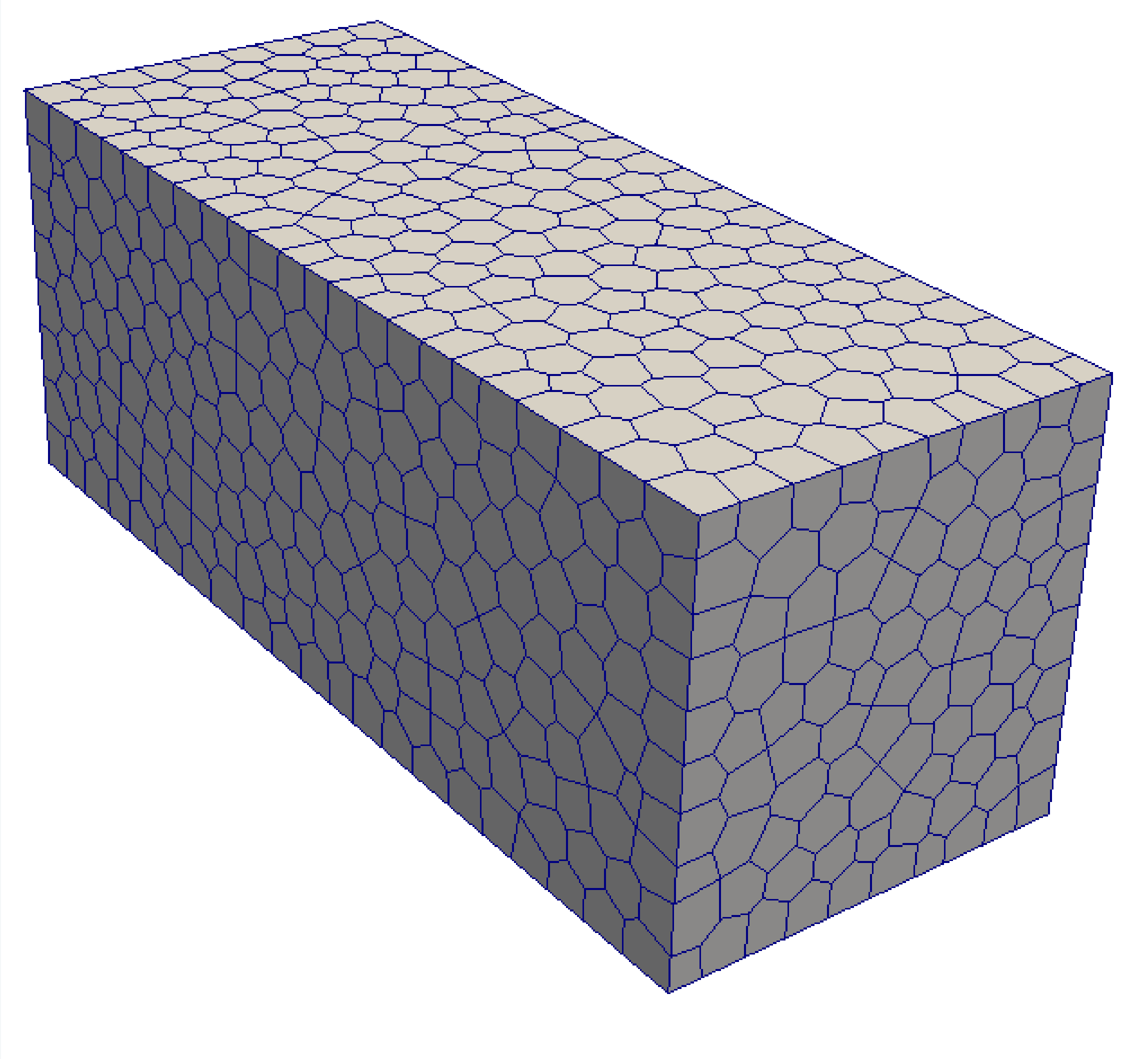


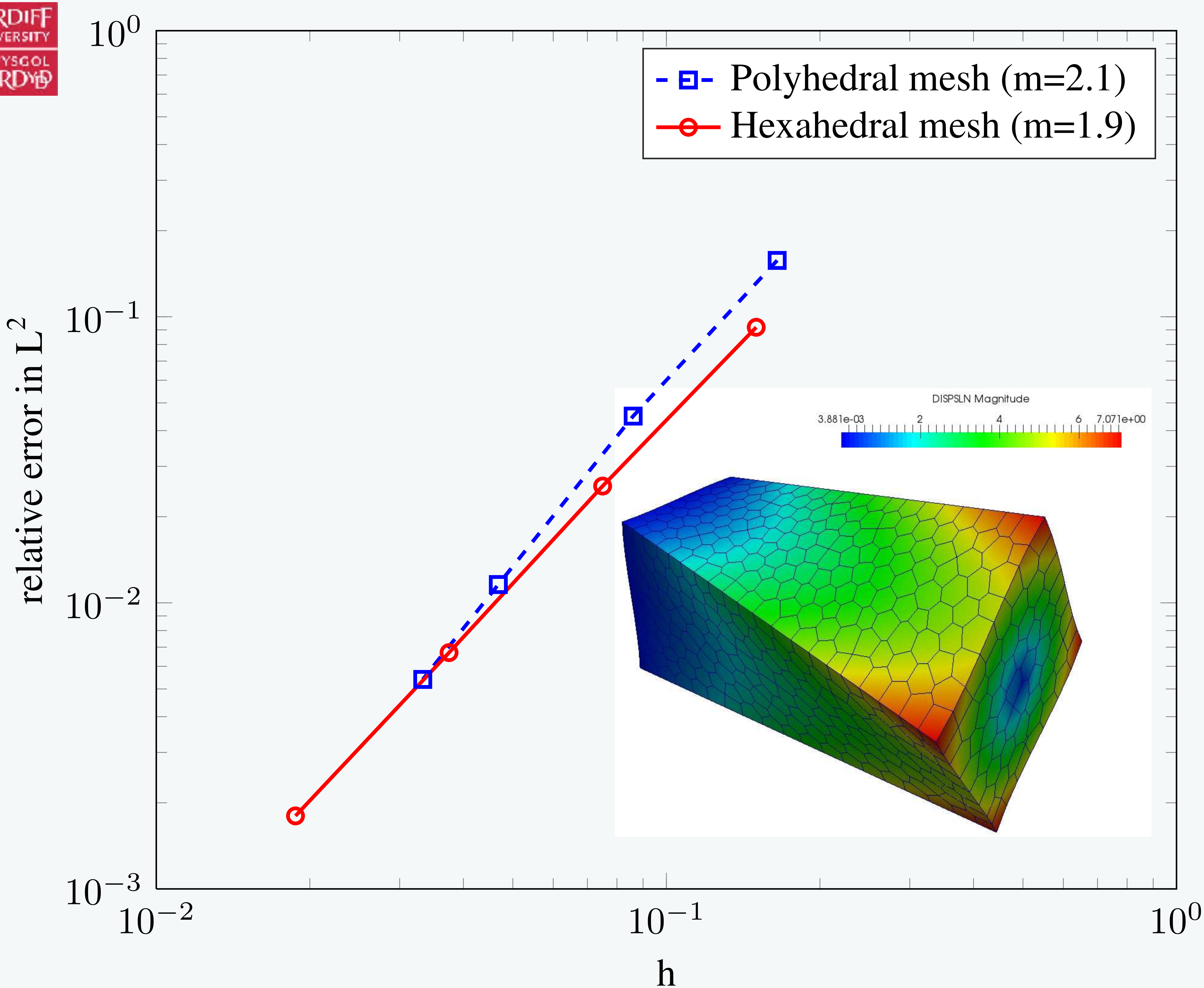




Cantilever Beam - Torsion





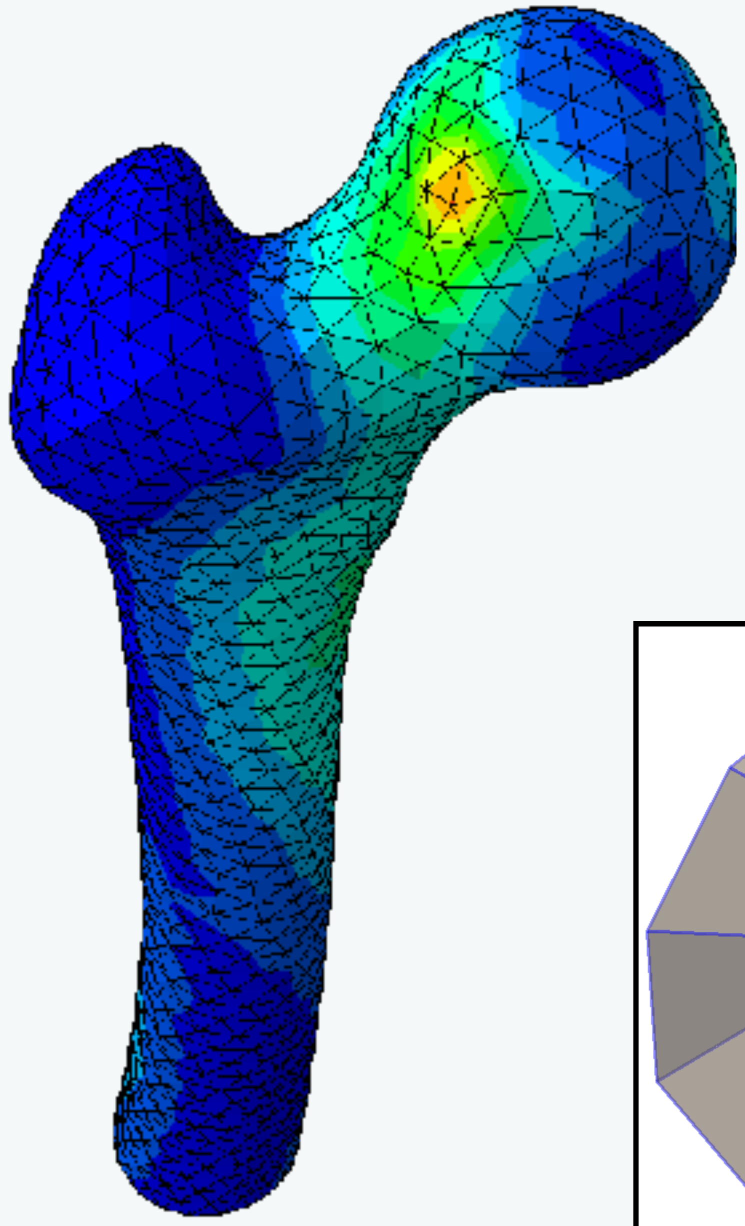


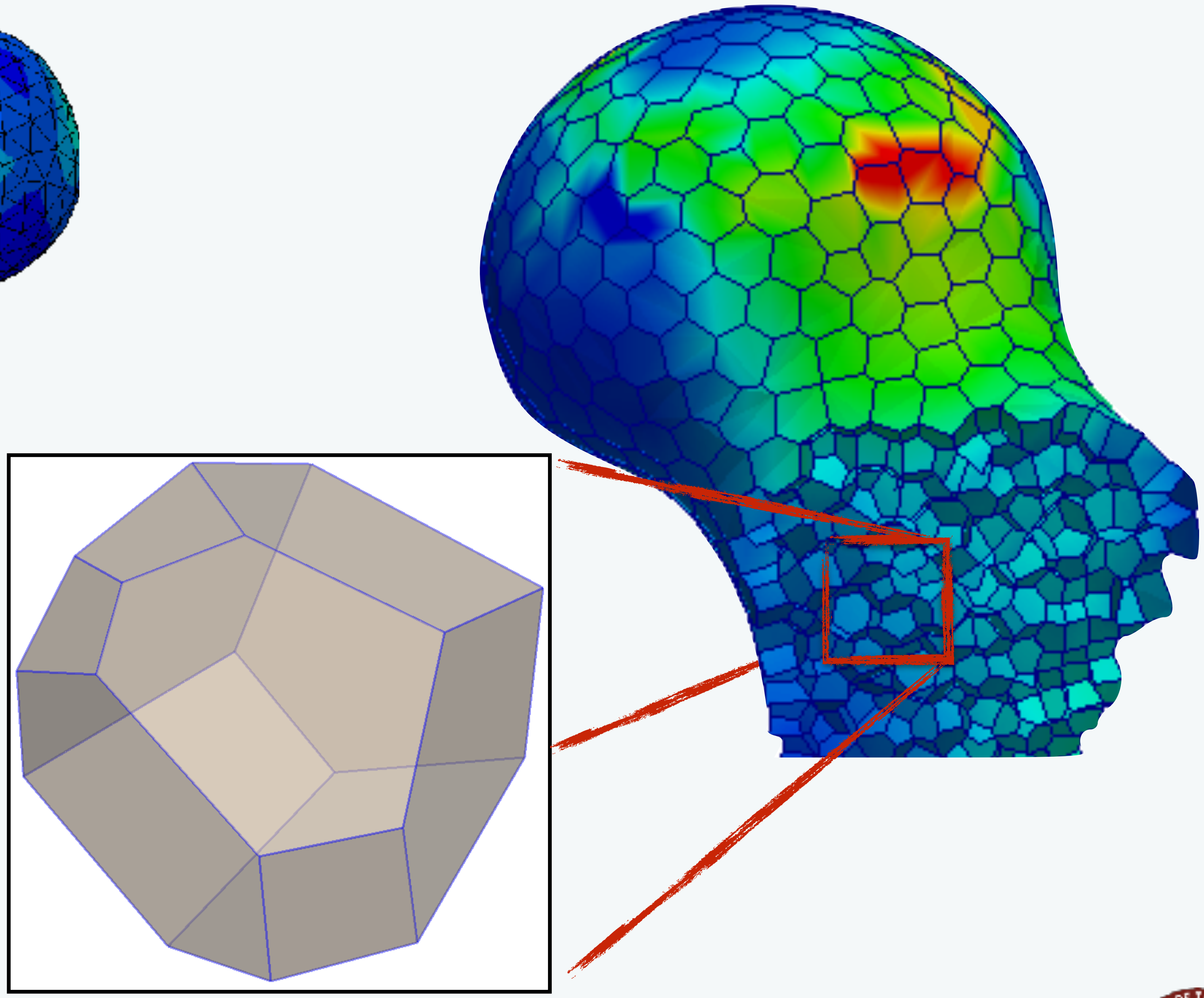
Polyhedral mesh (m=2.1)
Hexahedral mesh (m=1.9)
relative error in L2
h
10^0
10^{-1}
10^{-2}
10^{-3}
10^{-2}
10^{-1}
10^0
DISPSLN Magnitude
3.881e-03
2
4
6
7.071e+00

FE approach

Conventional FE

Polygonal FE





Francis et al, CMAME, 2019

FE implementation

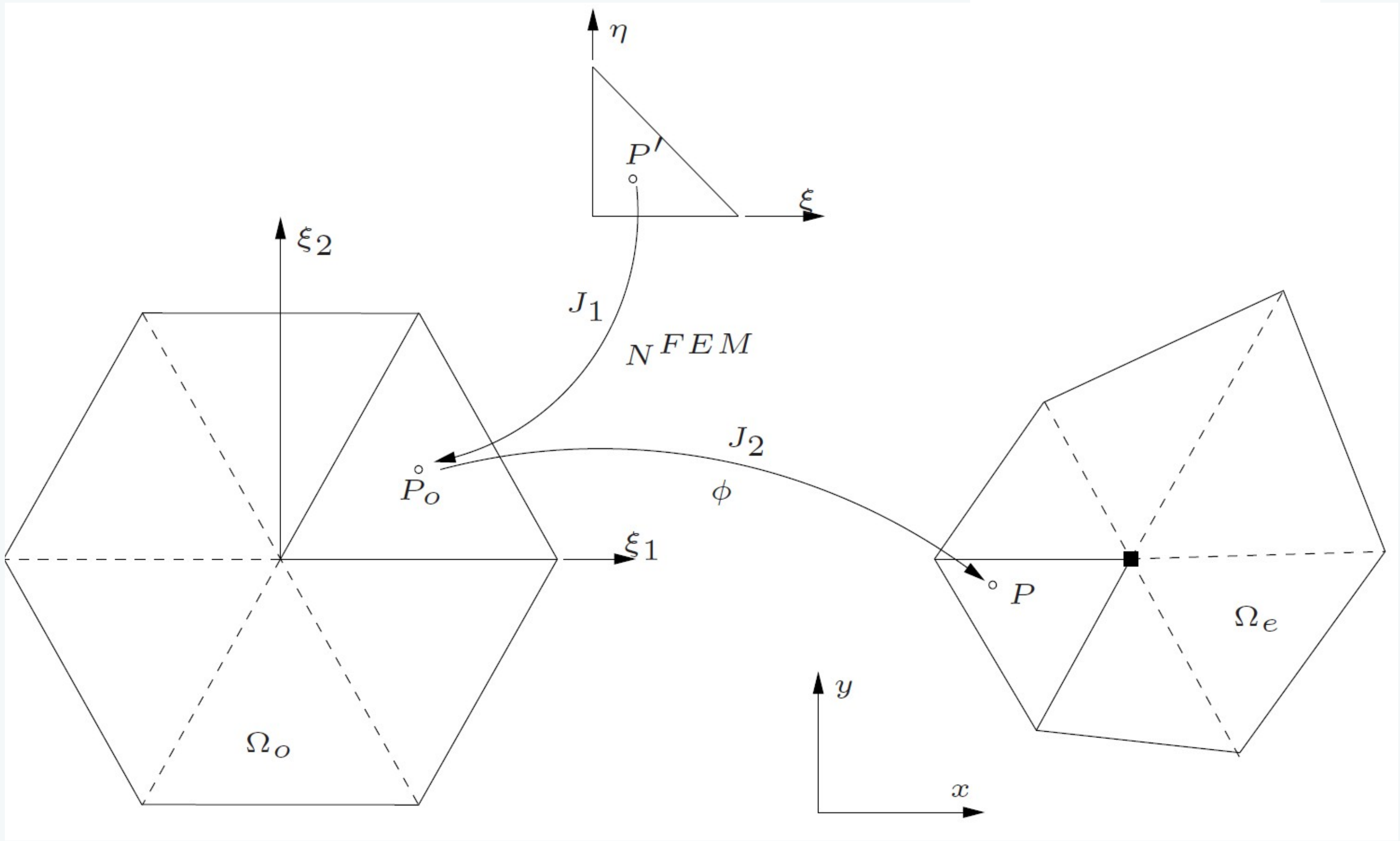
Bilinear form

$$a(u^h, v^h) = \int_{\Omega^h} \nabla\Phi \cdot \nabla\Phi \, \mathrm{d}\Omega$$

Numerical integration

- Sub- triangulation
- Green - Gauss quadrature
- Conforming interpolant quadrature
- Complex mapping
- Nodal quadrature
- Strain smoothing

Numerical integration



Sub triangulation

Moment fitting

Complex mapping

Homogeneous Numerical integration

Green-Gauss quadrature

Strain smoothing

Strain written as the divergence of a spatial average of the standard (compatible) strain field

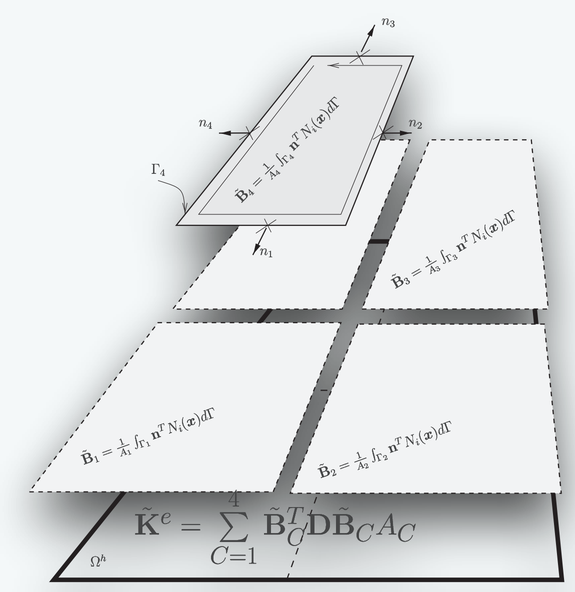
$$\tilde{\varepsilon}^h_{ij}(\mathbf{x}_C) = \int_{\Omega^h} \varepsilon^h_{ij}(\mathbf{x})\Phi(\mathbf{x}-\mathbf{x}_C)\mathrm{d}\mathbf{x}$$

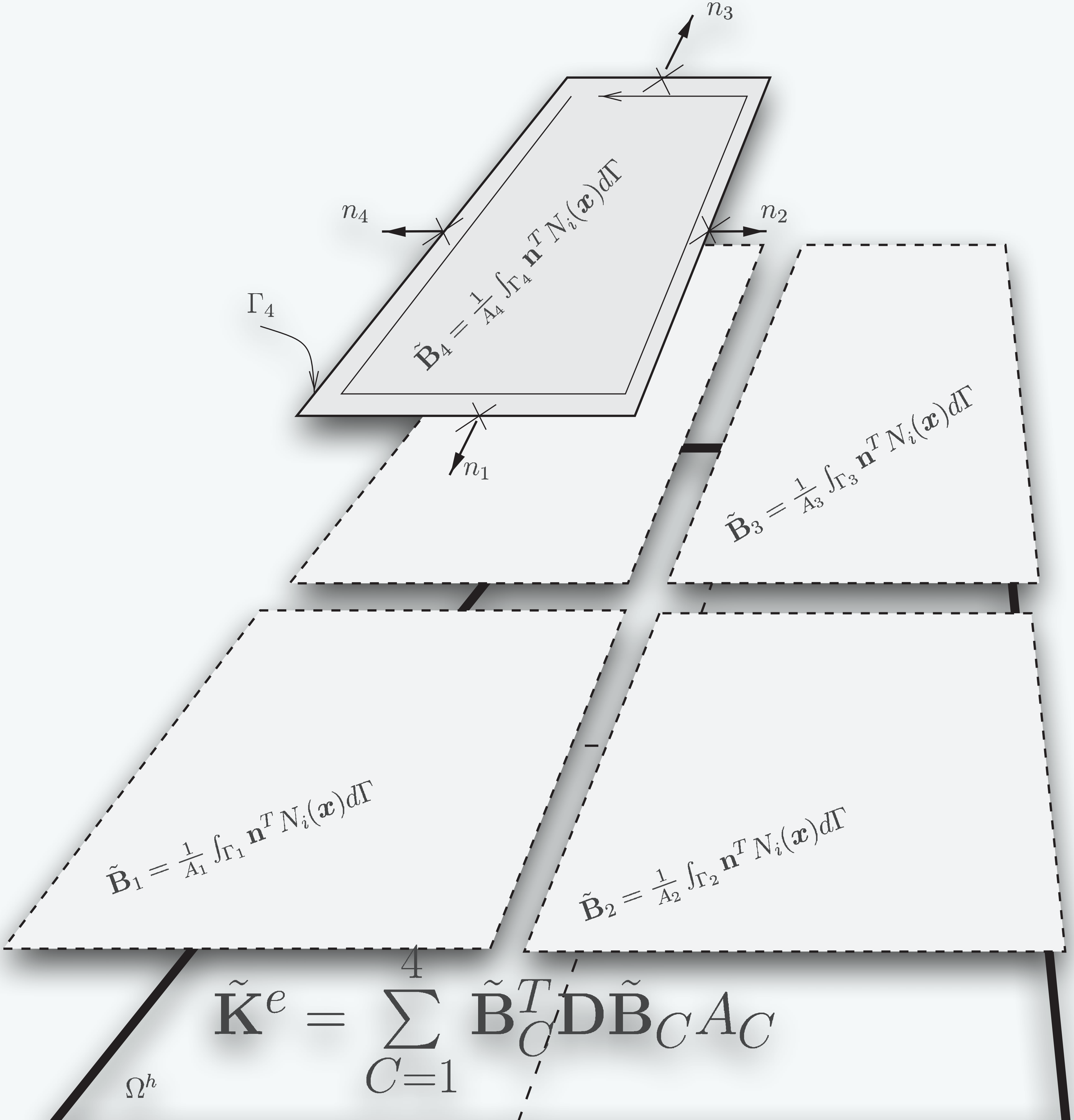
$$\Phi \geq 0 \quad \text{and} \quad \int_{\Omega^h} \Phi(\mathbf{x})\mathrm{d}\mathbf{x} = 1$$

$$\Phi = \frac{1}{A_C} \quad \text{in} \quad \Omega_C \quad \text{and} \quad \Phi = 0 \quad \text{elsewhere}$$

Stiffness matrix

$$\tilde{\mathbf{K}}^e = \sum_{C=1}^{nc} \int_{\Omega_C} \tilde{\mathbf{B}}_C^{\mathrm{T}} \mathbf{D} \tilde{\mathbf{B}}_C \mathrm{d}\Omega$$

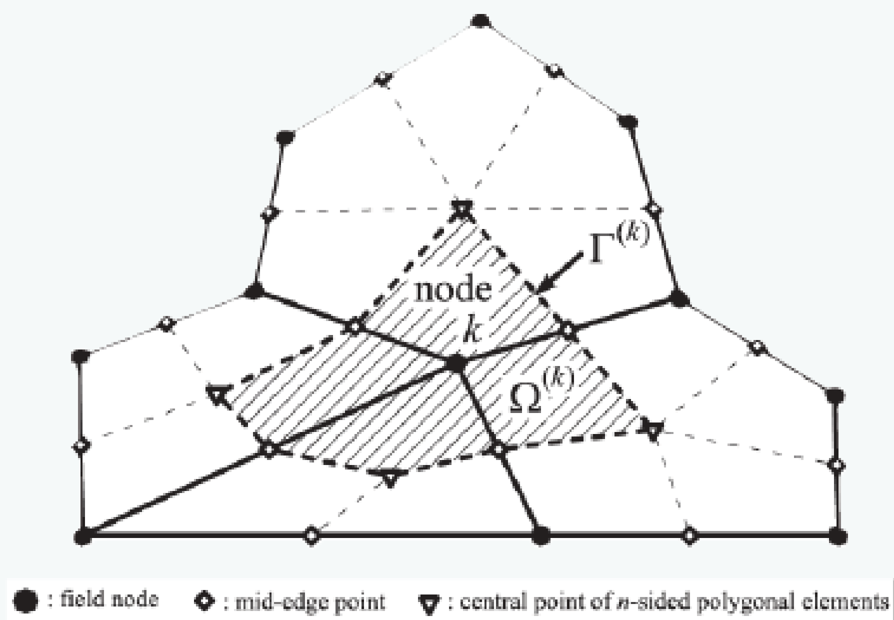




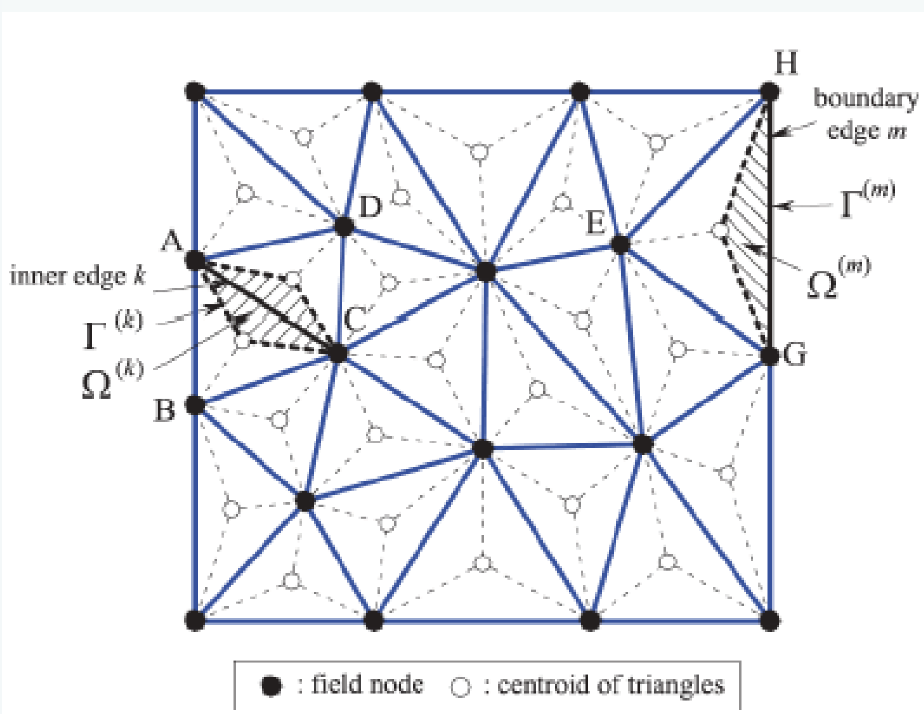
n_3
n_4
n_2
Γ_4
$\tilde{\mathbf{B}}_4 = \frac{1}{A_4}\int_{\Gamma_4} \mathbf{n}^T N_i(\boldsymbol{x}) d\Gamma$
n_1
$\tilde{\mathbf{B}}_3 = \frac{1}{A_3}\int_{\Gamma_3} \mathbf{n}^T N_i(\boldsymbol{x}) d\Gamma$
$\tilde{\mathbf{B}}_1 = \frac{1}{A_1}\int_{\Gamma_1} \mathbf{n}^T N_i(\boldsymbol{x}) d\Gamma$
$\tilde{\mathbf{B}}_2 = \frac{1}{A_2}\int_{\Gamma_2} \mathbf{n}^T N_i(\boldsymbol{x}) d\Gamma$
$\tilde{\mathbf{K}}^e = \sum_{C=1}^{4} \tilde{\mathbf{B}}_C^T \mathbf{D} \tilde{\mathbf{B}}_C A_C$
Ω^h

Strain smoothing variants

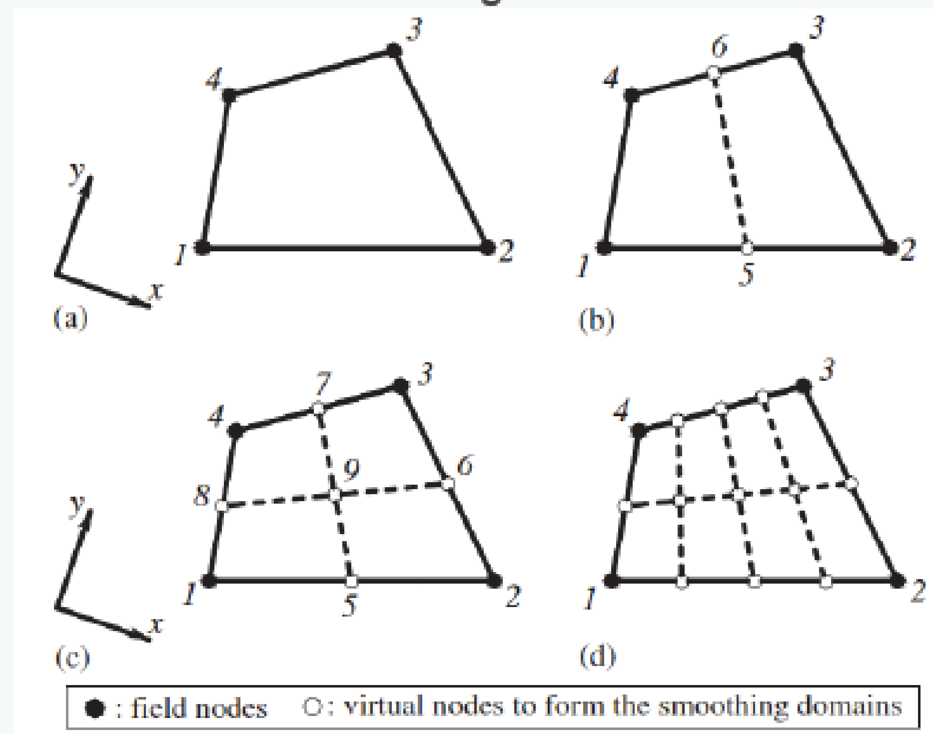
- node-based smoothing (NS-FEM)



- edge-based smoothing



- cell-based smoothing



$$\bar{\varepsilon}^{(k)} = \int_{\Omega_k^s} \varepsilon(\mathbf{x})\Phi^{(k)}(\mathbf{x})\,d\Omega$$

$$\bar{\boldsymbol{\varepsilon}}^{(k)} = \frac{1}{A_k^s}\int_{\Omega_k^s} \partial \mathbf{u}^h\, d\Omega$$

$$\Phi^{(k)}(\mathbf{x}) = \begin{cases} 1/A_k^s, & \mathbf{x}\in\Omega_k^s \\ 0, & \mathbf{x}\notin\Omega_k^s \end{cases}$$

Strain smoothing - features

+ Insensitive to mesh distortion (No isoparametric mapping)

+ Derivatives of shape functions not required

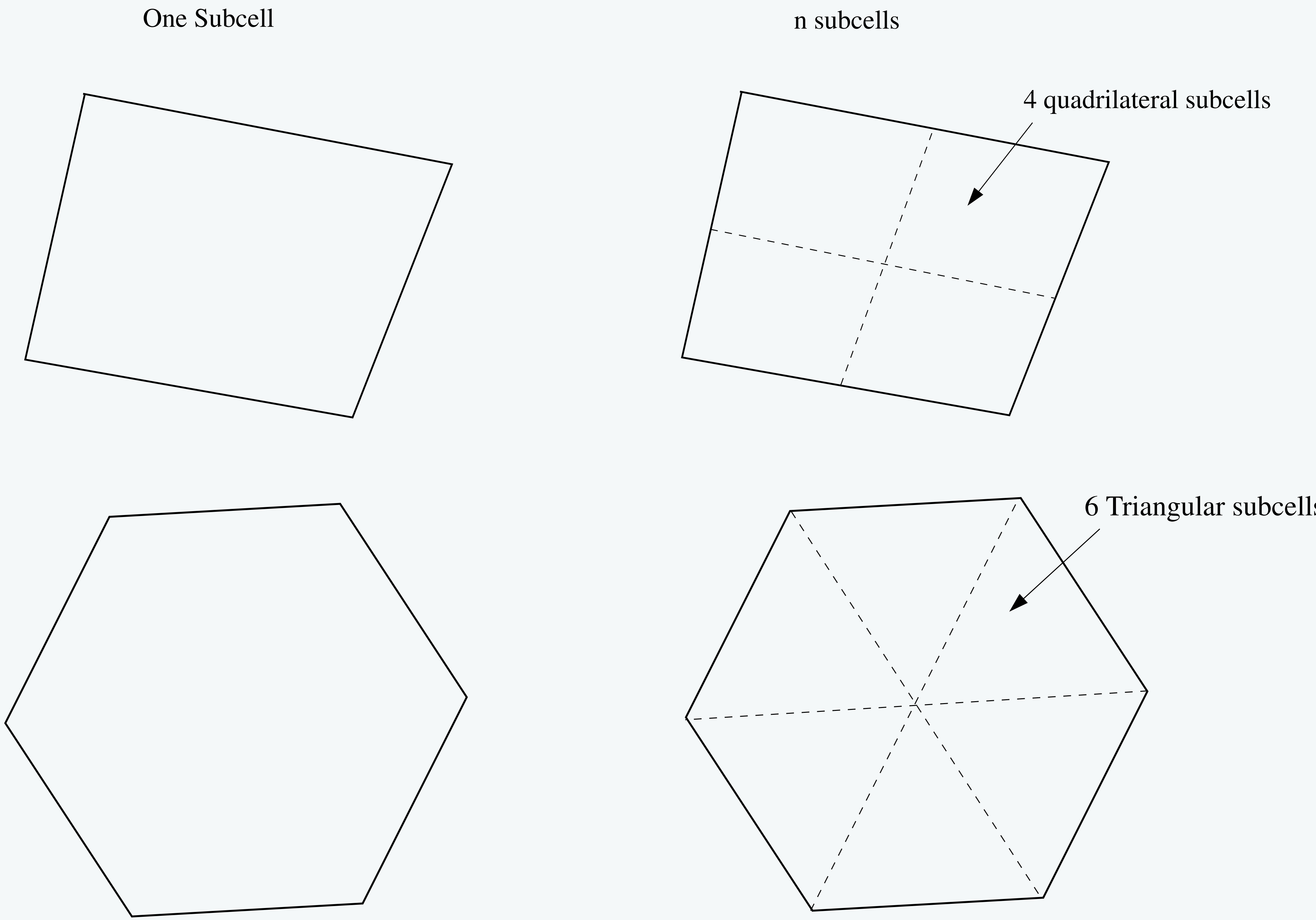
+ Insensitive to locking for low number of subcells

- Rank deficiency when using one sub-cell

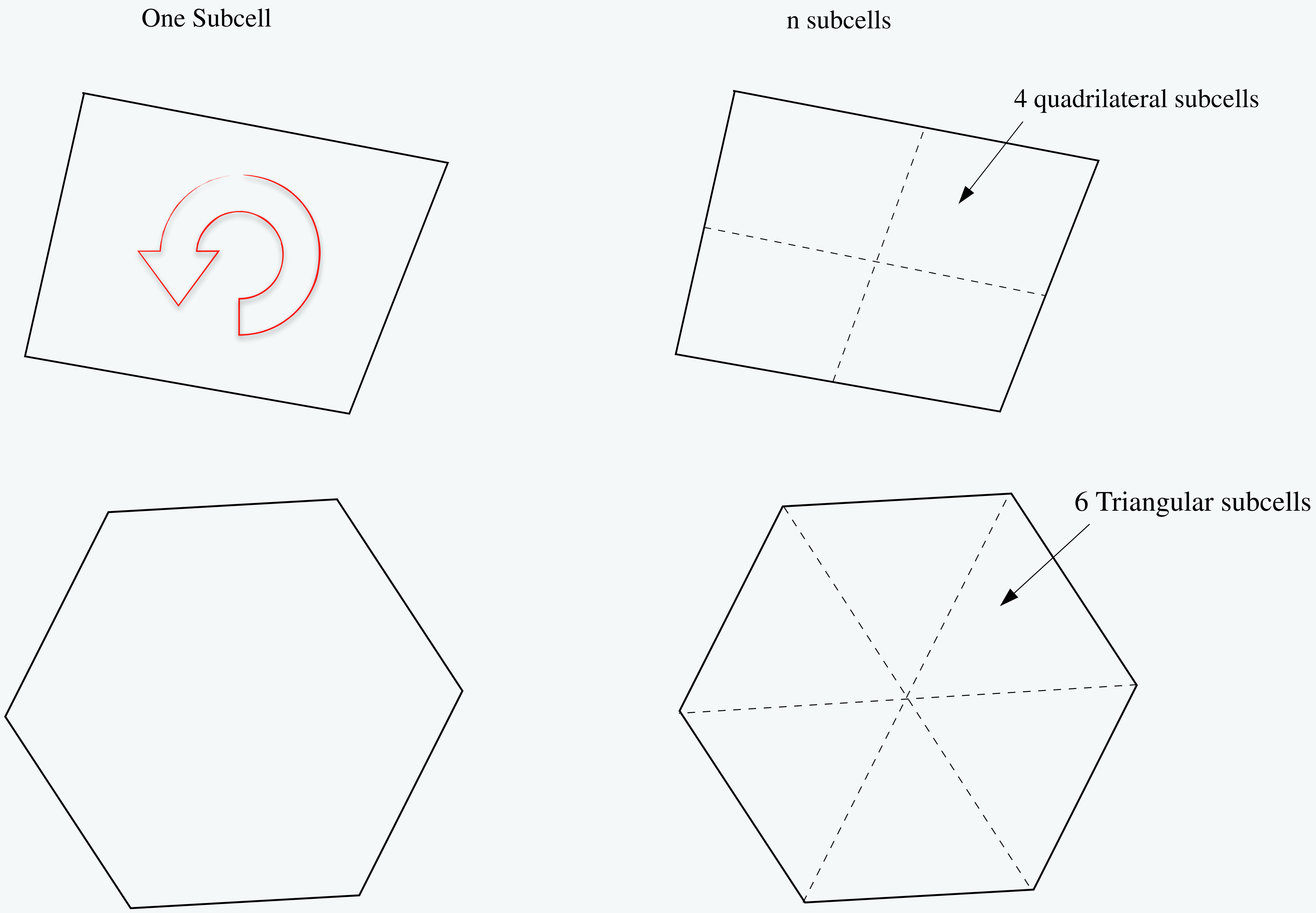
+ Better for triangular elements

+ When combined with enrichment techniques - avoids integration of stress singularity

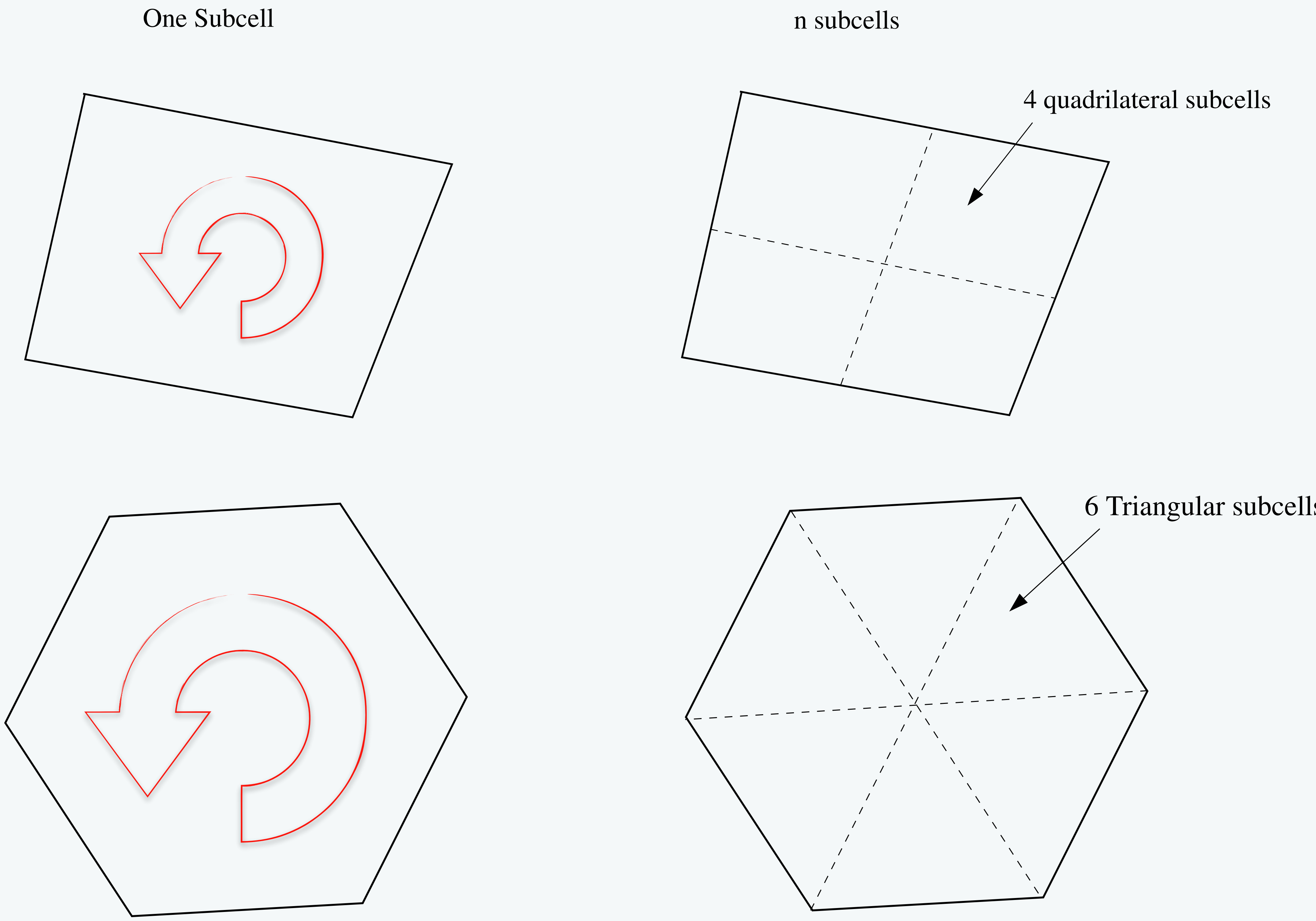
+ Decreases the complexity of sub-division in XFEM



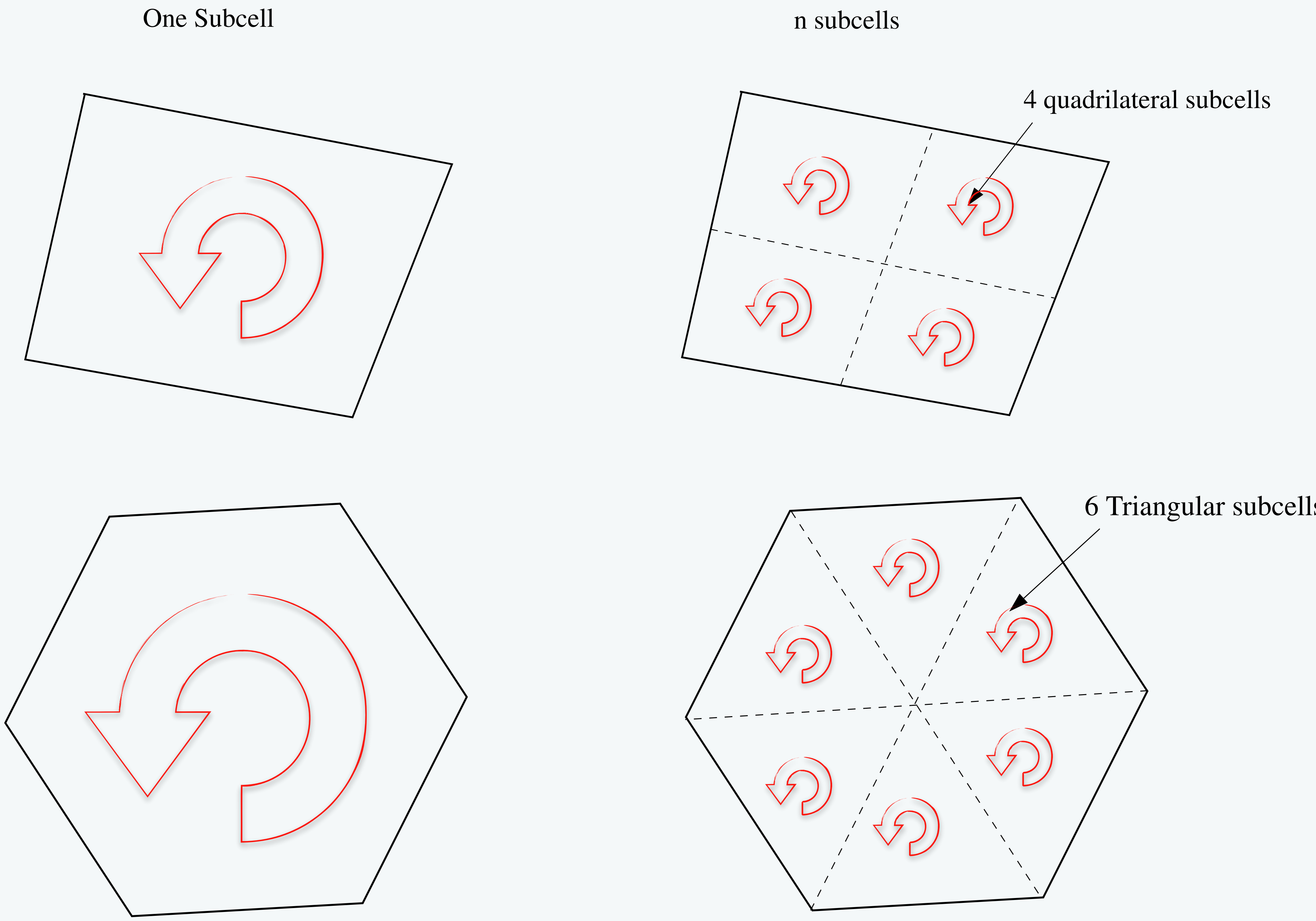
Apply strain smoothing technique over each subcell to compute the stiffness matrix.



Apply strain smoothing technique over each subcell to compute the stiffness matrix.

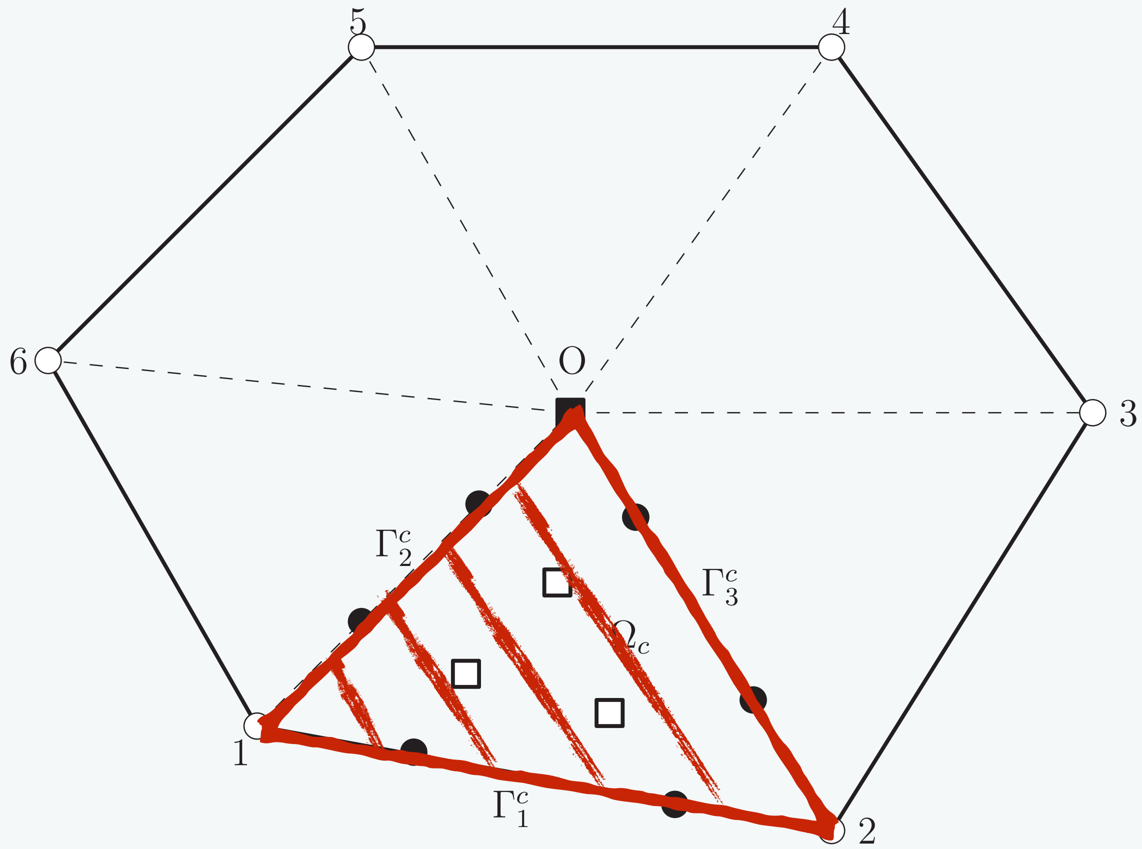


Apply strain smoothing over each subcell to compute the stiffness matrix.



Apply strain smoothing over each subcell to compute the stiffness matrix.

Linear Smoothing



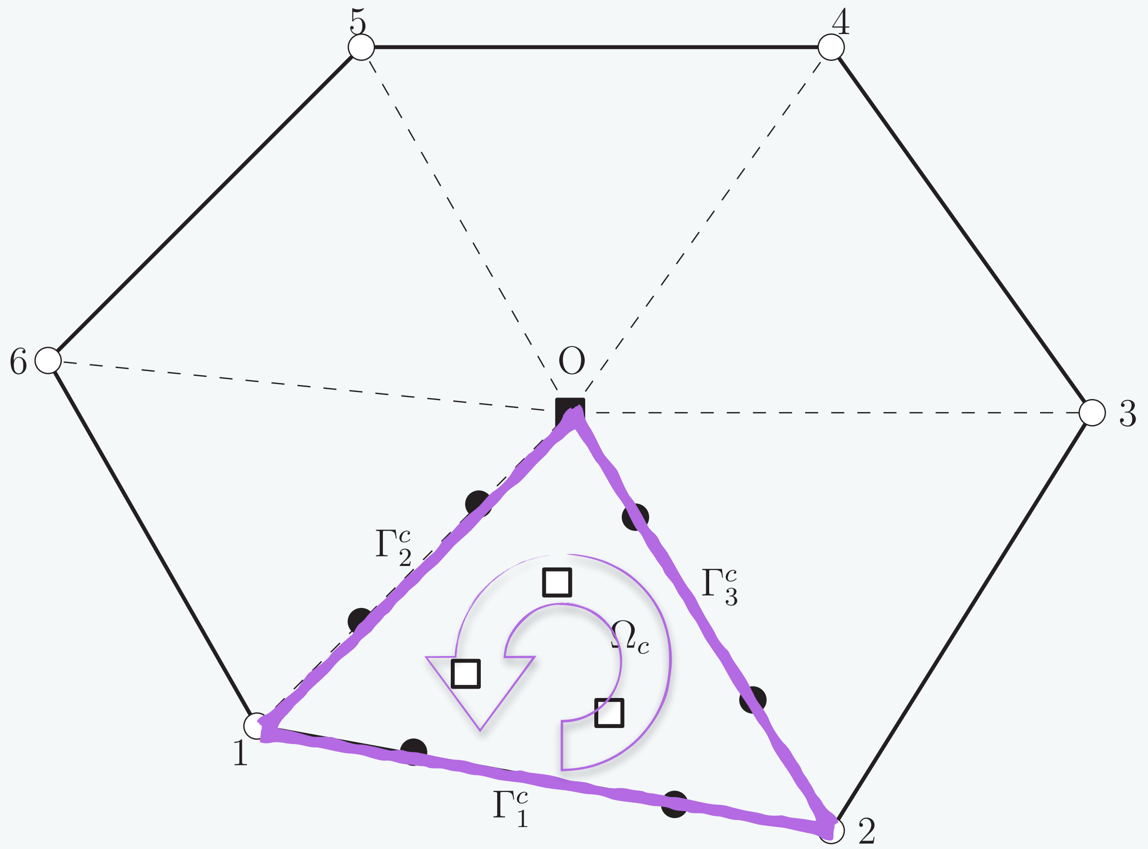
$$\tilde{\varepsilon}_{ij}^h(\mathbf{x}) = \int_{\Omega_C^h} \varepsilon_{ij}^h(\mathbf{x}) f(\mathbf{x}) \mathrm{d}V$$

$$\int_{\Omega_C^h} \phi_{a,j} f(\mathbf{x})\, \mathrm{d}V = \int_{\Gamma_C^h} \phi_a f(\mathbf{x}) n_j \, \mathrm{d}S - \int_{\Omega_C^h} \phi_a f_{,j}(\mathbf{x})\, \mathrm{d}V.$$

$$f(\mathbf{x}) = [1 \;\; x_1 \;\; x_2]^{\mathrm{T}}$$

Francis et al., IJNME, 2017.

Linear Smoothing



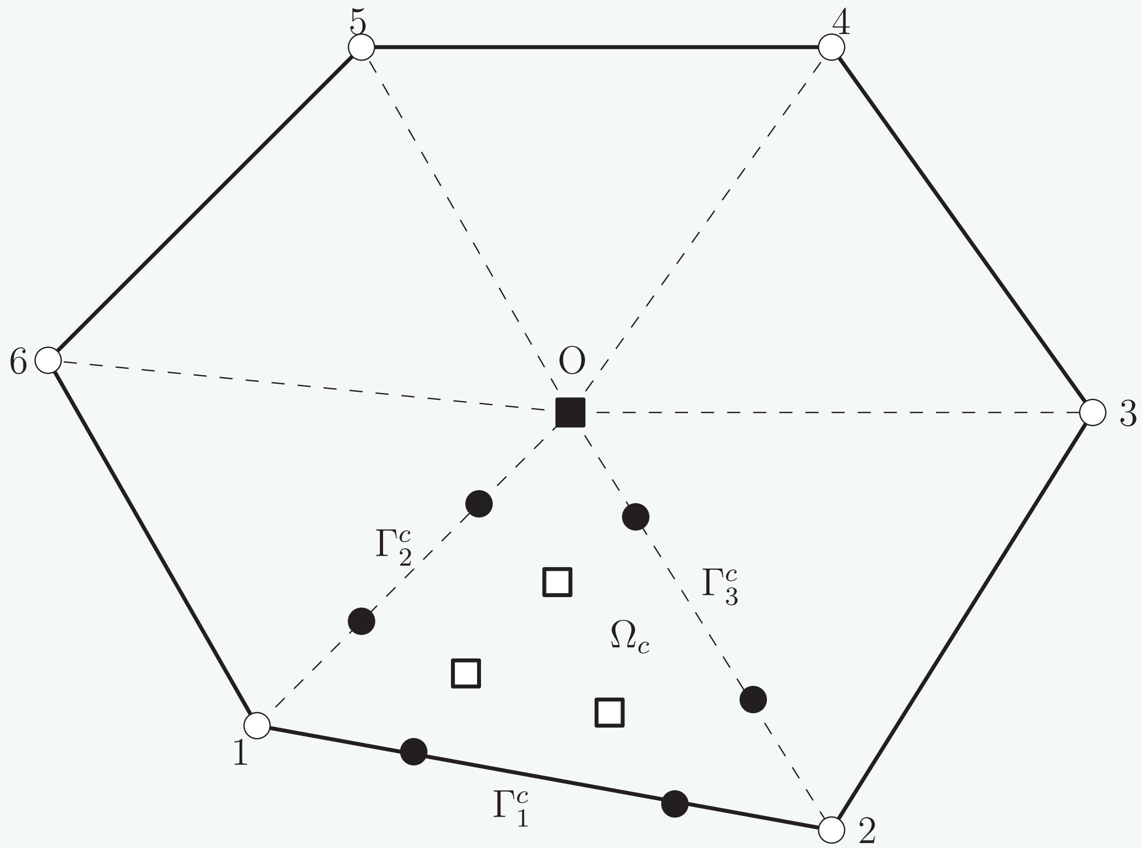
$$\tilde{\varepsilon}_{ij}^h(\mathbf{x}) = \int_{\Omega_C^h} \varepsilon_{ij}^h(\mathbf{x}) f(\mathbf{x}) \mathrm{d}V$$

$$\int_{\Omega_C^h} \phi_{a,j} f(\mathbf{x})\, \mathrm{d}V = \int_{\Gamma_C^h} \phi_a f(\mathbf{x}) n_j \, \mathrm{d}S - \int_{\Omega_C^h} \phi_a f_{,j}(\mathbf{x})\, \mathrm{d}V.$$

$$f(\mathbf{x}) = [1 \quad x_1 \quad x_2]^{\mathrm{T}}$$

Francis et al., IJNME, 2017.

Linear Smoothing



$$\tilde{\varepsilon}_{ij}^h(\mathbf{x}) = \int_{\Omega_C^h} \varepsilon_{ij}^h(\mathbf{x}) f(\mathbf{x}) \mathrm{d}V$$

$$\int_{\Omega_C^h} \phi_{a,j} f(\mathbf{x})\, \mathrm{d}V = \int_{\Gamma_C^h} \phi_a f(\mathbf{x}) n_j \,\mathrm{d}S - \int_{\Omega_C^h} \phi_a f_{,j}(\mathbf{x})\, \mathrm{d}V.$$

$$f(\mathbf{x}) = [1 \;\; x_1 \;\; x_2]^{\mathrm{T}}$$

$$\mathbf{W}\mathbf{d}_j = \mathbf{f}_j, \quad j = 1, 2$$

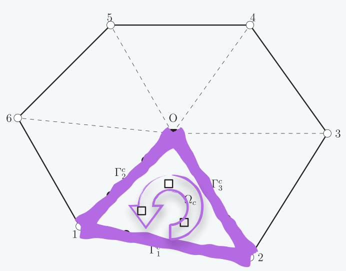
$$\mathbf{W} = \begin{bmatrix} {}^1w & {}^2w & {}^3w \\ {}^1w\,{}^1x_1 & {}^2w\,{}^2x_1 & {}^3w\,{}^3x_1 \\ {}^1w\,{}^1x_2 & {}^2w\,{}^2x_2 & {}^3w\,{}^3x_2 \end{bmatrix}$$

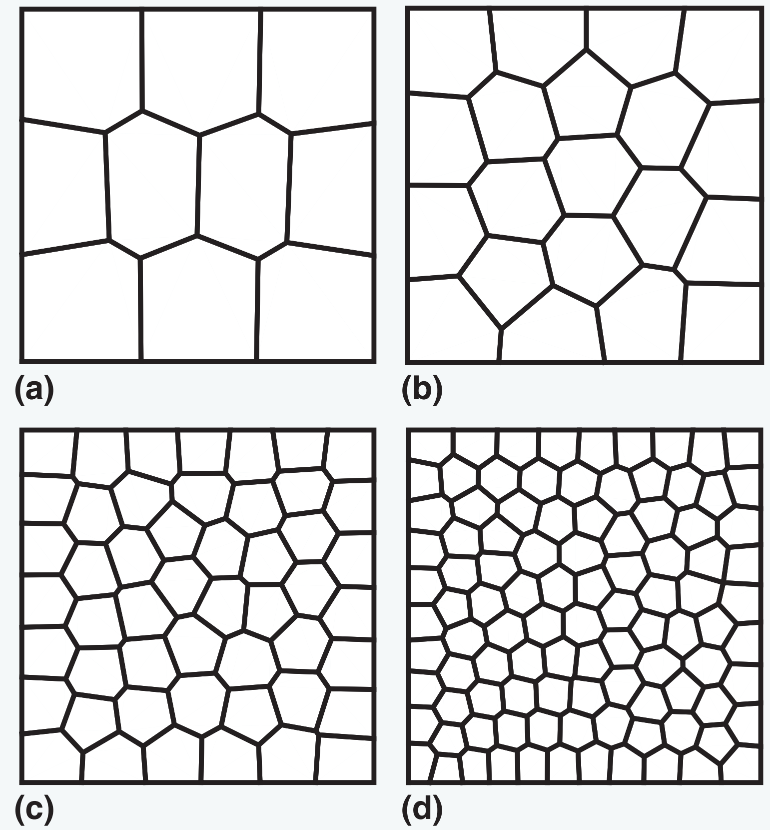
$$\mathbf{f}_1 = \begin{bmatrix} \sum_{k=1}^{3}\sum_{g=1}^{2} \phi_a\left({}_k^g\mathbf{s}\right)\, {}_kn_1\, {}_k^gv \\ \sum_{k=1}^{3}\sum_{g=1}^{2} \phi_a\left({}_k^g\mathbf{s}\right)\, {}_k^gs_1\, {}_kn_1\, {}_k^gv - \sum_{m=1}^{3} \phi_a({}^m\mathbf{r})\, {}^mw \\ \sum_{k=1}^{3}\sum_{g=1}^{2} \phi_a\left({}_k^g\mathbf{s}\right)\, {}_k^gs_2\, {}_kn_1\, {}_k^gv \end{bmatrix}$$

Francis et al., IJNME, 2017.

Test for linear consistency

$$\begin{pmatrix} \hat{u} \\ \hat{v} \end{pmatrix} = \begin{pmatrix} 0.1 + 0.1x + 0.2y \\ 0.05 + 0.15x + 0.1y \end{pmatrix}$$





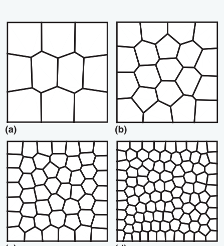
	CS-Poly2D (linear)		LS-Poly2D (linear)	
Mesh	L^2	H^1	L^2	H^1
a	1.7334×10^{-07}	2.3328×10^{-05}	5.3835×10^{-14}	2.8388×10^{-11}
b	1.6994×10^{-07}	3.4094×10^{-05}	1.9255×10^{-13}	4.4373×10^{-11}
c	7.2017×10^{-07}	2.2573×10^{-04}	2.0030×10^{-13}	7.0017×10^{-11}
d	7.4144×10^{-07}	2.5773×10^{-04}	2.9567×10^{-13}	1.0199×10^{-10}

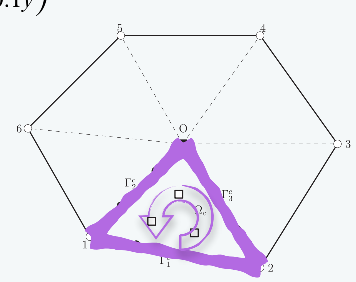
Test for linear consistency

Mesh	CS-Poly2D (linear)		LS-Poly2D (linear)	
	L^2	H^1	L^2	H^1
a	1.7334×10^{-07}	2.3328×10^{-05}	5.3835×10^{-14}	2.8388×10^{-11}
b	1.6994×10^{-07}	3.4094×10^{-05}	1.9255×10^{-13}	4.4373×10^{-11}
c	7.2017×10^{-07}	2.2573×10^{-04}	2.0030×10^{-13}	7.0017×10^{-11}
d	7.4144×10^{-07}	2.5773×10^{-04}	2.9567×10^{-13}	1.0199×10^{-10}

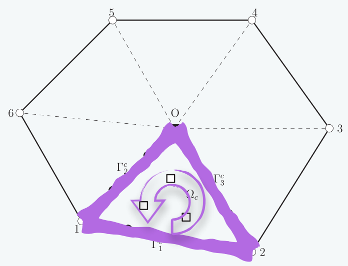
Linear smoothing passes the patch test
Constant smoothing does not

$$\begin{pmatrix}\hat{u}\\ \hat{v}\end{pmatrix} = \begin{pmatrix}0.1 + 0.1x + 0.2y\\ 0.05 + 0.15x + 0.1y\end{pmatrix}$$

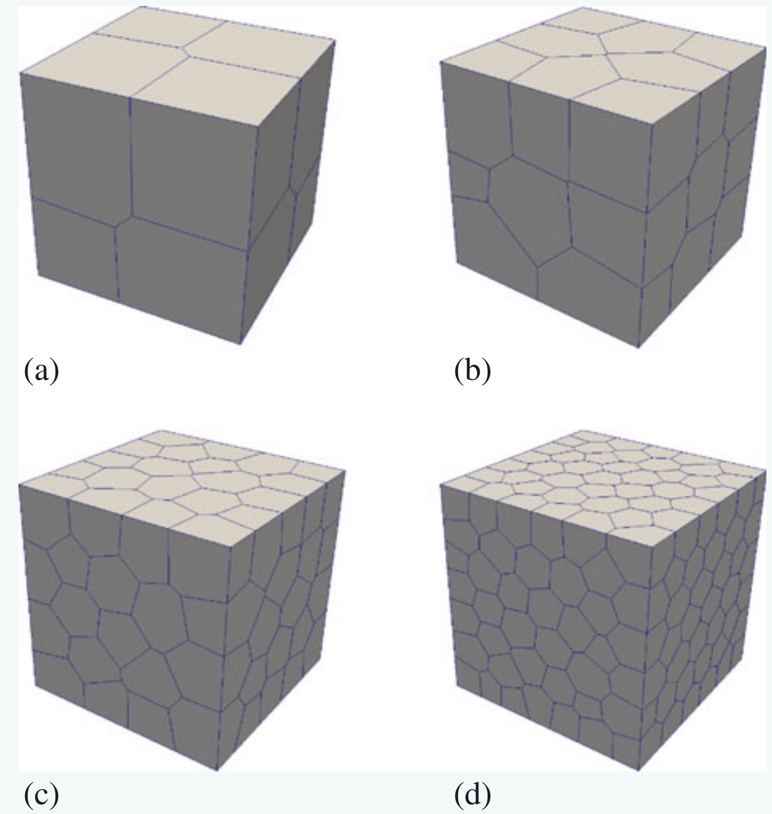




Test of linear consistency - 3D



$$\begin{pmatrix} \hat{u} \\ \hat{v} \\ \hat{w} \end{pmatrix} = \begin{pmatrix} 0.1 + 0.1x + 0.2y + 0.2z \\ 0.05 + 0.15x + 0.1y + 0.2z \\ 0.05 + 0.1x + 0.2y + 0.2z \end{pmatrix}$$



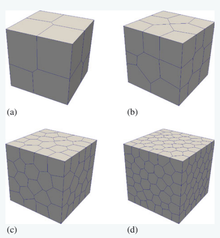
	LS-H8		Mesh	LS-Poly3D	
Mesh	L^2	H^1	(c.f. Figure 8)	L^2	H^1
2×2×2	2.5242×10^{-16}	2.4820×10^{-12}	a	2.0280×10^{-12}	3.3428×10^{-10}
4×4×4	7.9454×10^{-16}	4.9945×10^{-12}	b	1.9218×10^{-12}	1.7529×10^{-10}
8×8×8	2.9384×10^{-16}	1.0012×10^{-12}	c	2.6660×10^{-12}	4.9320×10^{-10}
16×16×16	8.9235×10^{-16}	2.0093×10^{-12}	d	3.2074×10^{-12}	3.1083×10^{-10}

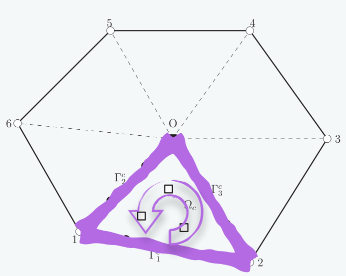
Test of linear consistency - 3D

	LS-H8		Mesh	LS-Poly3D	
Mesh	L^2	H^1	(c.f. Figure 8)	L^2	H^1
2×2×2	2.5242×10^{-16}	2.4820×10^{-12}	a	2.0280×10^{-12}	3.3428×10^{-10}
4×4×4	7.9454×10^{-16}	4.9945×10^{-12}	b	1.9218×10^{-12}	1.7529×10^{-10}
8×8×8	2.9384×10^{-16}	1.0012×10^{-12}	c	2.6660×10^{-12}	4.9320×10^{-10}
16×16×16	8.9235×10^{-16}	2.0093×10^{-12}	d	3.2074×10^{-12}	3.1083×10^{-10}

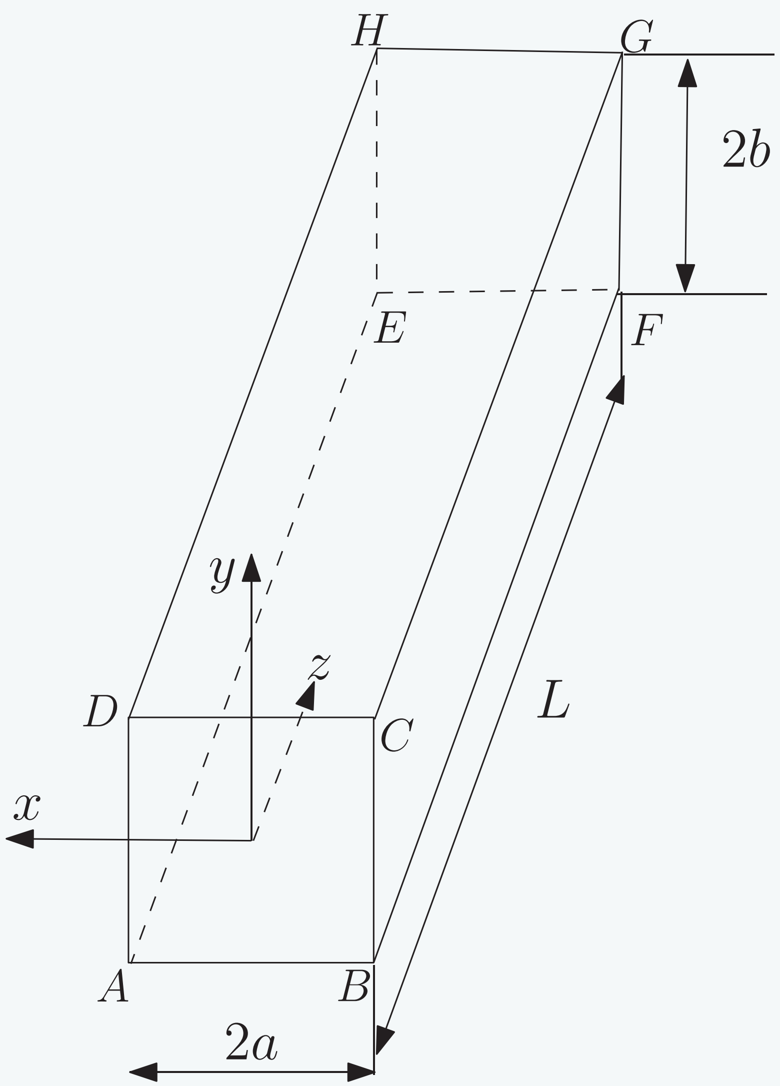
Linear smoothing passes the patch test, also in 3D

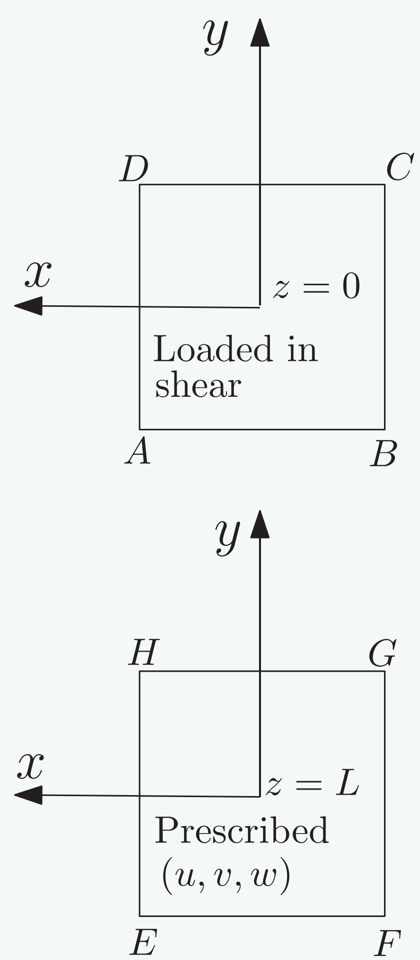
$$\begin{pmatrix}\hat{u}\\ \hat{v}\\ \hat{w}\end{pmatrix} = \begin{pmatrix}0.1+0.1x+0.2y+0.2z\\ 0.05+0.15x+0.1y+0.2z\\ 0.05+0.1x+0.2y+0.2z\end{pmatrix}$$

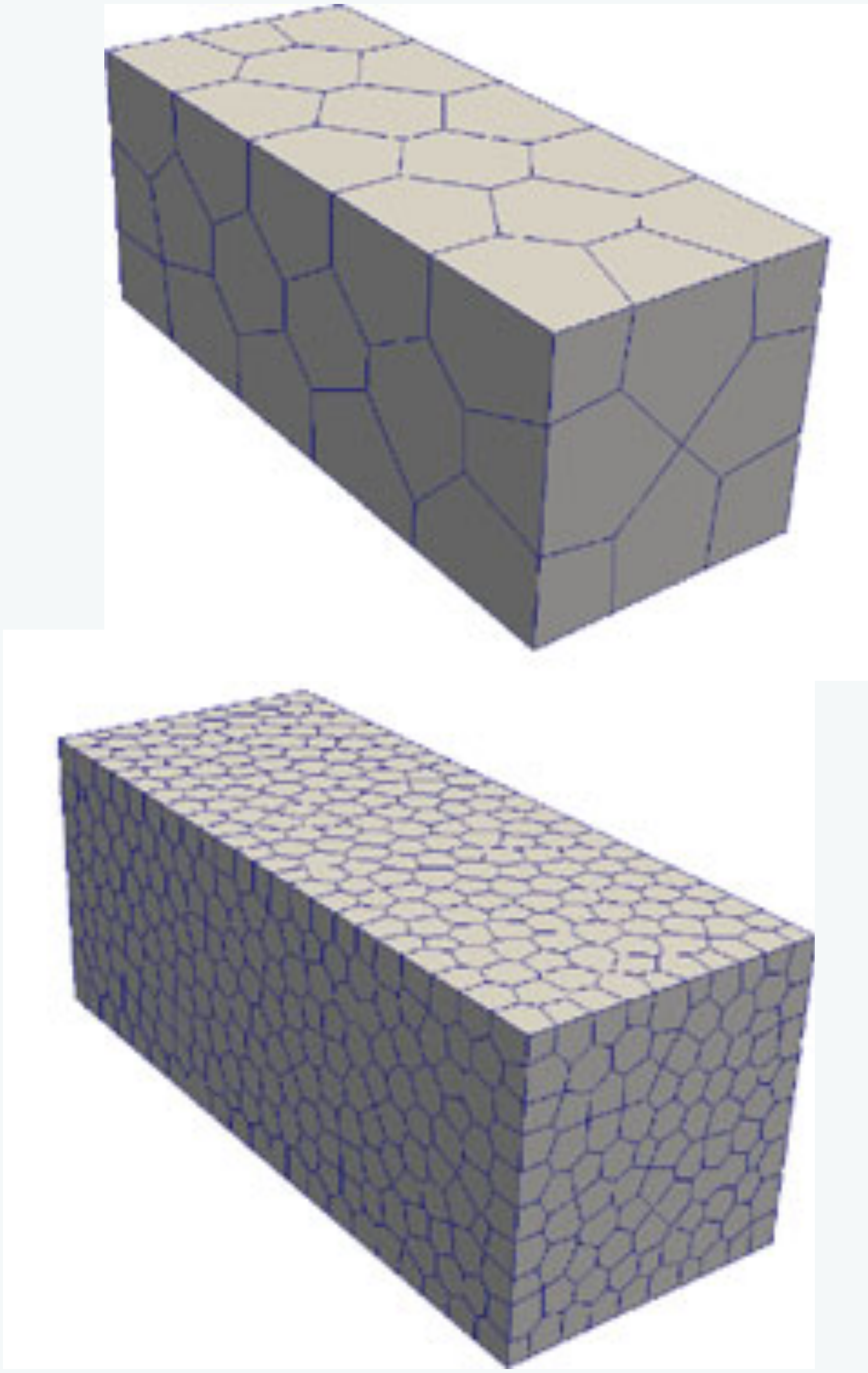




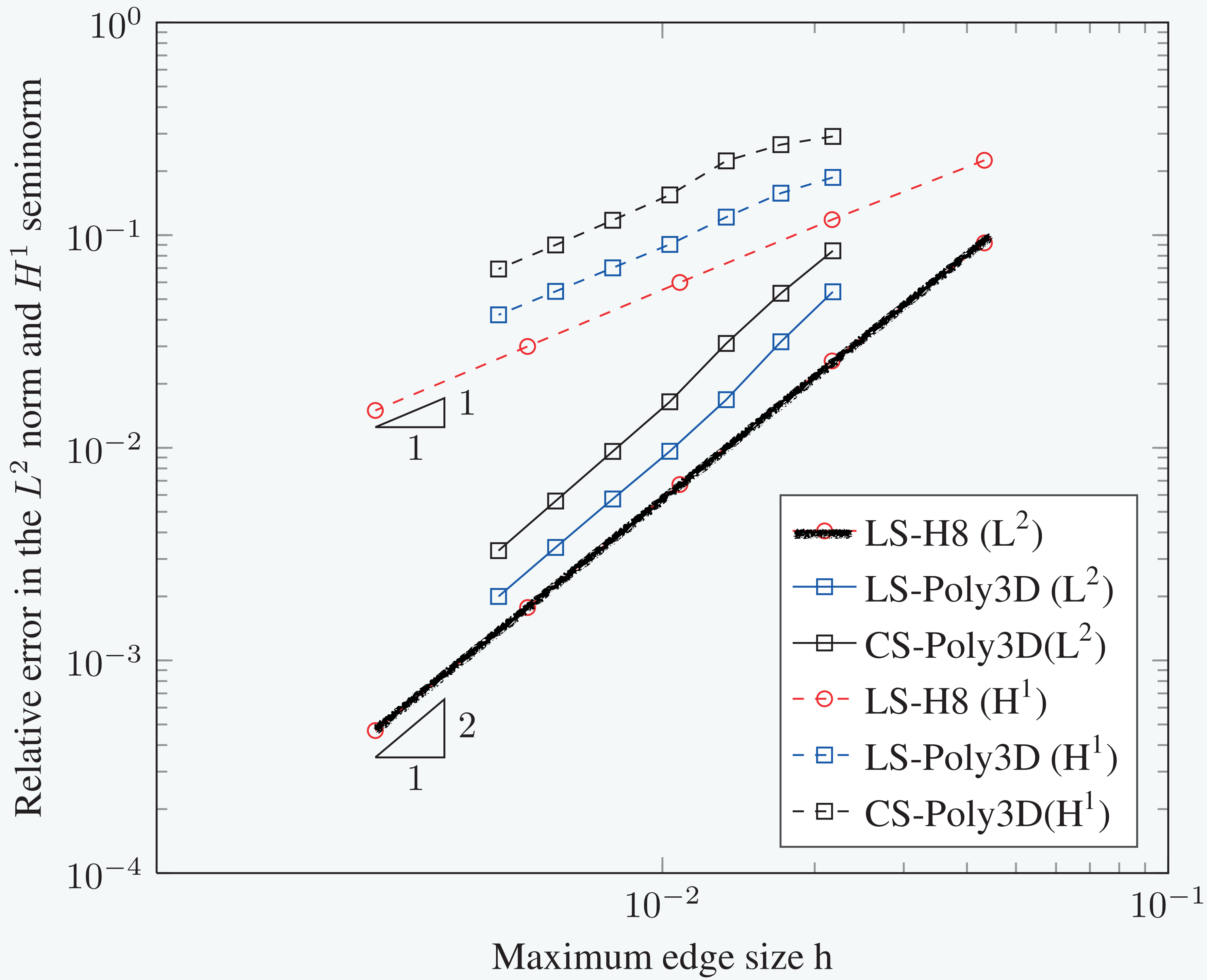
Cantilever bending



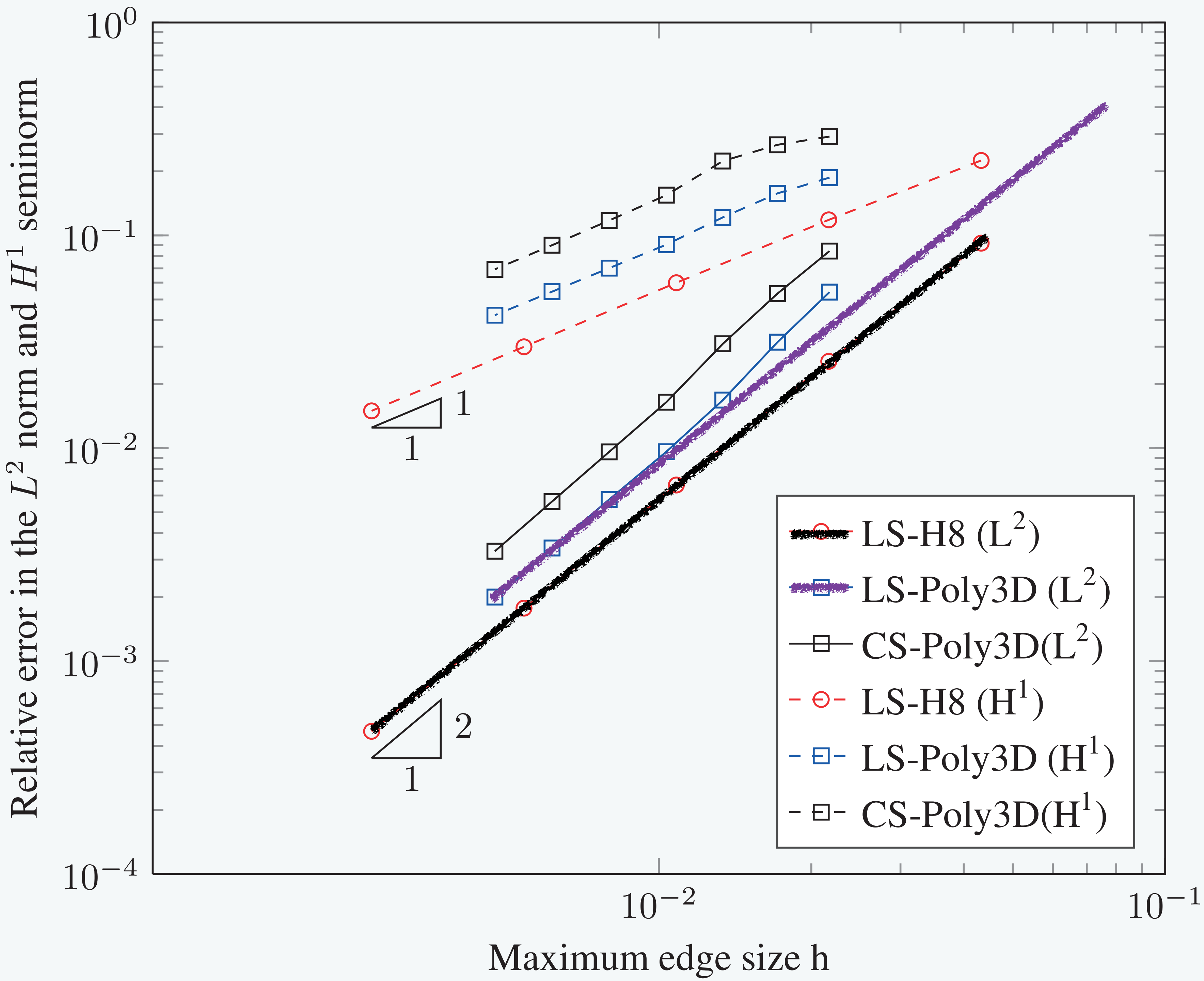




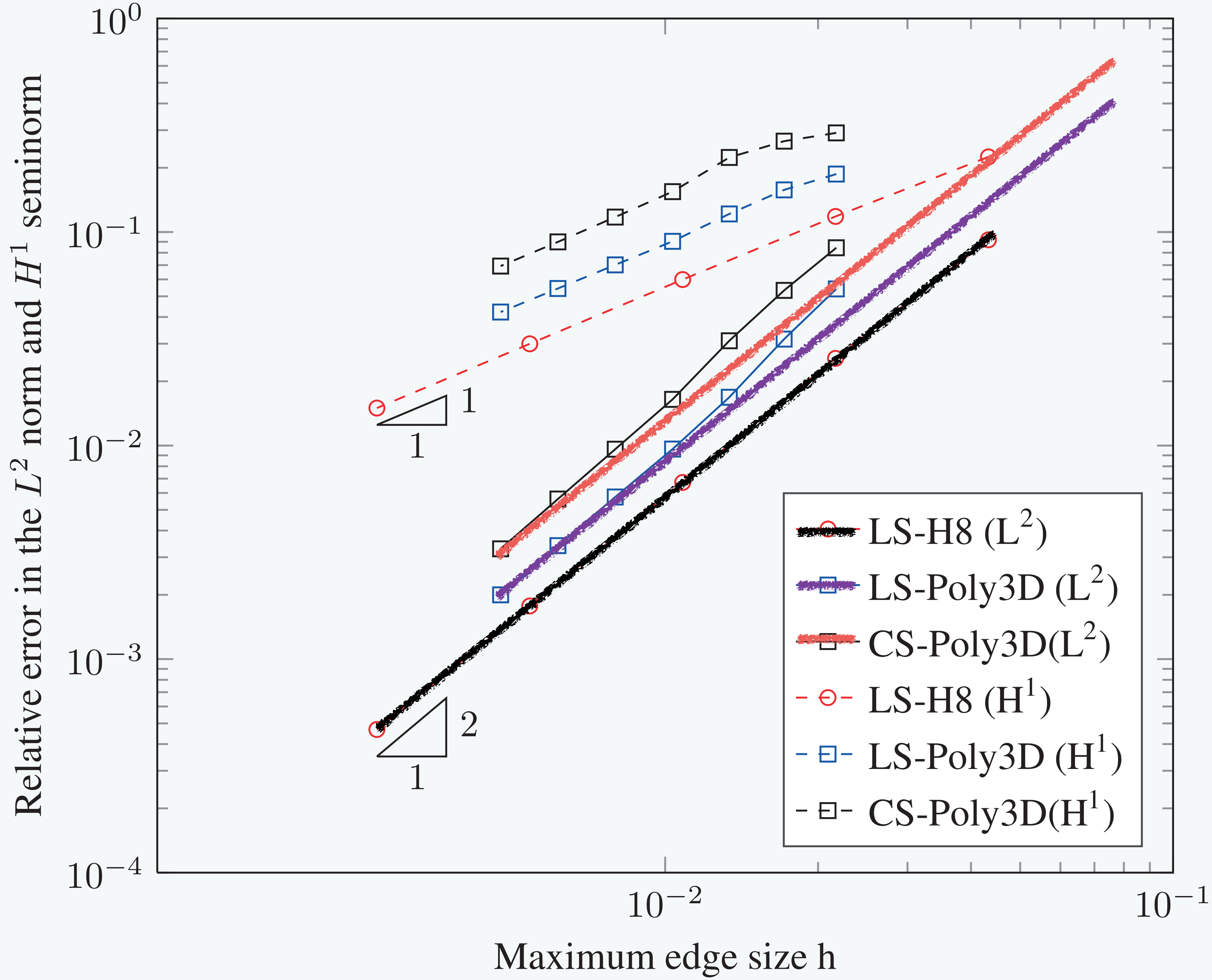
Cantilever bending



Cantilever bending



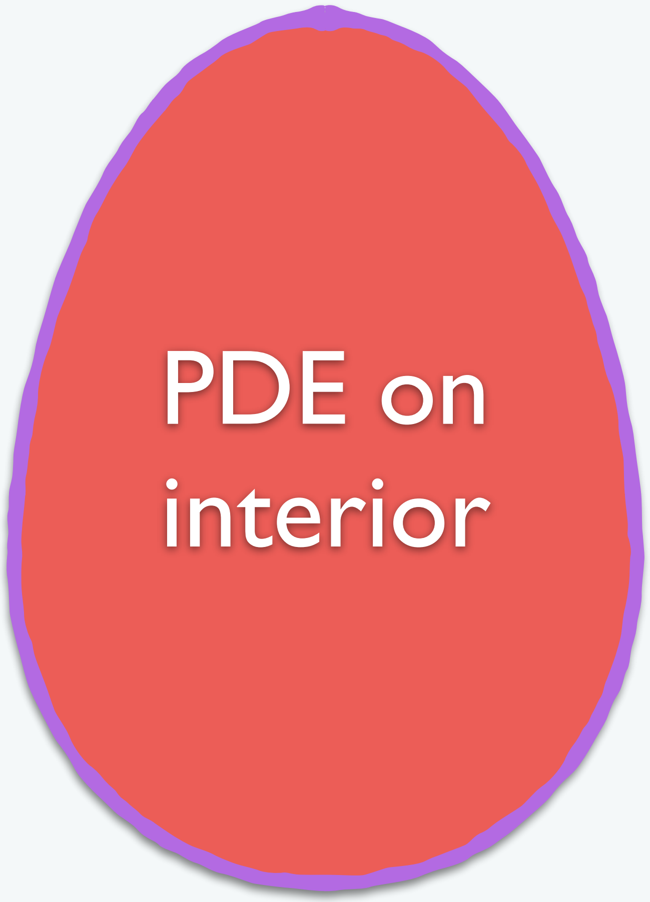
Cantilever bending



SBFEM

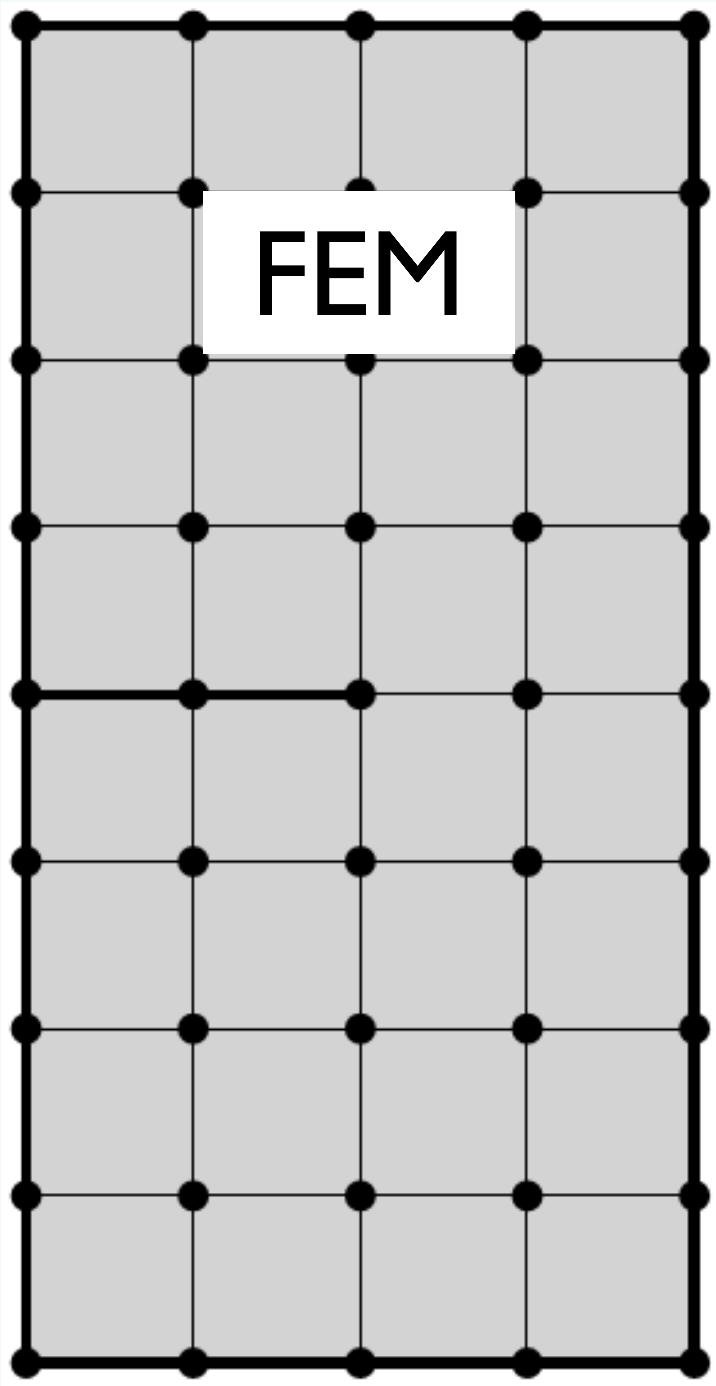
- A "niche" technique developed originally for dynamic soil-structure interaction analysis by Song and Wolf @EPFL
- Has been applied to several other fields such as fracture mechanics, fluid mechanics, fluid-structure interaction, acoustics, electromagnetism, etc.
- A *semi-analytical* procedure
- Only the boundary is discretized
- No requirement for fundamental solutions
- Appeared first in 1997, CMAME

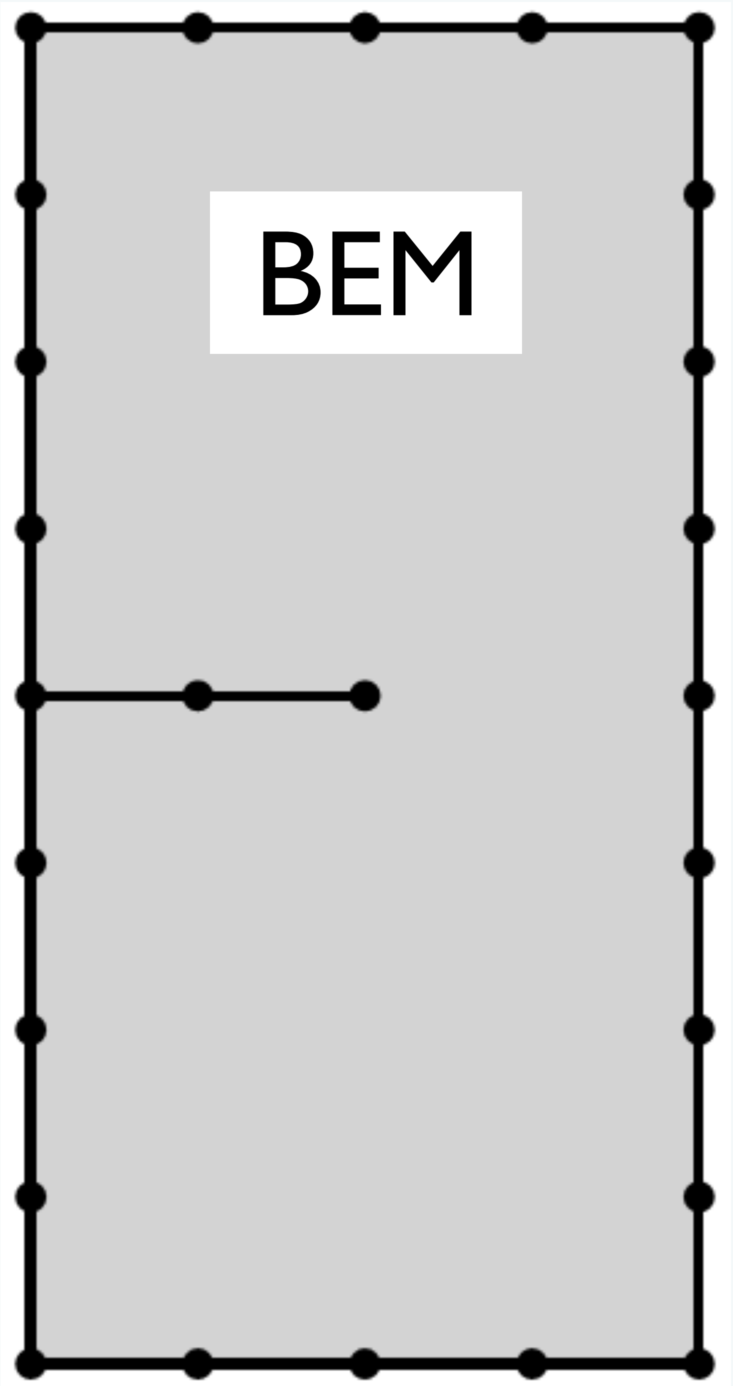
SBFEM



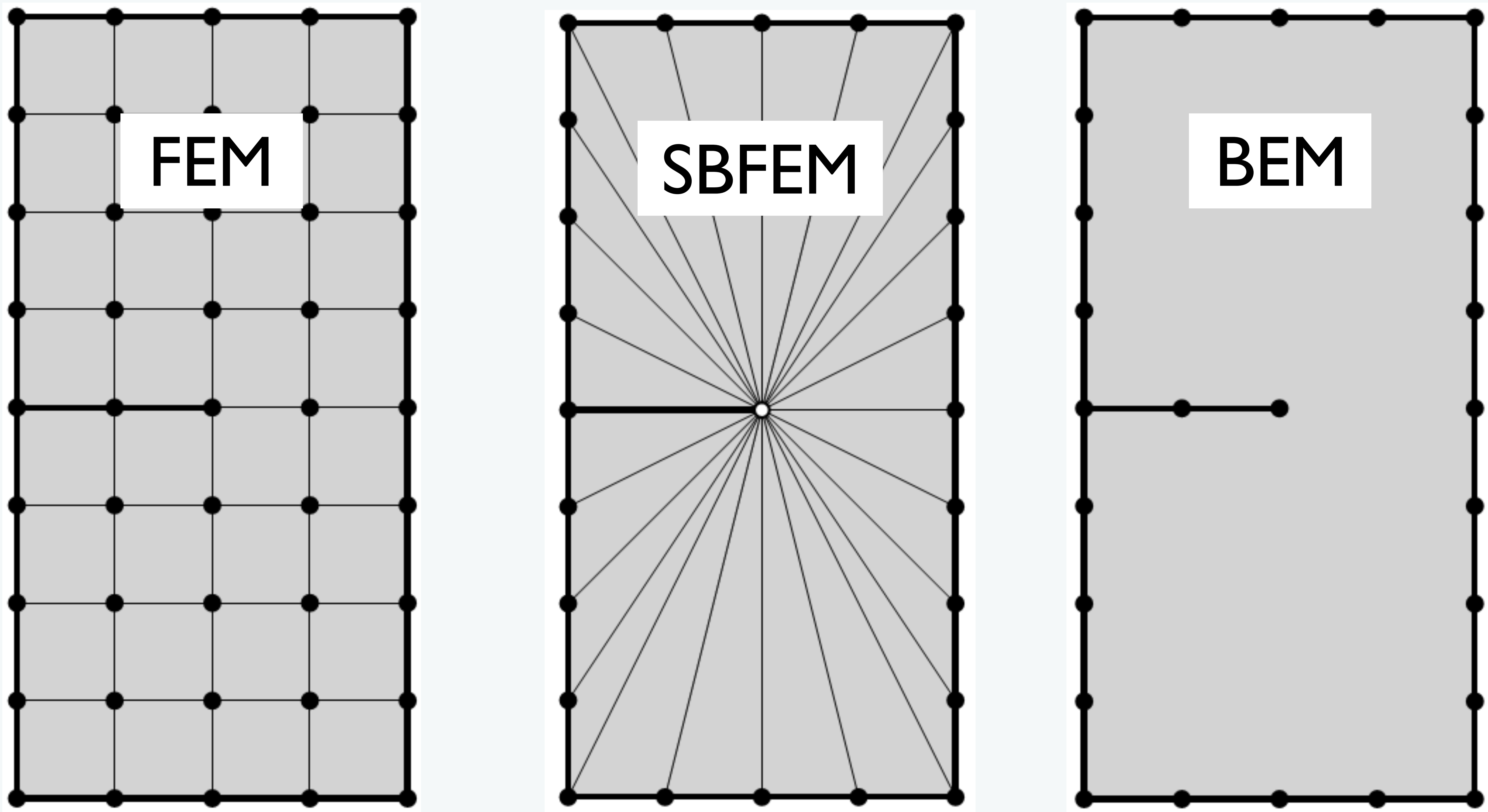
SBFEM

Conceptual comparison





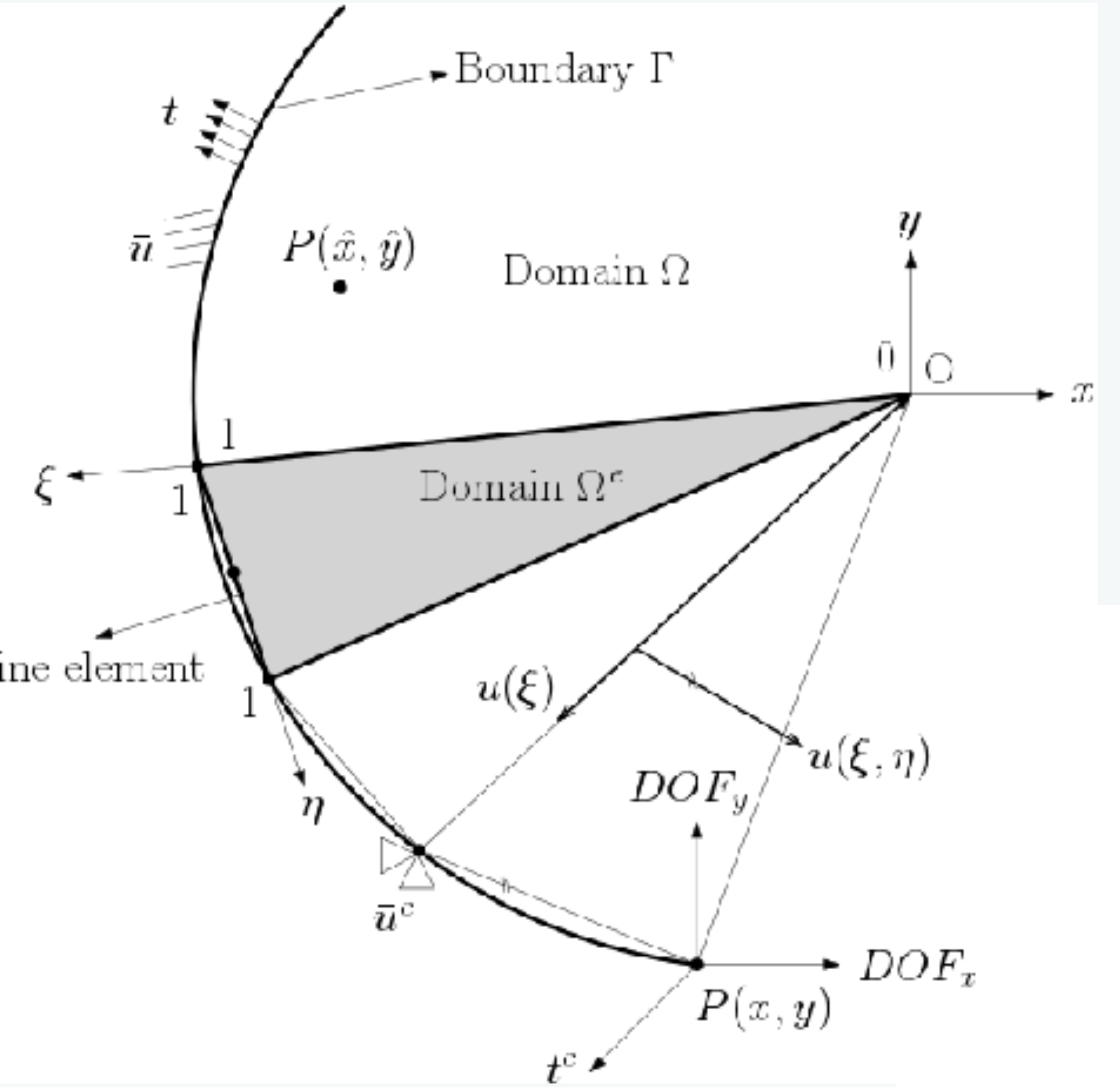
Conceptual comparison



Scaled boundary FEM - advantages

	FEM	BEM	SBFEM
Reduction of spatial dimension by one		✓	✓
Analytical solution inside the domain			✓
No fundamental solution	✓		✓
No discretisation of material interfaces			✓
Symmetric static and dynamic stiffness matrix	✓		✓
Straightforward calculation of SIF			✓
Seamless integration with FEM			✓
Arbitrary approximation orders in neighbouring domains			✓

Scaled boundary FEM



Geometry $\mathbf{x} = \mathbf{N}(\eta)\mathbf{x}_b$

Displacements $\mathbf{u}(\xi, \eta) = \mathbf{N}(\eta)\mathbf{u}(\xi)$

Strain $\boldsymbol{\varepsilon}(\xi, \eta) = \mathbf{B}(\eta)\mathbf{u}(\xi)$

Virtual work

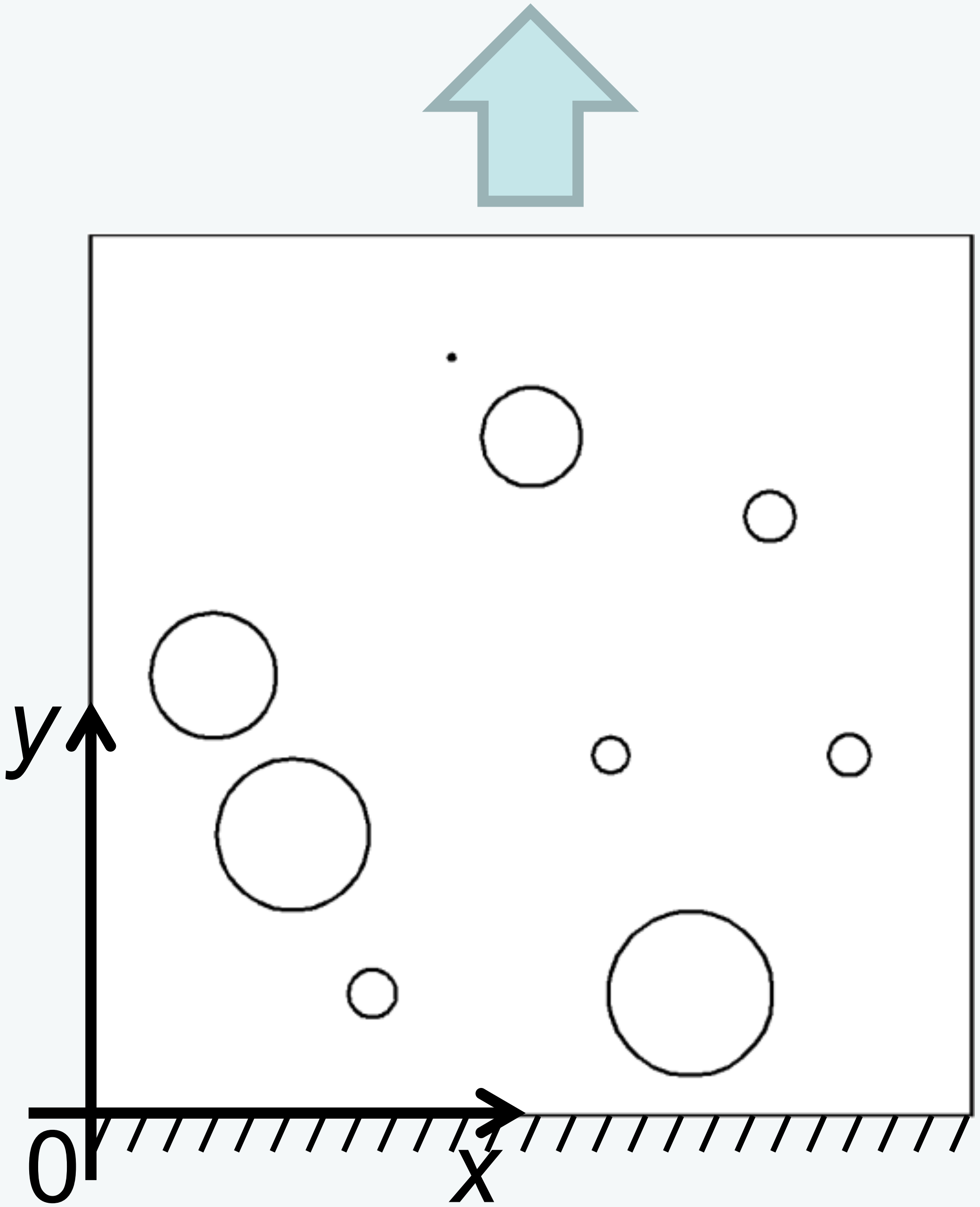
SBFEM equations

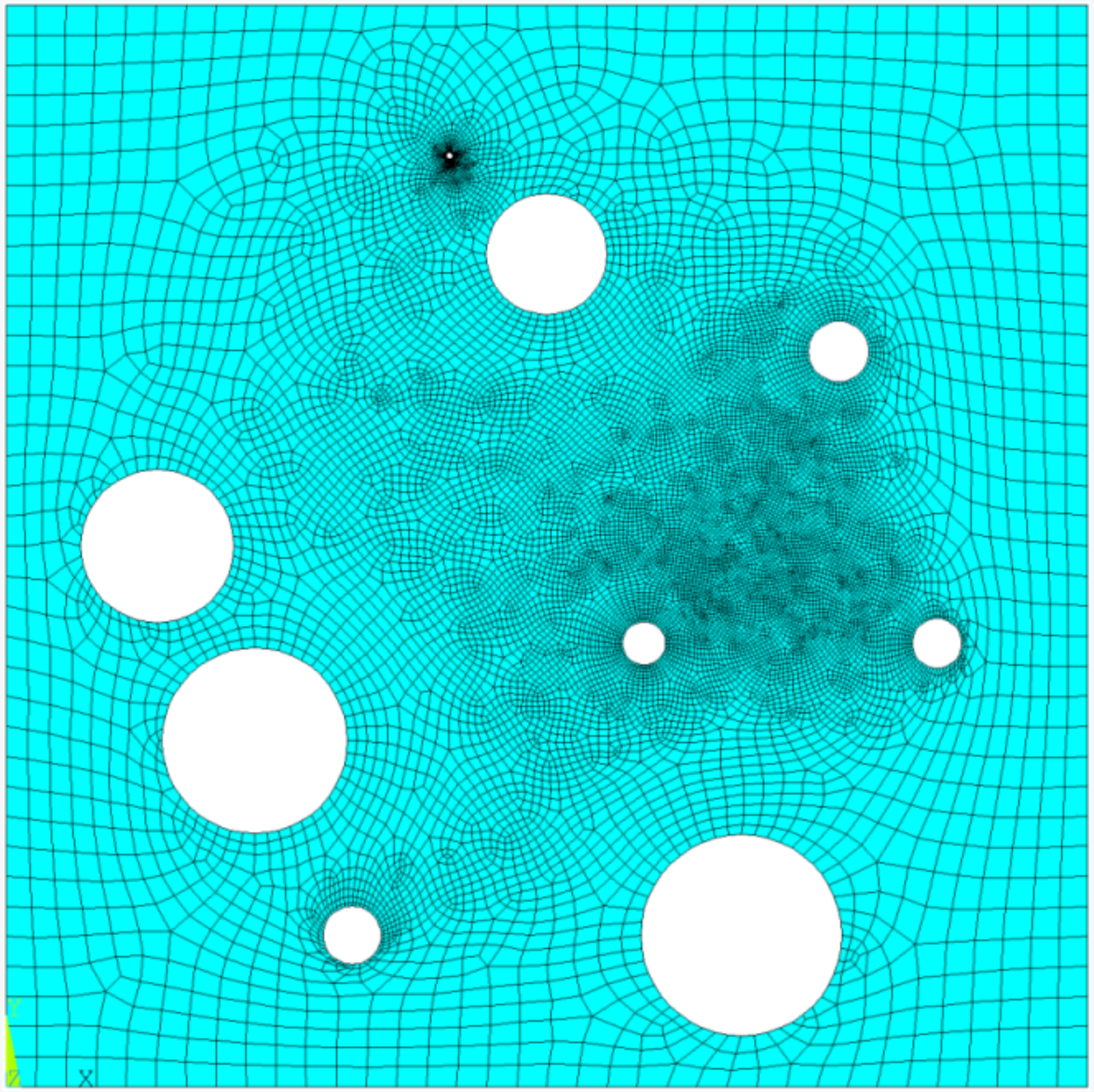
Stiffness matrix

$$\mathbf{K} = \int_{\partial\Omega} \mathbf{B}^{\mathrm{T}}\mathbb{C}\mathbf{B}\ \mathrm{d}\Omega$$

Scaling requirement on geometry: whole boundary should be directly visible from the scaling centre

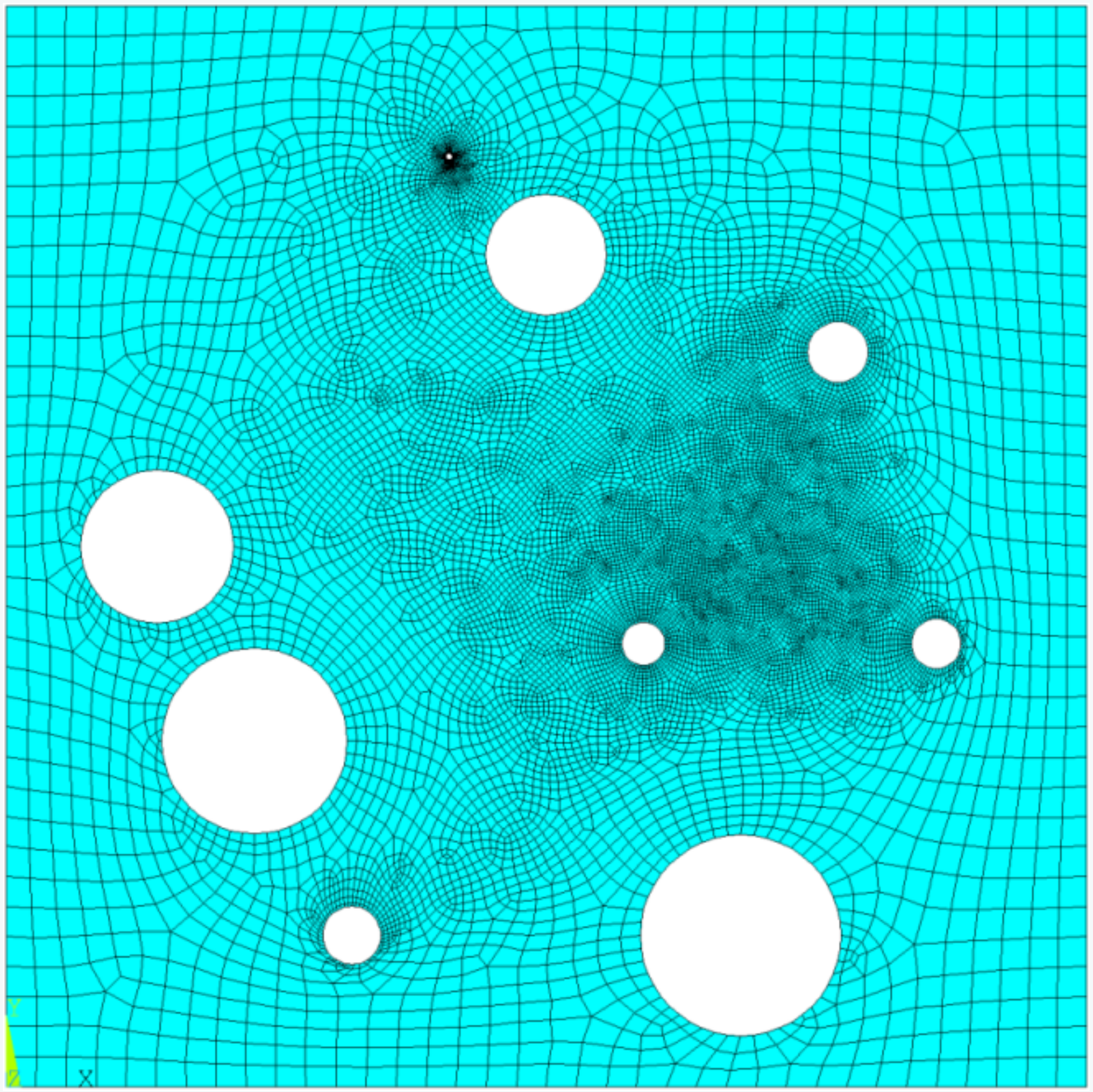
Plate with internal holes



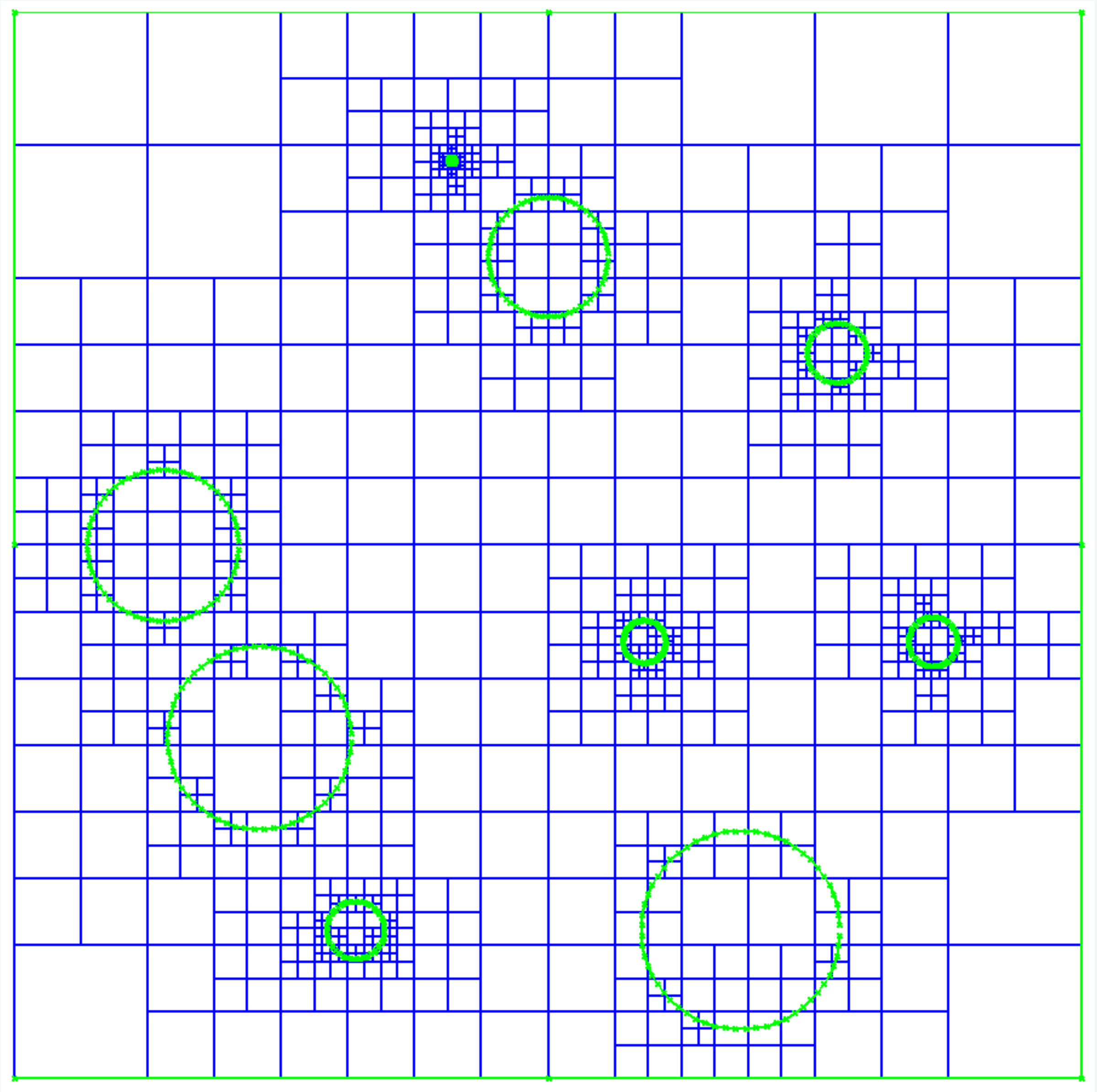


ANSYS
13,650 nodes

Quadtree

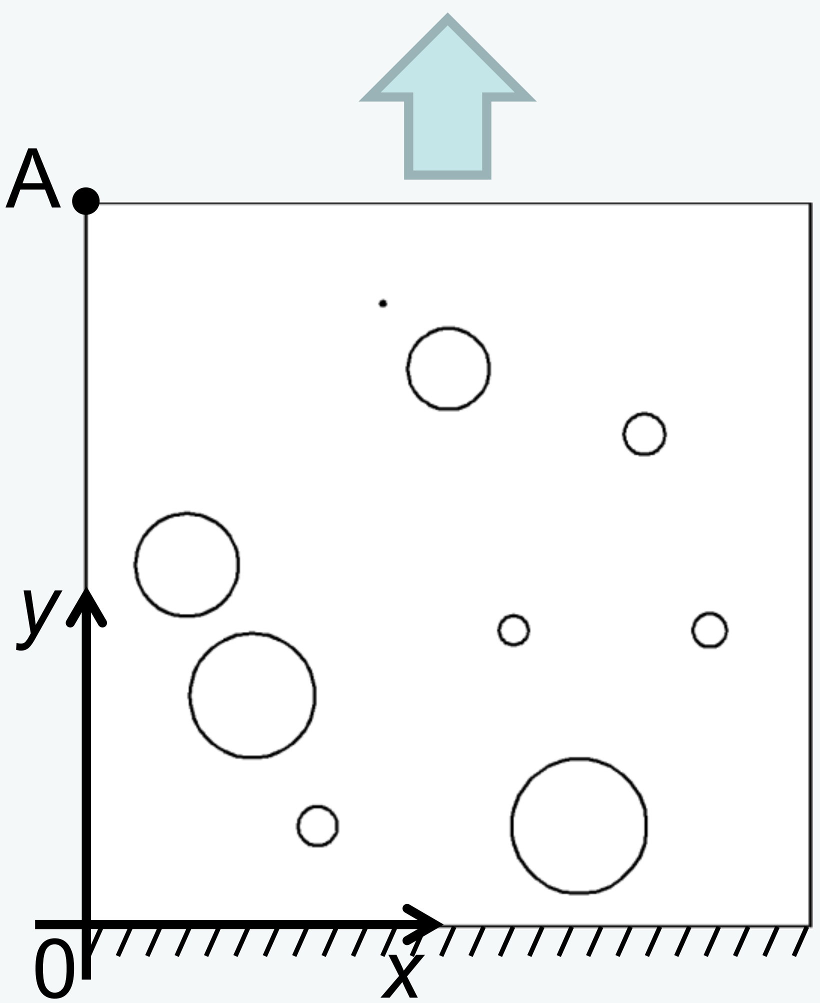


ANSYS
13,650 nodes

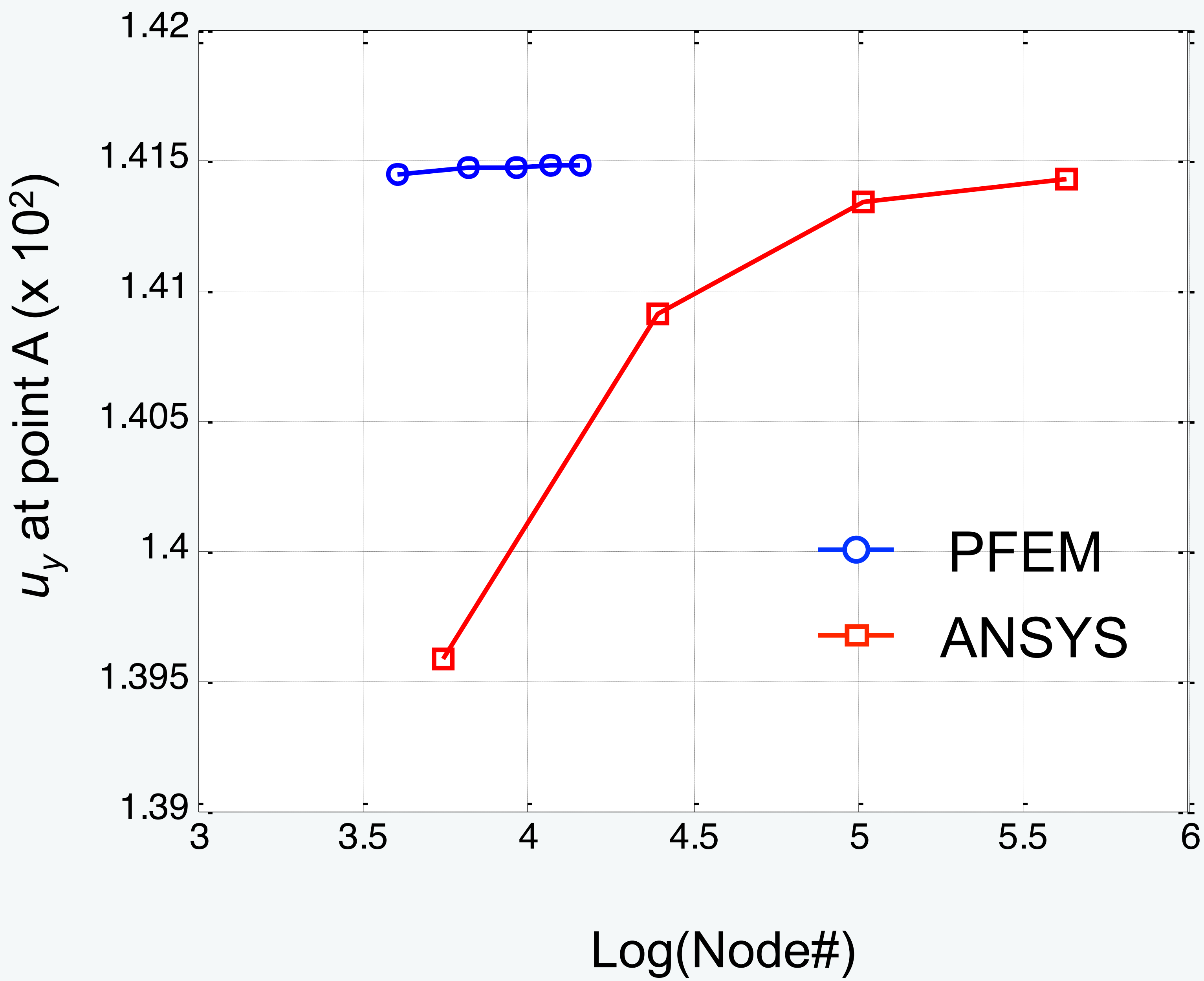


Quad tree
6,619 nodes

Quadtree

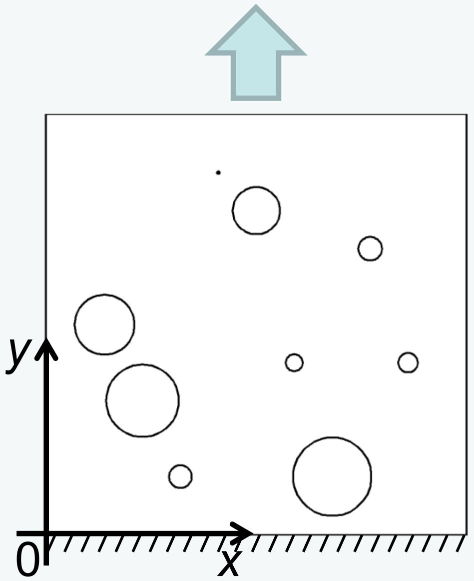


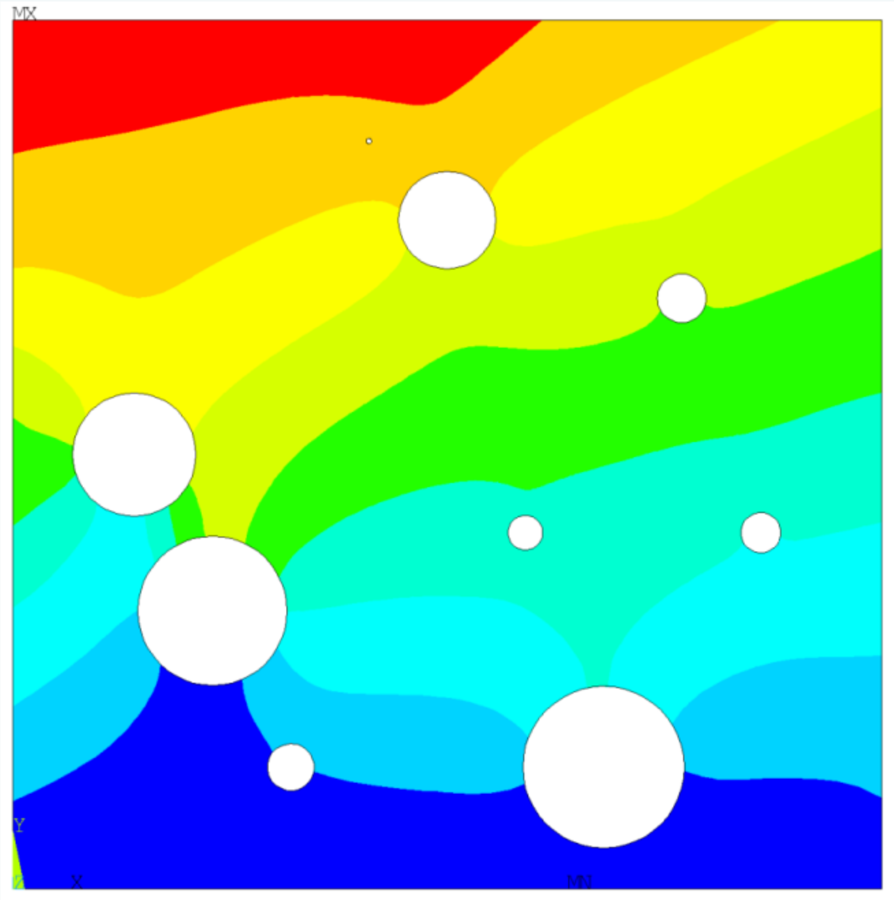
A
y
0
x



u_y at point A (x 10^2)
1.42
1.415
1.41
1.405
1.4
1.395
1.39
3
3.5
4
4.5
5
5.5
6
Log(Node#)
PFEM
ANSYS

Displacement compassion

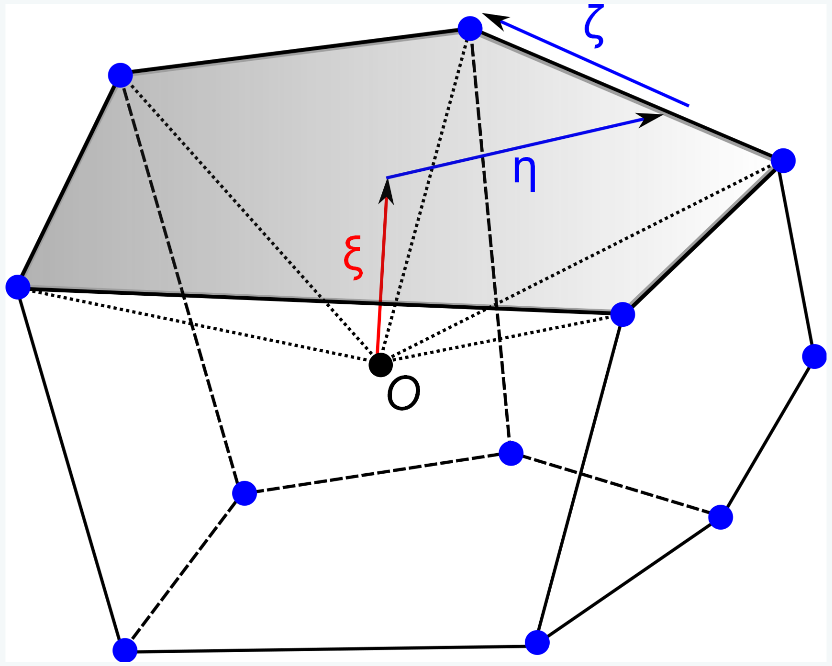




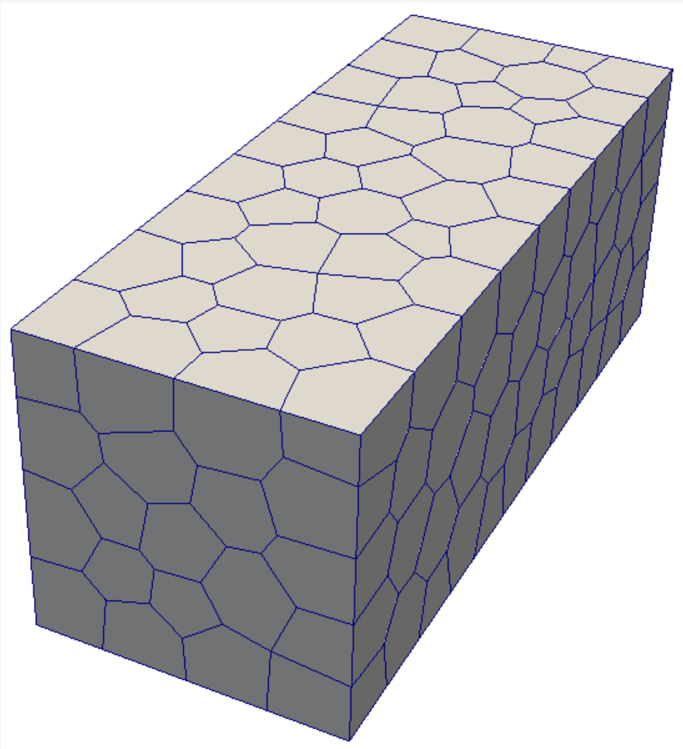
ANSYS
13,650 nodes

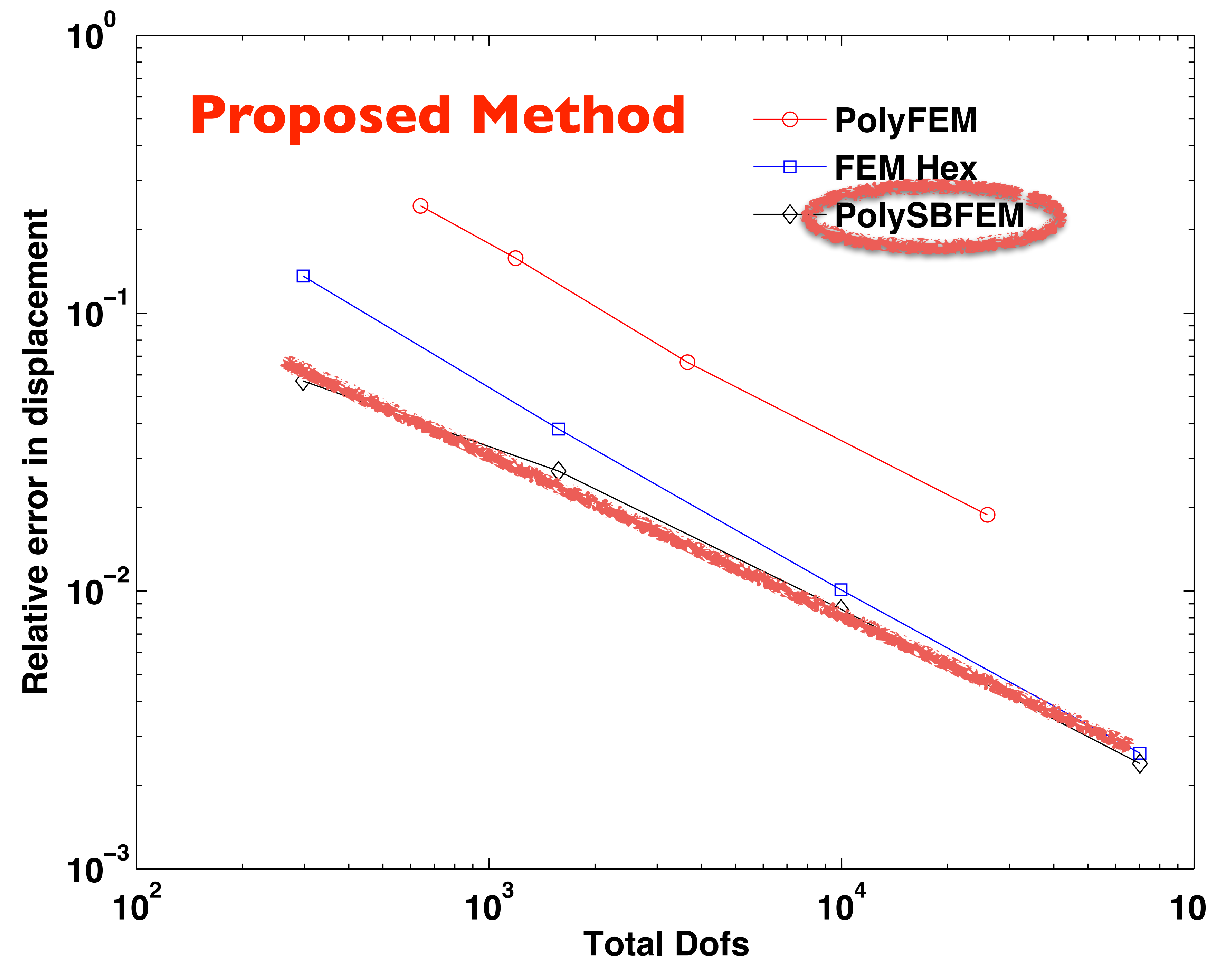
Quad tree
6,619 nodes

Extension to polyhedra

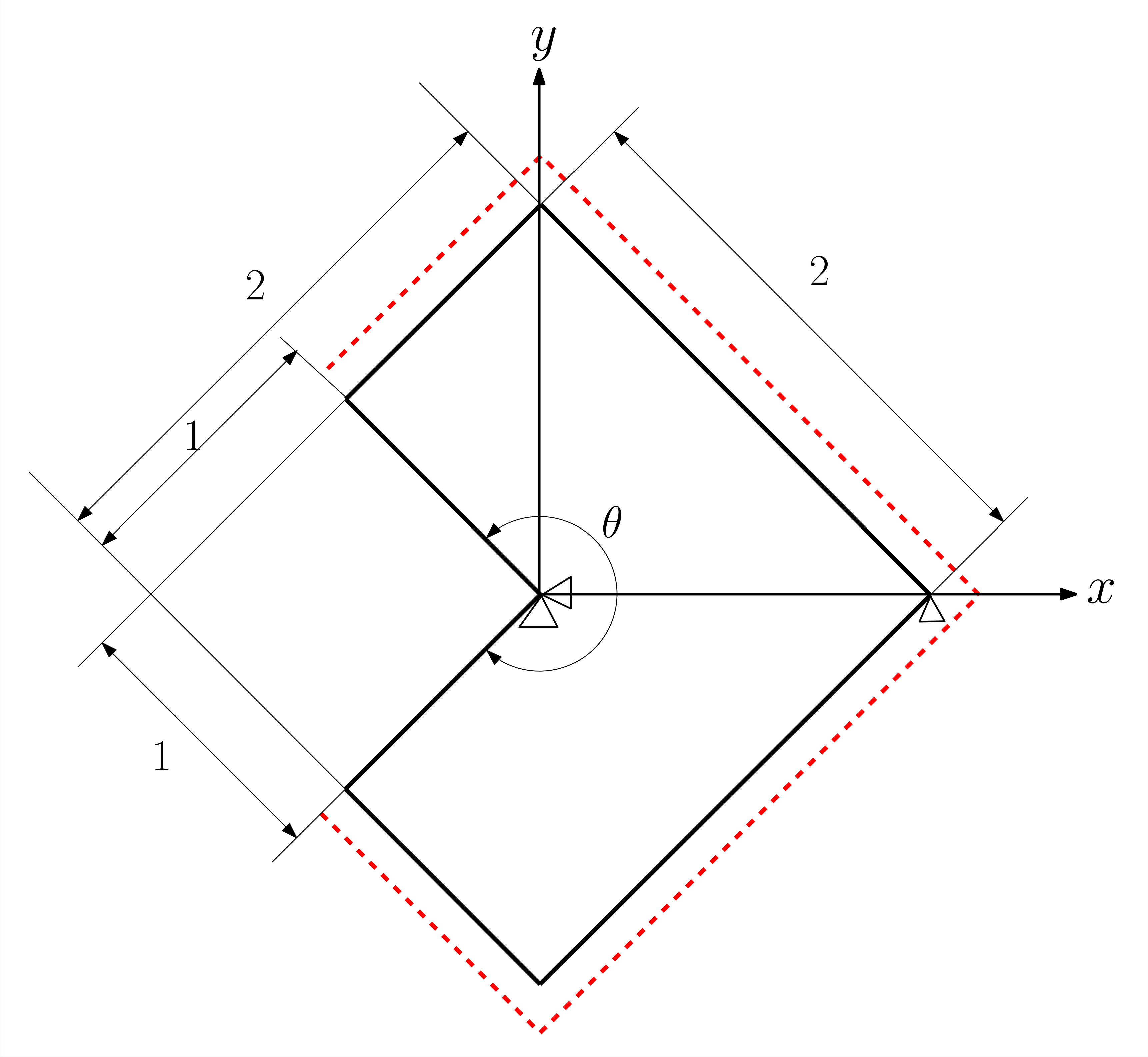


$$u(\xi,\eta,\zeta) = N(\eta,\zeta)u(\xi)$$

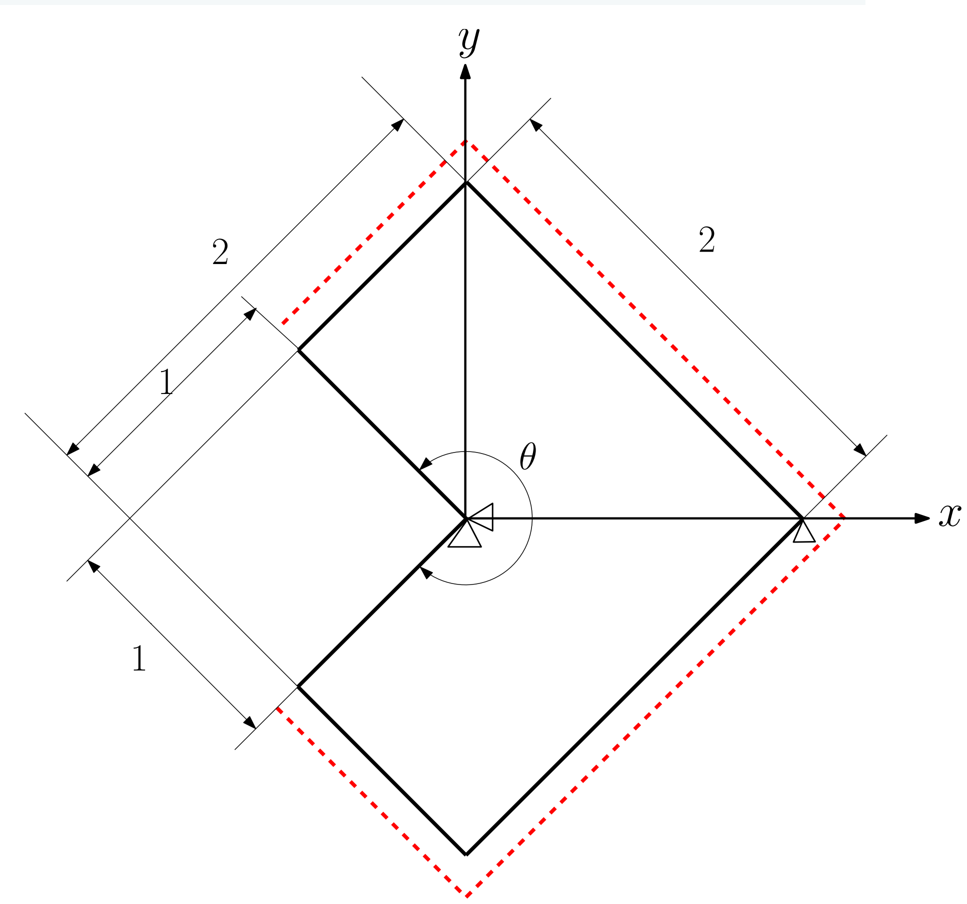


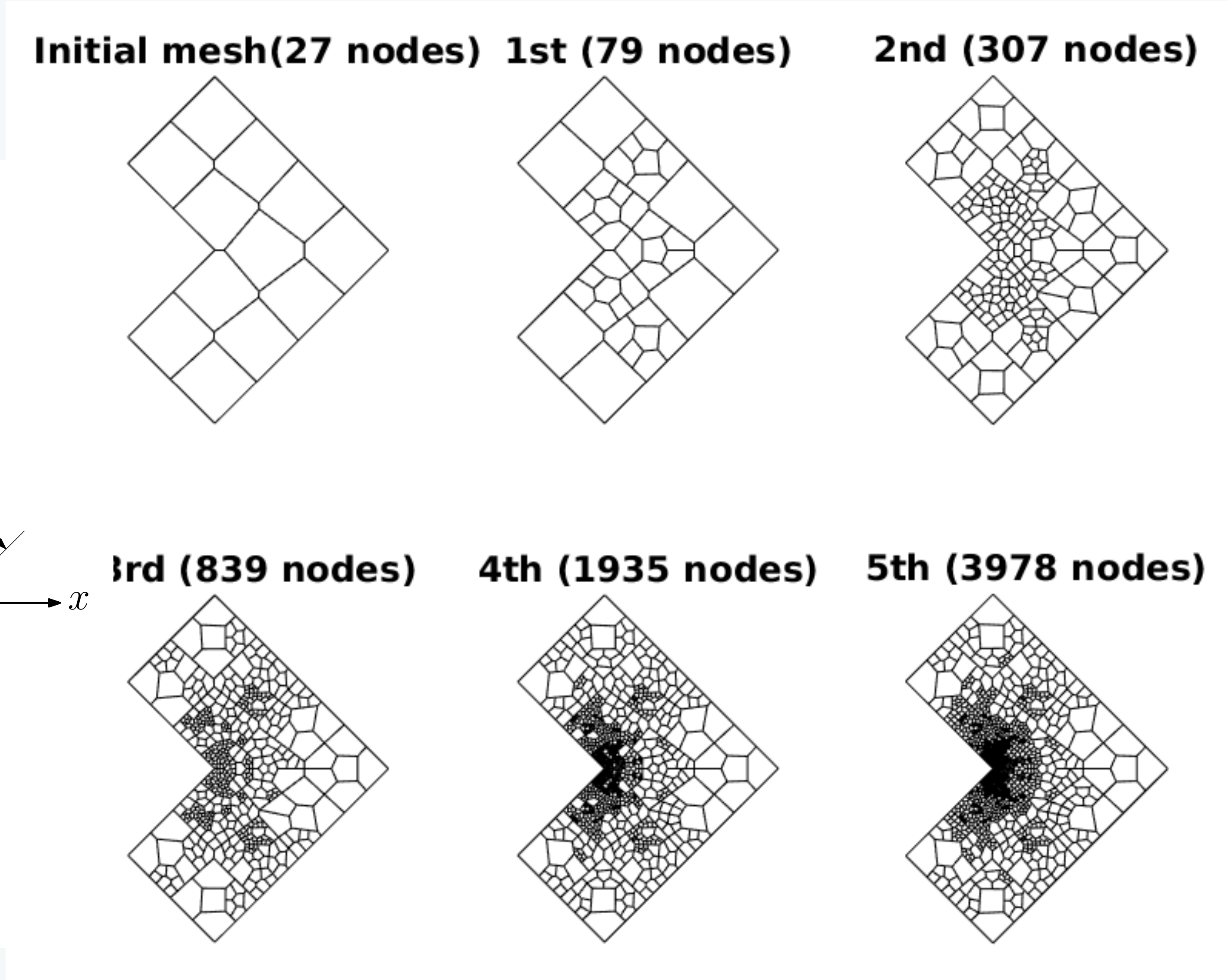


Novel error indicator



Novel error indicator

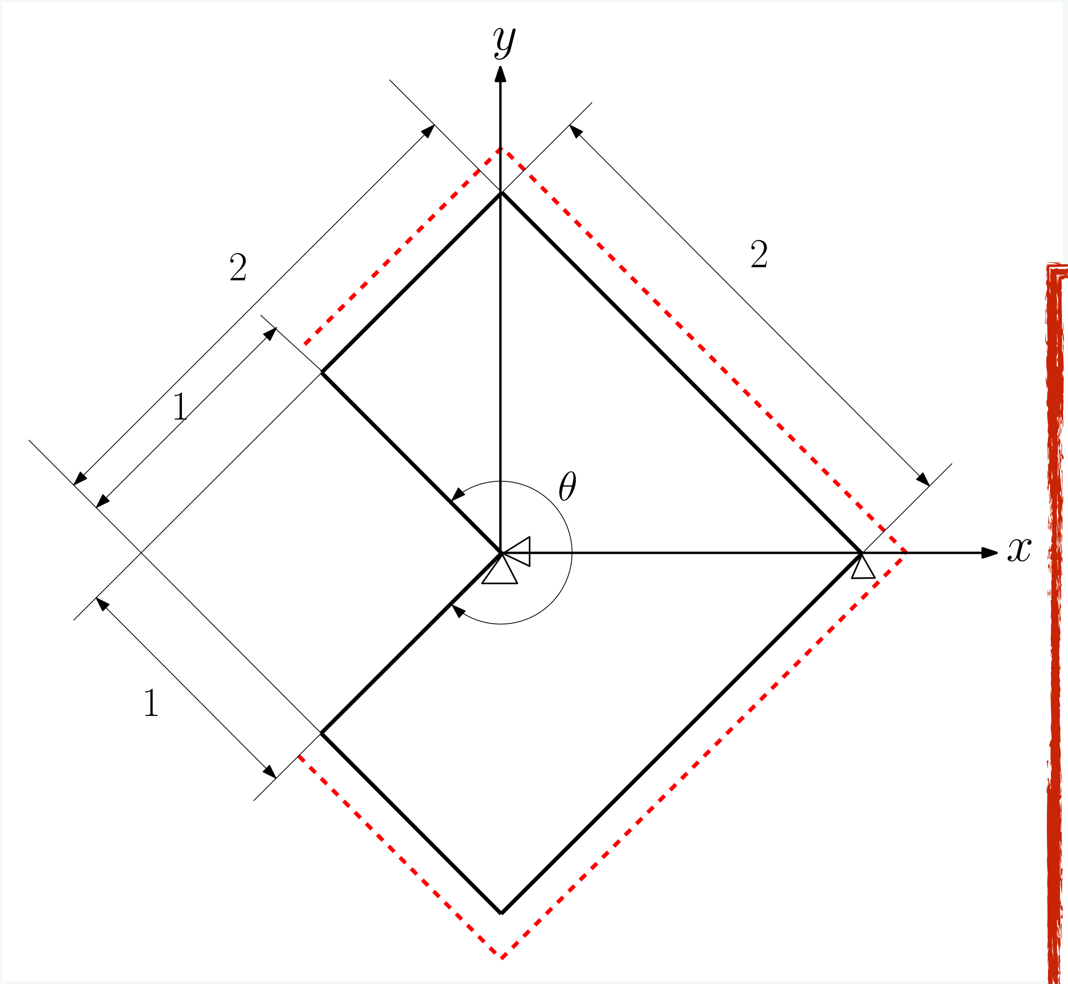


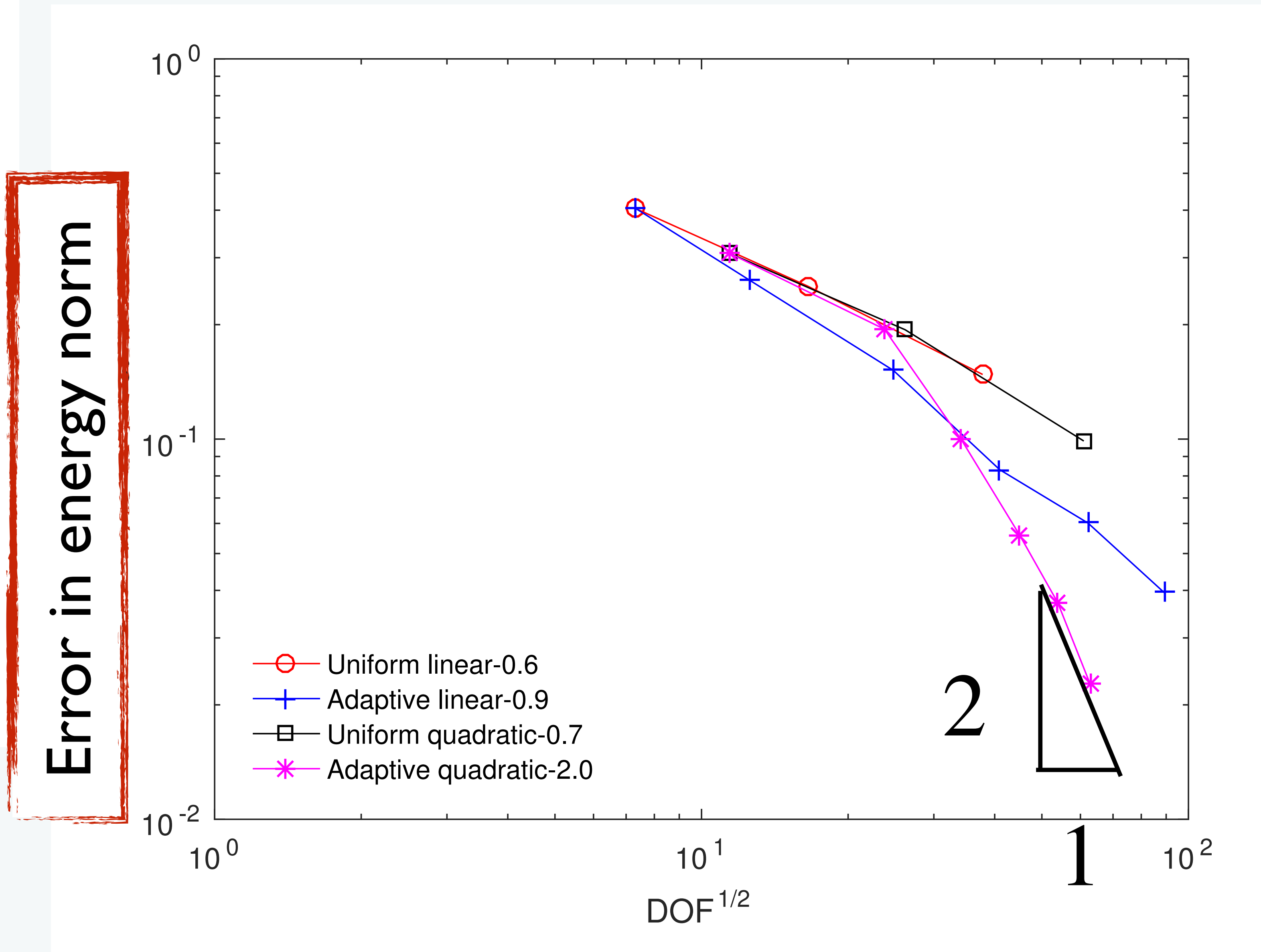


Adapted mesh is obtained **before any simulation is performed**

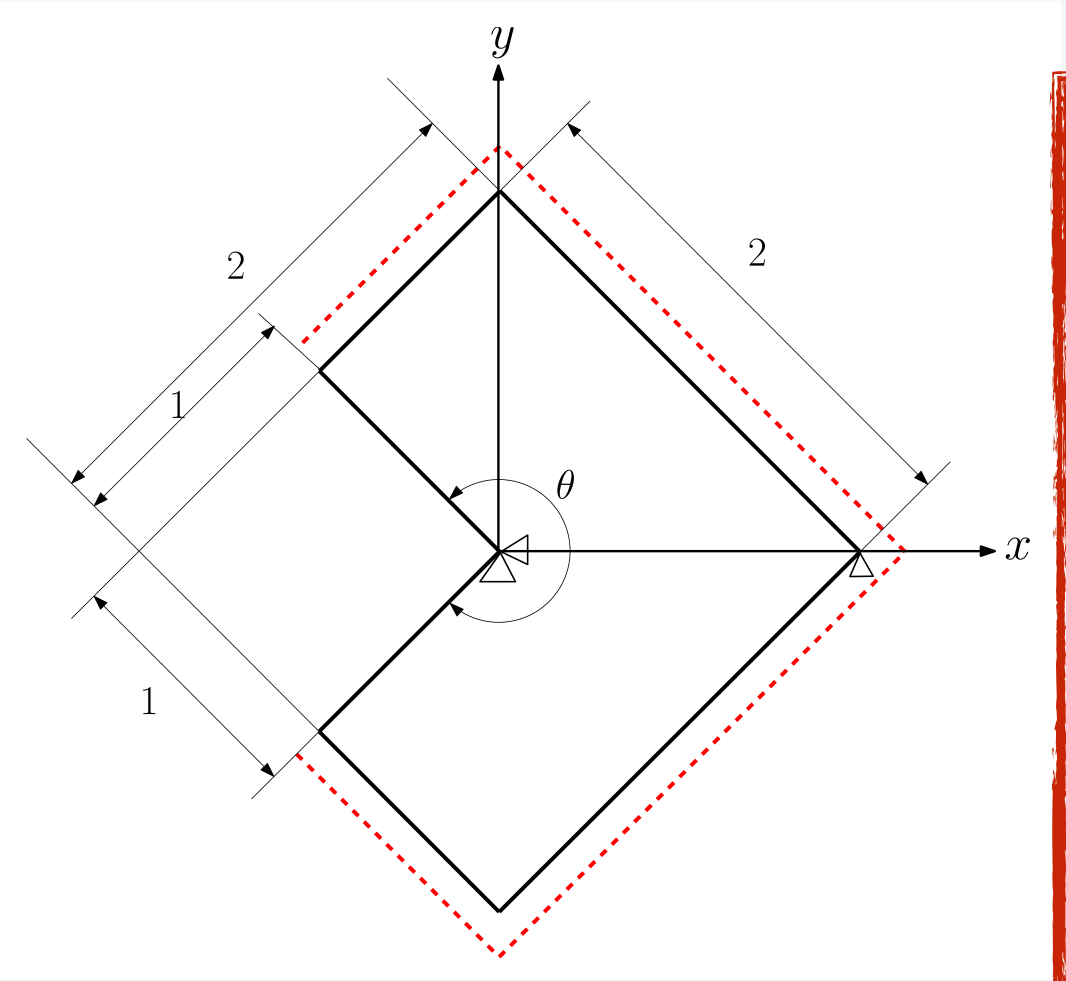
using the eigenvalues of the Hamiltonian

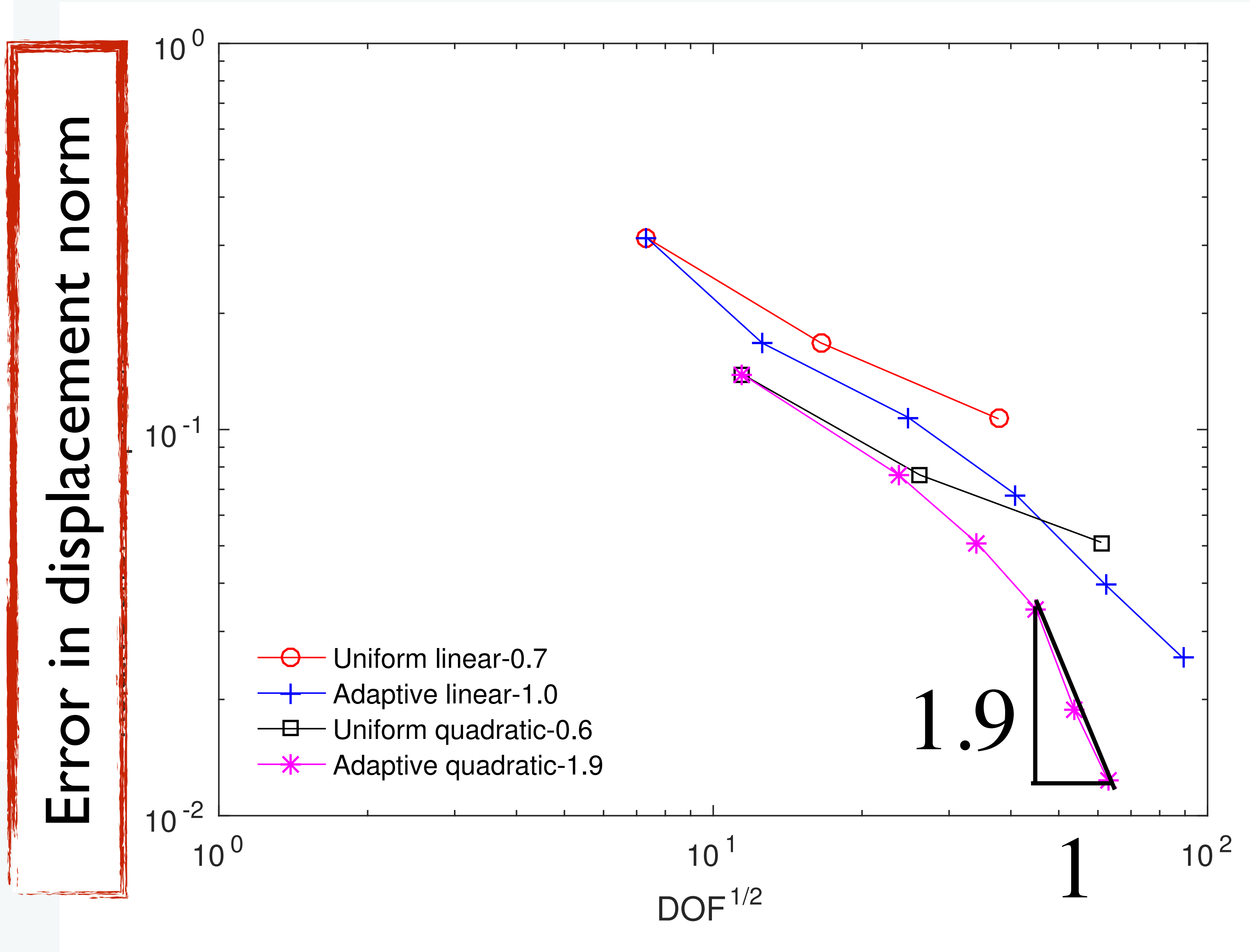
Novel error indicator





Novel error indicator





WCCM2020 Symposium in Paris

14th World Congress on Computational Mechanics (WCCM XIV)
8th European Congress on Computational Methods in Applied Science and Engineering (ECCOMAS 2020)
July 19- 24, 2020, Paris, France

BENCHMARKING ADVANCED DISCRETISATION TECHNIQUES: PART I. MESH BURDEN ALLEVIATION WITH APPLICATIONS TO CAD-ANALYSIS TRANSITION, FRACTURE MECHANICS AND HIGHER-ORDER PDES

TRACK NUMBER 20

Elena Atroshchenko, Stéphane Bordas, Franz Chouly, Daniel Dias-Da-costa, Jakub Lengiewicz, Sundararajan Natarajan, Timon Rabczuk, Chongmin Song, Satyendra Tomar, Giulio Ventura, Eric Wyart

Key words: verification and validation, benchmarking, mesh-burden, IGA, XFEM, embedded discontinuities,

ABSTRACT

The last 50 years have seen the birth of a large number of "special" approximation methods aiming at complementing finite difference and finite element methods and alleviating their intrinsic difficulties. Major advances have been made, and yet, it is not always obvious to identify the most relevant advantages and drawbacks of a given approach

Legato-team

University of Luxembourg

Department of Computational
Engineering Sciences
University of Luxembourg