On the analysis of OpenFOAM finite volume solution methodologies for linear elasticity

Philip Cardiff

April 2019

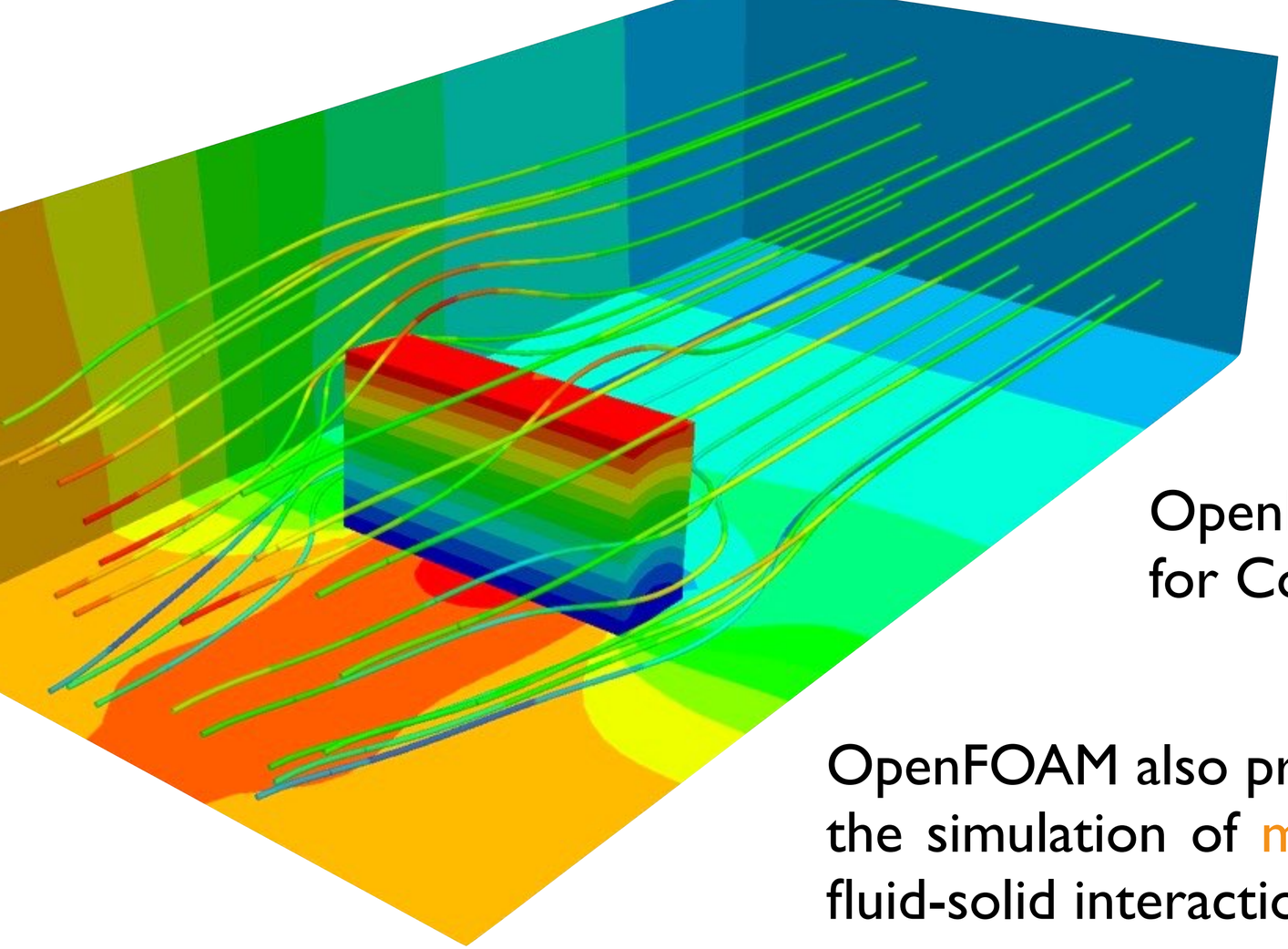
Online International Meeting for Users of OpenFOAM
Published on YouTube

On the analysis of OpenFOAM finite volume solution methodologies for linear elasticity

Philip Cardiff

School of Mechanical and Materials Engineering
University College Dublin

Background & Motivation



Background

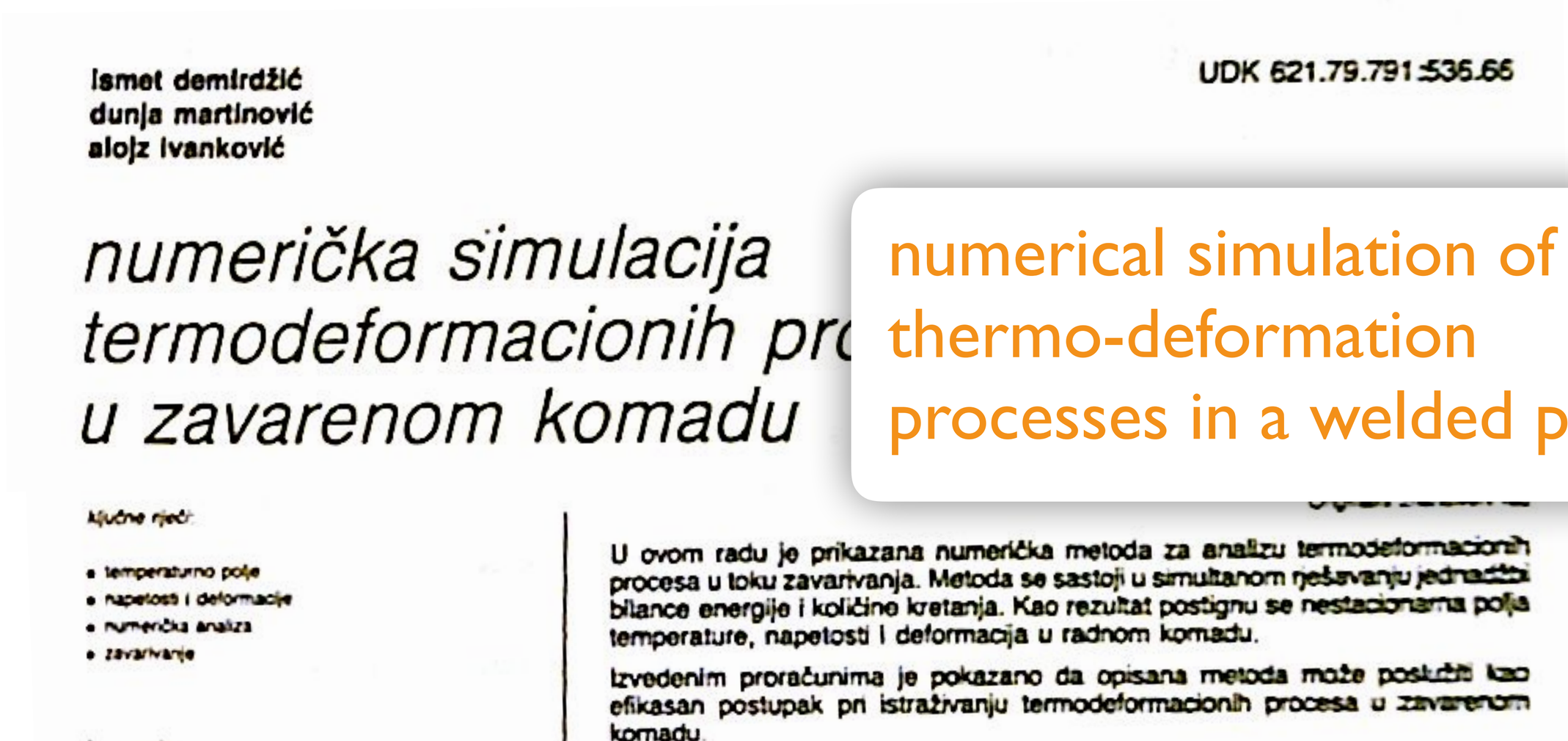
OpenFOAM is not just an excellent tool for Computational Fluid Dynamics

OpenFOAM also provides a suitable framework for the simulation of multi-physics problems, including fluid-solid interaction

The finite volume method offers an interesting alternative to the finite element method for solid mechanics

Historical Perspective

The first reported application of the finite volume method to solid mechanics was Demirdžić et al. in 1988:



Ismet demirdžić
dunja martinović
alojz ivanković

UDK 621.79.791:536.66

numerička simulacija
termodeformacionih pro
u zavarenom komadu

numerical simulation of thermo-deformation processes in a welded piece

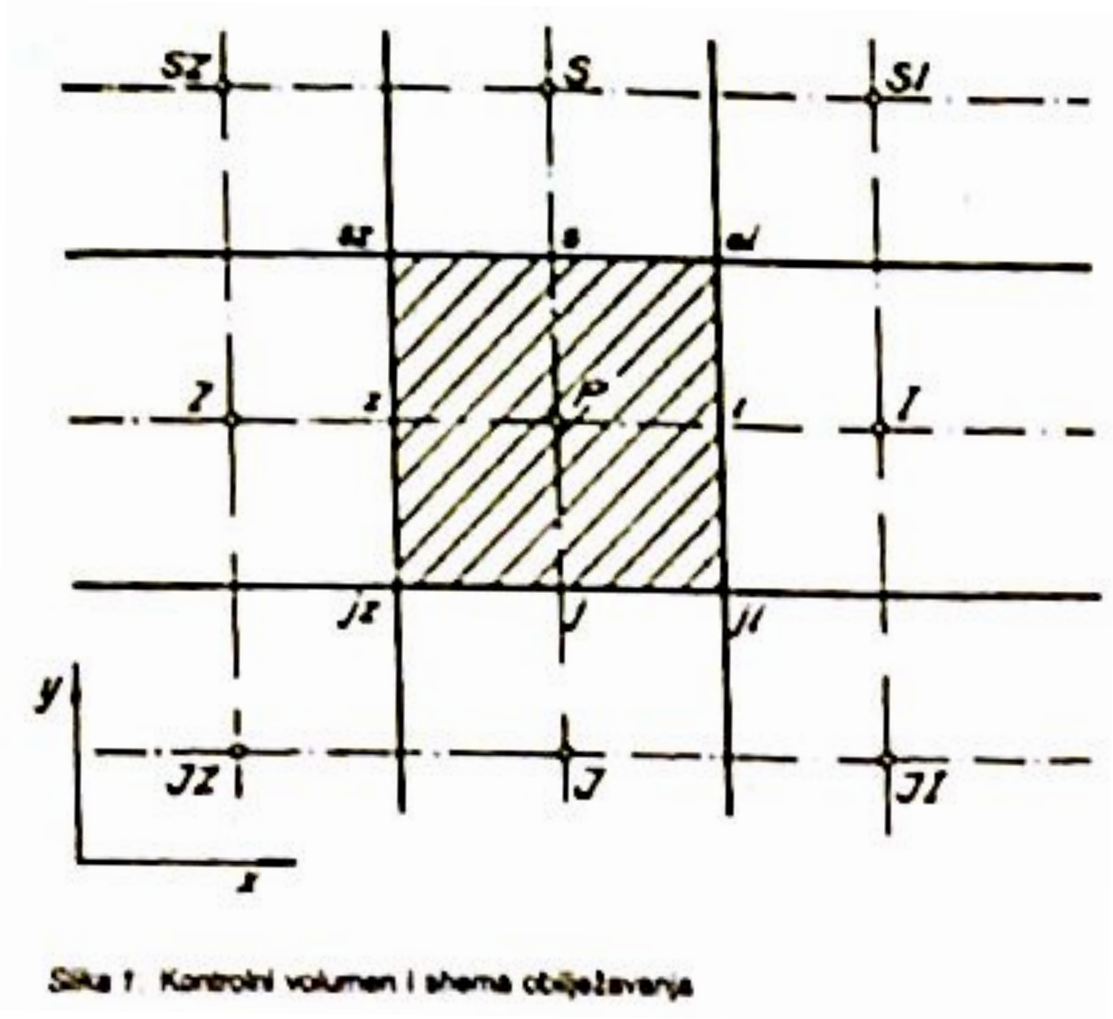
Ključne riječi:

- temperaturno polje
- napetosti i deformacije
- numerička analiza
- zavarivanje

U ovom radu je prikazana numerička metoda za analizu termodeformacionih procesa u toku zavarivanja. Metoda se sastoji u simultanom rješavanju jednadžbi bilance energije i količine kretanja. Kao rezultat postignu se nestacionarna polja temperature, napetosti i deformacija u radnom komadu.

Izvedenim proračunima je pokazano da opisana metoda može poslužiti kao efikasan postupak pri istraživanju termodeformacionih procesa u zavarenom komadu.

The method was initially limited to 2-D structured quadrilaterals and small-strain thermo-elasticity

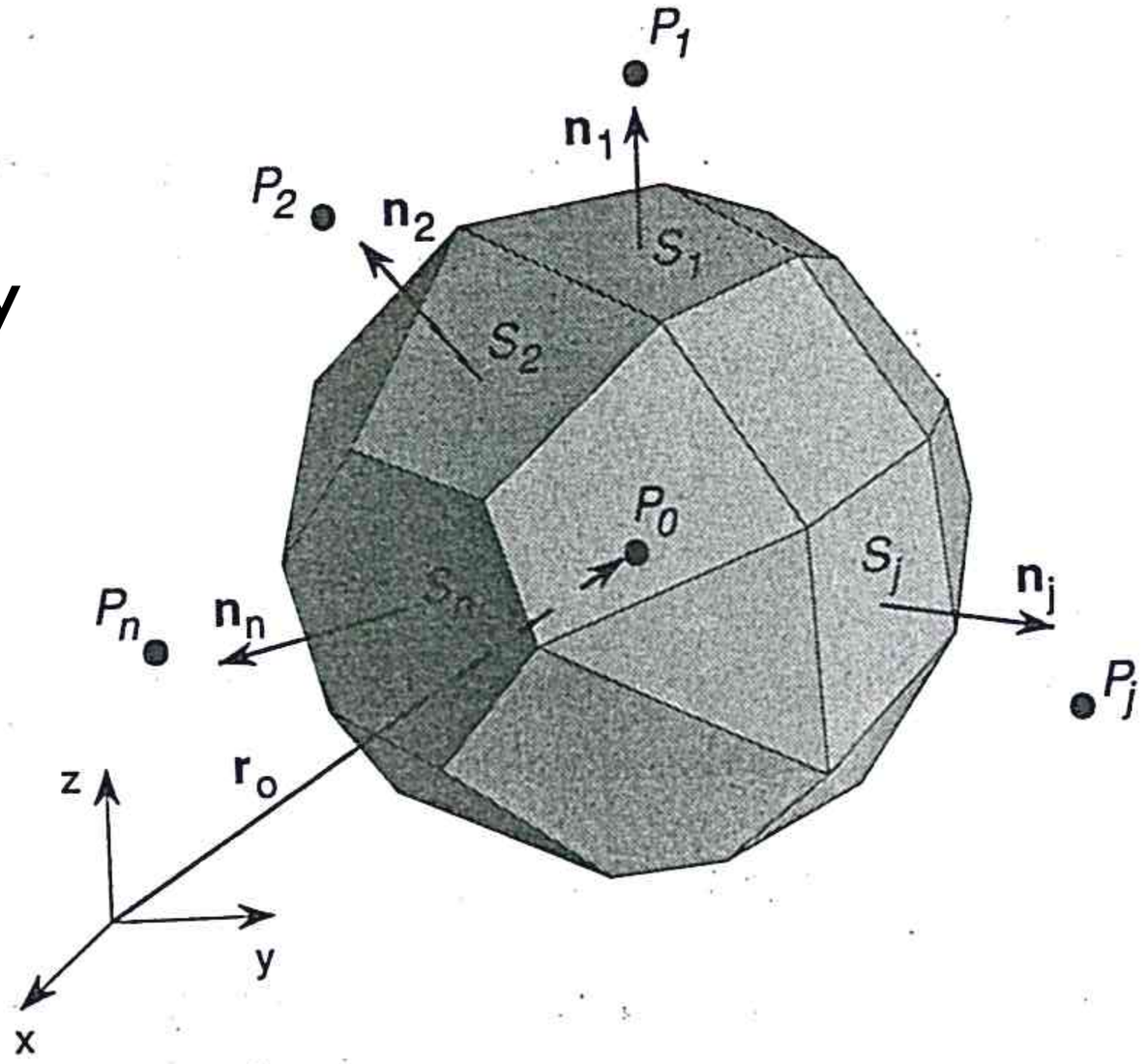


Slika 1 Kontrolni volumen i shema obilježavanja

I. Demirdžić, D. Martinović, and A. Ivanković. Numerical simulation of thermal deformation in welded workpiece. *Zavarivanje*, 31:209–219, 1988. in Croatian.

The method was initially limited to 2-D structured quadrilaterals and small-strain thermo-elasticity

and later extended to 3-D unstructured polyhedra, and a large range of physical phenomena



I. Demirdžić and S. Muzaferija. Finite volume method for stress analysis in complex domains. Int. J. Numer. Meth. Engng., 37:3751–3766, 1994.

In the seminal OpenFOAM paper, Weller et al. (1998) developed the first OpenFOAM stress analysis solver, `solidDisplacementFoam`, based on the method of Demirdžić and co-workers

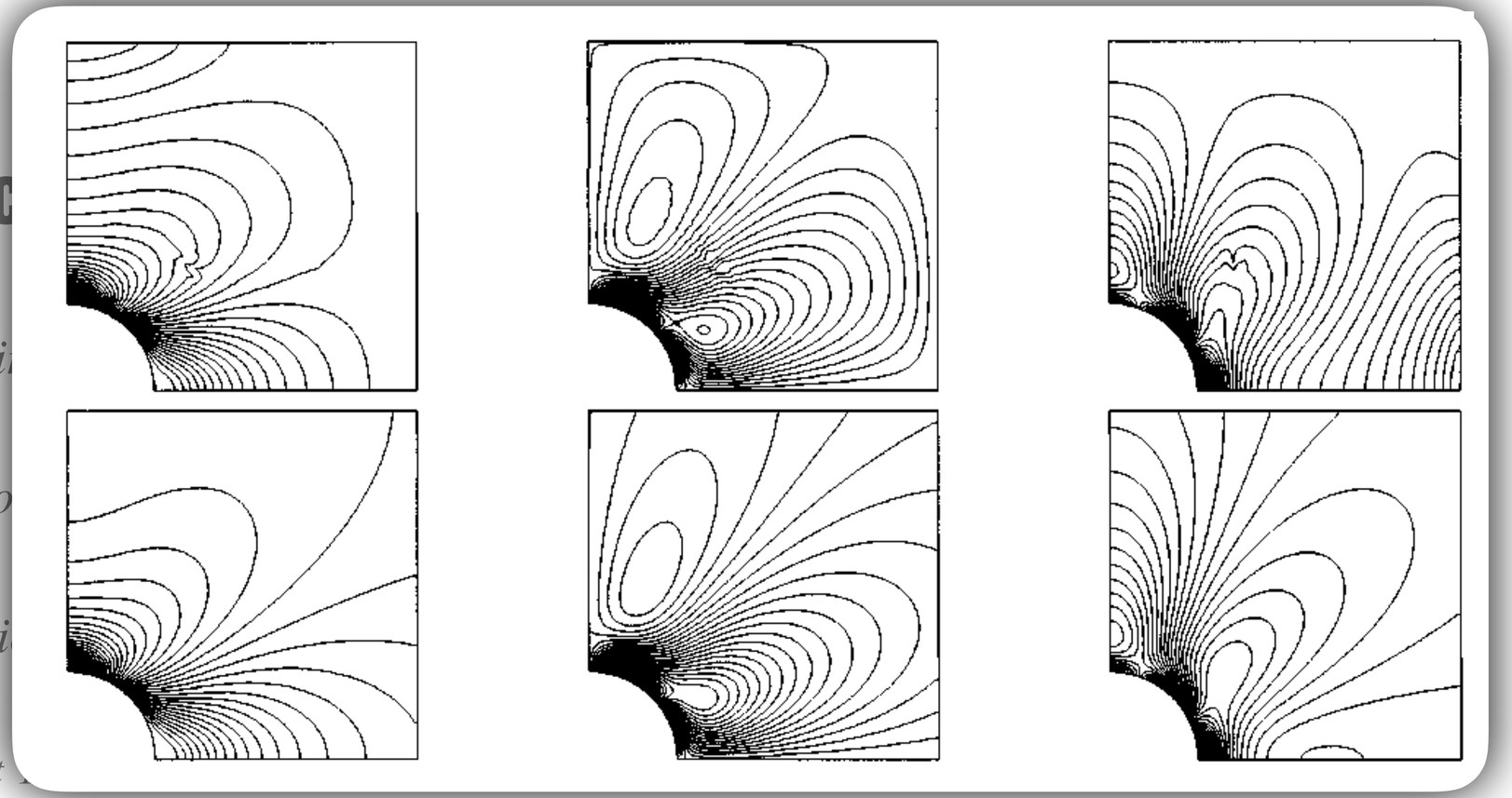
A tensorial approach to
using object-oriented tec

H. G. Weller and G. Tabor[a)]
Department of Mechanical Engineeri

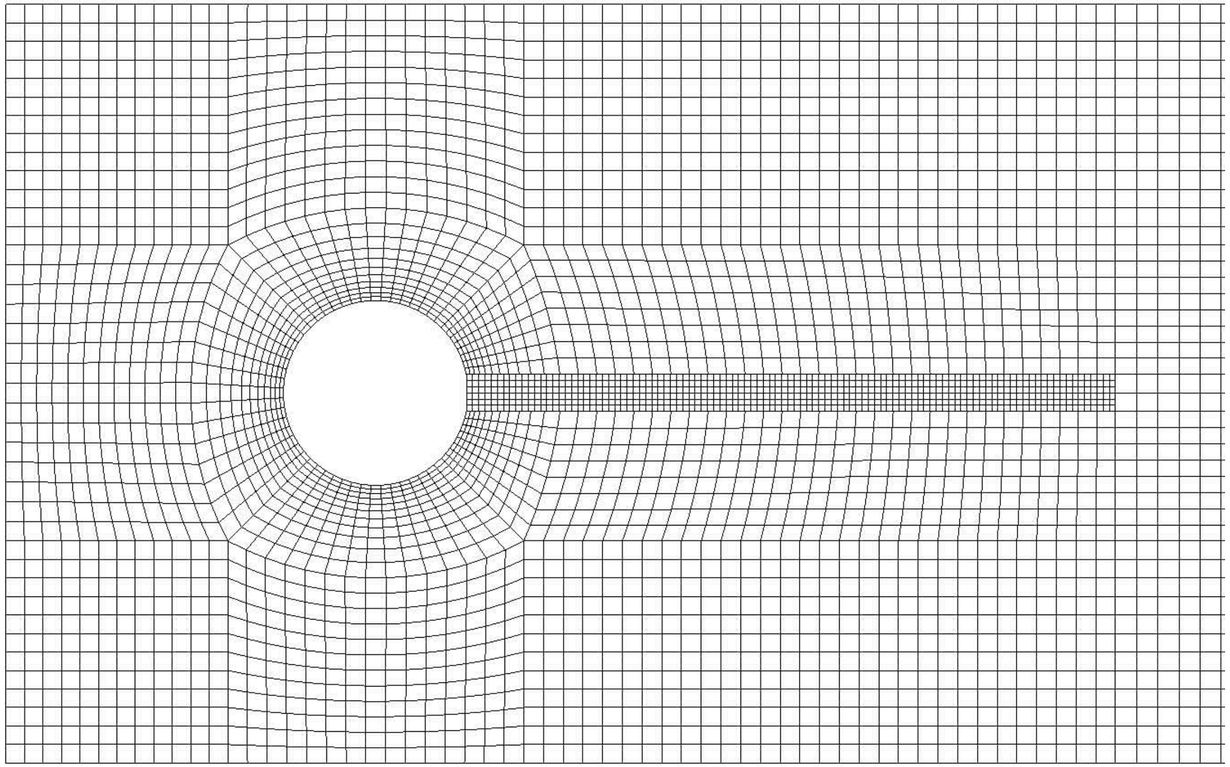
H. Jasak
Computational Dynamics Limited, Lo

C. Fureby
Department of Weapons and Protecti
S-17290 Stockholm, Sweden

(Received 1 June 1998; accepted 13 August



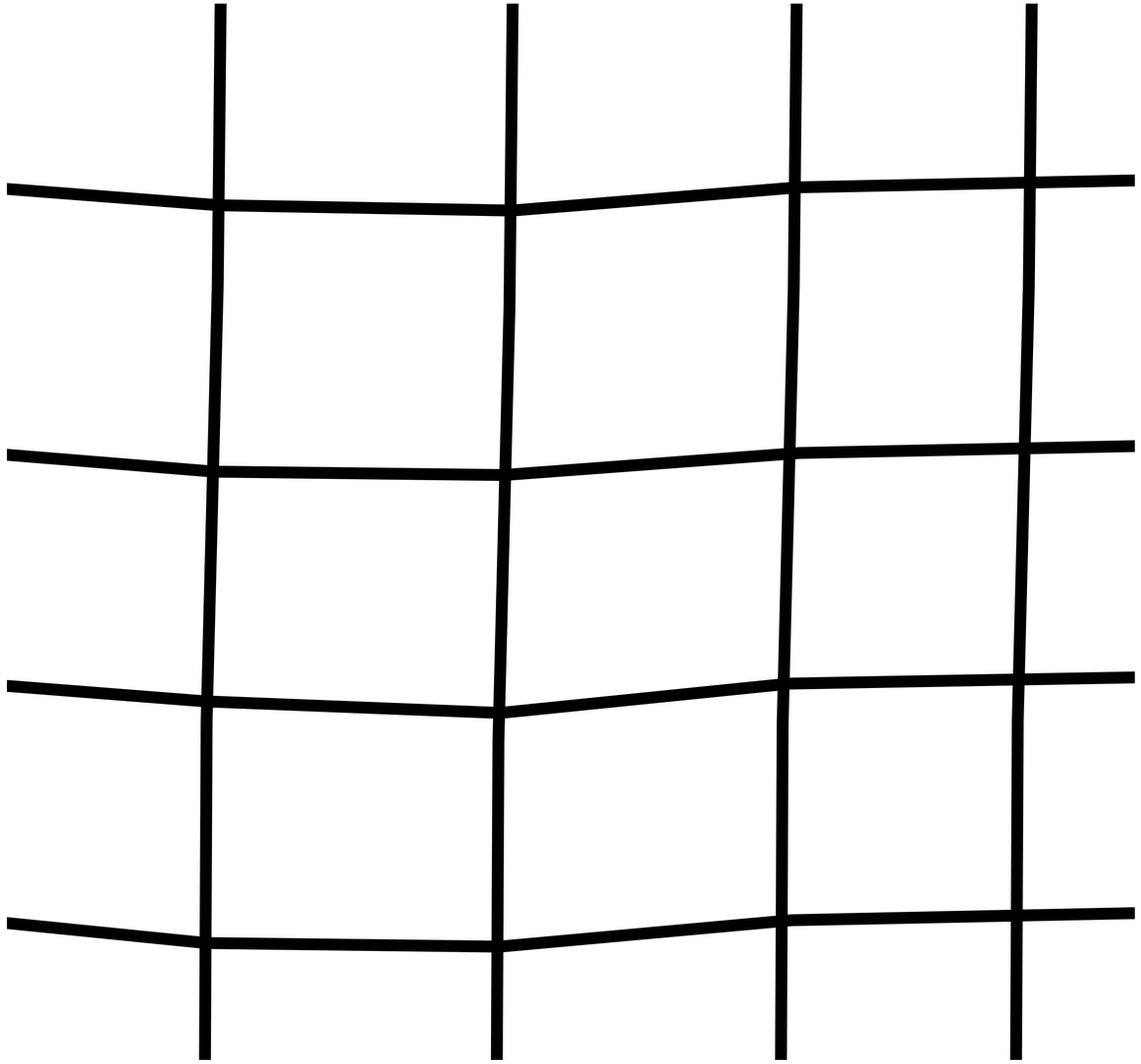
In the seminal OpenFOAM paper, Weller et al. (1998) developed the first OpenFOAM stress analysis solver, `solidDisplacementFoam`, based on the method of Demirdžić and co-workers



Subsequently, developments of solid mechanics in OpenFOAM have included more complex phenomena e.g. fluid-solid interaction, elastoplasticity, viscoelasticity, multi-materials, fracture, …

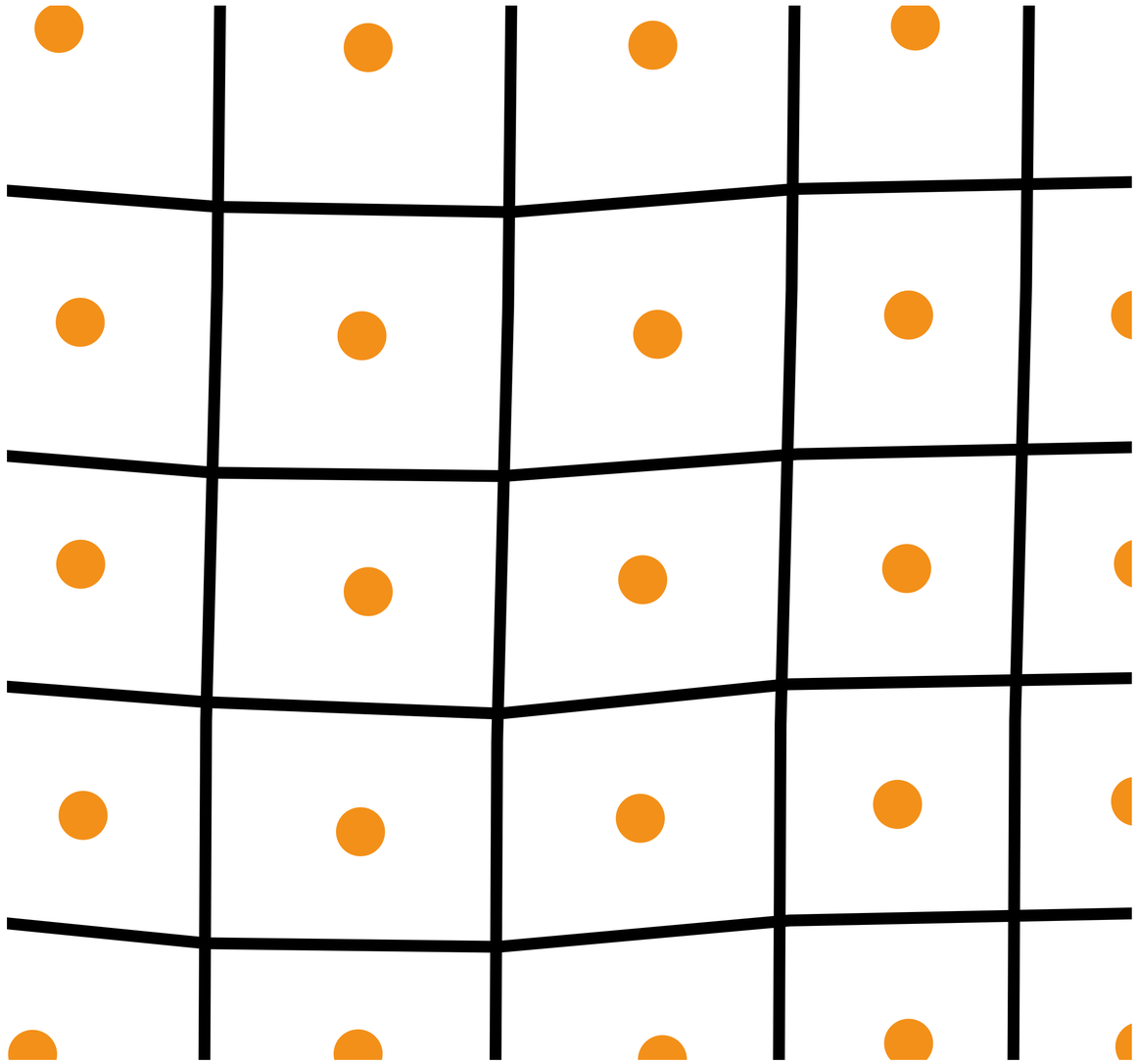
Finite volume approaches for solid mechanics can be classified in a number of ways:

- by **grid arrangement**



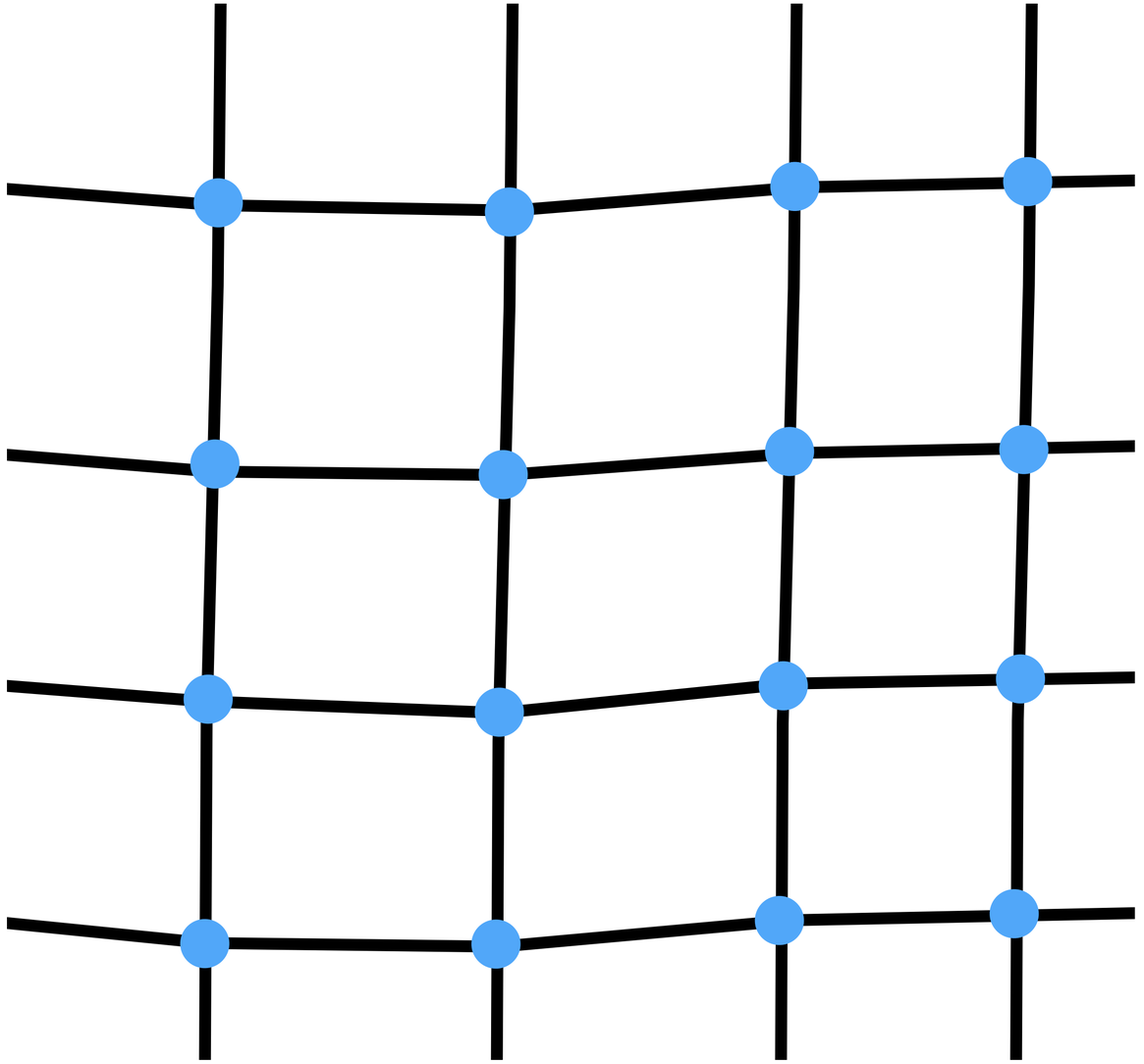
Finite volume approaches for solid mechanics can be classified in a number of ways:

- by **grid arrangement**
  - cell-centred
  - vertex-centred
  - staggered grid
  - other e.g. face-centred



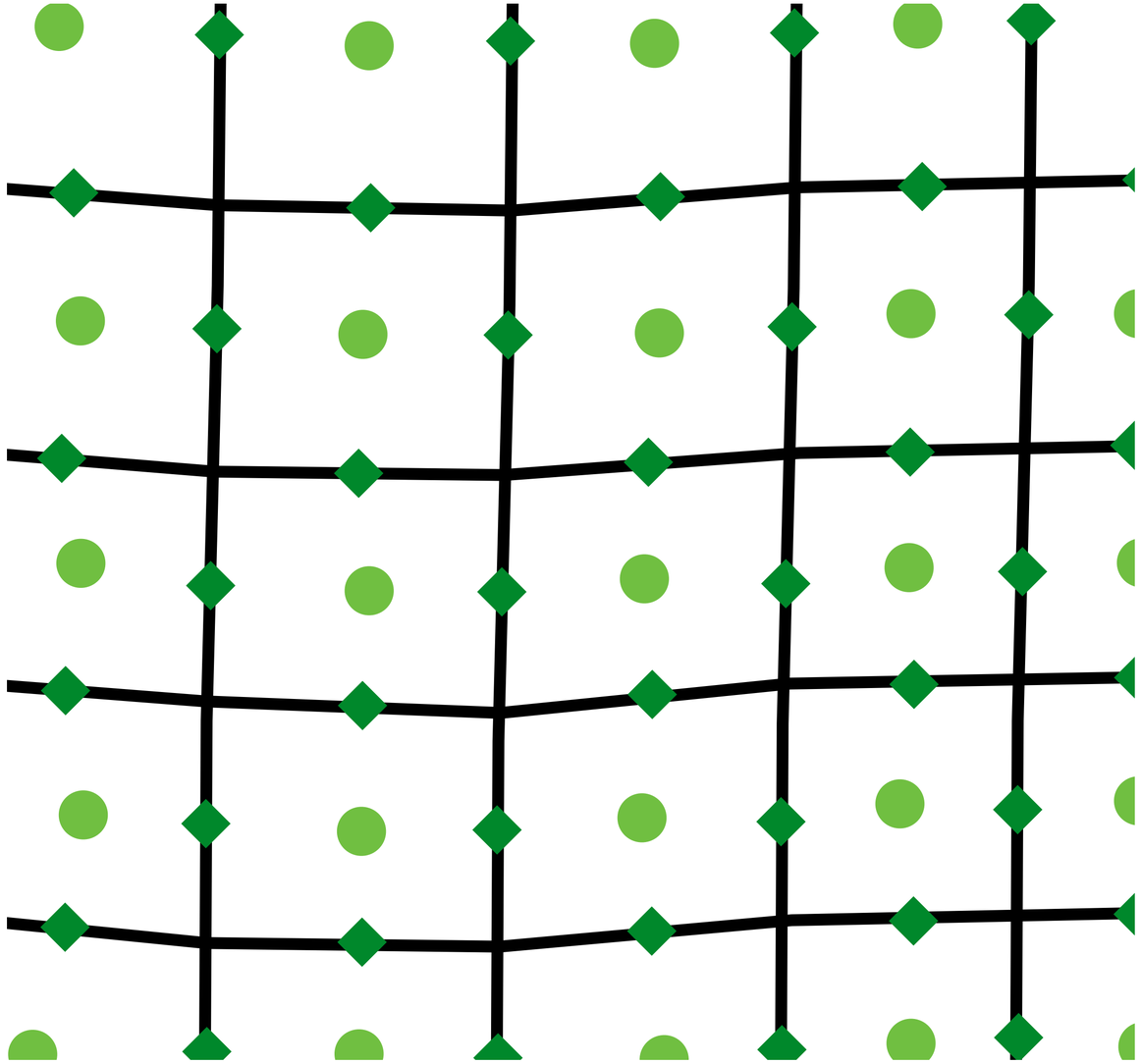
Finite volume approaches for solid mechanics can be classified in a number of ways:

- by **grid arrangement**
  - cell-centred
  - vertex-centred
  - staggered grid
  - other e.g. face-centred



Finite volume approaches for solid mechanics can be classified in a number of ways:

- by **grid arrangement**
  - cell-centred
  - vertex-centred
  - staggered grid
  - other e.g. face-centred



Finite volume approaches for solid mechanics can be classified in a number of ways:

- by grid arrangement
  - cell-centred
  - vertex-centred
  - staggered grid
  - other e.g. face-centred
- by **solution methodology**
  - implicit vs explicit

$$\rho \frac{\partial \boldsymbol{v}}{\partial t} = \boldsymbol{\nabla} \cdot \boldsymbol{\sigma}$$

$$\rho \frac{\boldsymbol{v}_m - \boldsymbol{v}_{m-1}}{\Delta t} = \boldsymbol{\nabla} \cdot \boldsymbol{\sigma}_{m-1}$$

$$\rho \frac{\boldsymbol{v}_m - \boldsymbol{v}_{m-1}}{\Delta t} = \boldsymbol{\nabla} \cdot \boldsymbol{\sigma}_m$$

Finite volume approaches for solid mechanics can be classified in a number of ways:

- by grid arrangement
  - cell-centred
  - vertex-centred
  - staggered grid
  - other e.g. face-centred
- by solution methodology
  - implicit vs explicit
- by **stabilisation method**
  - Rhie-Chow vs JST vs Godunov

Stabilisation of pressure oscillation:

large stencil

$$(\boldsymbol{\nabla} p)_f \approx \omega_f (\boldsymbol{\nabla} p)_P + (1 - \omega_f)(\boldsymbol{\nabla} p)_N$$

small stencil

$$(\boldsymbol{\nabla} p)_f \approx \frac{p_N - p_P}{\boldsymbol{n} \cdot \boldsymbol{d}}$$

Diffusion term that damps pressure oscillations

$$\mathbb{D}_{\mathrm{RC}} = \frac{p_N - p_P}{\boldsymbol{n} \cdot \boldsymbol{d}} - [\omega_f (\boldsymbol{\nabla} p)_P + (1 - \omega_f)(\boldsymbol{\nabla} p)_N]$$

Other diffusion terms

$$\mathbb{D}_{\mathrm{JST}} = \gamma \boldsymbol{\nabla}^2 (\boldsymbol{\nabla}^2 p)$$

Solution Methodologies

Governing equation

$$\frac{\mathrm{D}}{\mathrm{D}t}\int_{\Omega}\rho\boldsymbol{v}\,\mathrm{d}\Omega \;=\; \oint_{\Gamma}\boldsymbol{n}\cdot\boldsymbol{\sigma}\,\mathrm{d}\Gamma \;+\; \int_{\Omega}\rho\,\boldsymbol{b}\,\mathrm{d}\Omega$$

Governing equation

$$\frac{\mathrm{D}}{\mathrm{D}t}\int_\Omega \rho\boldsymbol{v}\,\mathrm{d}\Omega = \oint_\Gamma \boldsymbol{n}\cdot\boldsymbol{\sigma}\,\mathrm{d}\Gamma + \int_\Omega \rho\,\boldsymbol{b}\,\mathrm{d}\Omega$$

Reynold's transport theorem

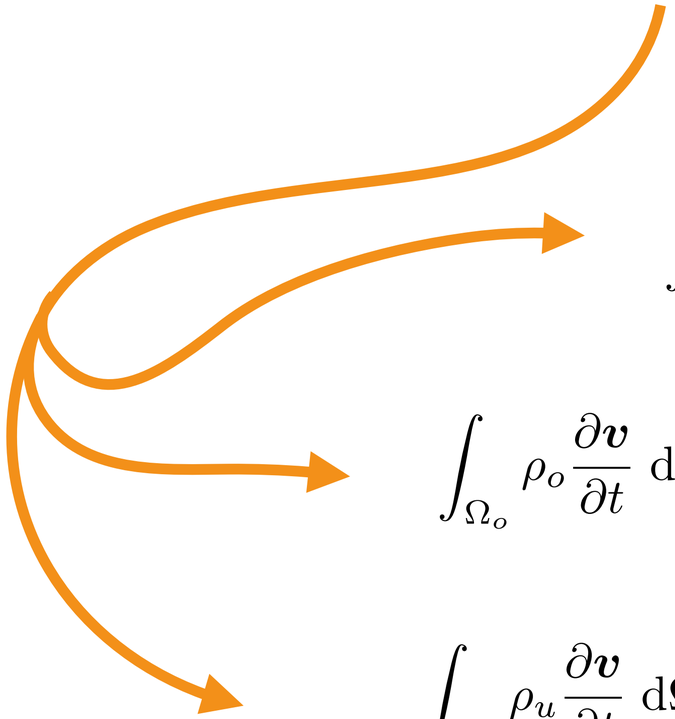
$$\int_\Omega \frac{\partial}{\partial t}(\rho\boldsymbol{v})\,\mathrm{d}\Omega + \oint_\Gamma \rho\boldsymbol{v}\,[\boldsymbol{n}\cdot(\boldsymbol{v}-\boldsymbol{v}_\Gamma)]\,\mathrm{d}\Gamma = \oint_\Gamma \boldsymbol{n}\cdot\boldsymbol{\sigma}\,\mathrm{d}\Gamma + \int_\Omega \rho\,\boldsymbol{b}\,\mathrm{d}\Omega$$

Lagrangian

$\boldsymbol{v}-\boldsymbol{v}_\Gamma = 0$

$\boldsymbol{\sigma} = f(\boldsymbol{v}, \boldsymbol{u}, \Delta\boldsymbol{u}, ...)$

$$\int_\Omega \frac{\partial \rho \boldsymbol{v}}{\partial t}\,\mathrm{d}\Omega = \oint_\Gamma \boldsymbol{n}\cdot\boldsymbol{\sigma}\,\mathrm{d}\Gamma + \int_\Omega \rho\boldsymbol{b}\,\mathrm{d}\Omega$$



linear geometry

$$\int_{\Omega_o} \rho_o \frac{\partial \boldsymbol{v}}{\partial t}\,\mathrm{d}\Omega_o = \oint_{\Gamma_o} \boldsymbol{n}_o\cdot\boldsymbol{\sigma}\,\mathrm{d}\Gamma_o + \int_{\Omega_o} \rho_o\boldsymbol{b}\,\mathrm{d}\Omega_o$$

nonlinear geometry total Lagrangian

$$\int_{\Omega_o} \rho_o \frac{\partial \boldsymbol{v}}{\partial t}\,\mathrm{d}\Omega_o = \oint_{\Gamma_o} \left(J\boldsymbol{F}^{-T}\cdot\boldsymbol{n}_o\right)\cdot\boldsymbol{\sigma}\,\mathrm{d}\Gamma_o + \int_{\Omega_o} \rho_o\boldsymbol{b}\,\mathrm{d}\Omega_o$$

$$\boldsymbol{F} = \mathbf{I} + (\nabla \boldsymbol{u})^T$$
$$J = \det[\boldsymbol{F}]$$

nonlinear geometry updated Lagrangian

$$\int_{\Omega_u} \rho_u \frac{\partial \boldsymbol{v}}{\partial t}\,\mathrm{d}\Omega_u = \oint_{\Gamma_u} \left(j\boldsymbol{f}^{-T}\cdot\boldsymbol{n}_u\right)\cdot\boldsymbol{\sigma}\,\mathrm{d}\Gamma_u + \int_{\Omega_u} \rho_u\boldsymbol{b}\,\mathrm{d}\Omega_u$$

$$\boldsymbol{f} = \mathbf{I} + (\nabla[\Delta \boldsymbol{u}])^T$$
$$j = \det[\boldsymbol{f}]$$

$$\int_{\Omega_o} \rho_o \frac{\partial \boldsymbol{v}}{\partial t} \, \mathrm{d}\Omega_o = \oint_{\Gamma_o} \boldsymbol{n}_o \cdot \boldsymbol{\sigma} \, \mathrm{d}\Gamma_o + \int_{\Omega_o} \rho_o \boldsymbol{b} \, \mathrm{d}\Omega_o$$

$$\int_{\Omega_o} \rho_o \frac{\partial^2 \boldsymbol{u}}{\partial t^2} \, \mathrm{d}\Omega_o = \oint_{\Gamma_o} \boldsymbol{n}_o \cdot \boldsymbol{\sigma} \, \mathrm{d}\Gamma_o + \int_{\Omega_o} \rho_o \boldsymbol{b} \, \mathrm{d}\Omega_o$$

$$\int_{\Omega_o} \rho_o \frac{\partial^2 \boldsymbol{u}}{\partial t^2} \, \mathrm{d}\Omega_o = \oint_{\Gamma_o} \boldsymbol{n}_o \cdot \left[\mu \boldsymbol{\nabla} \boldsymbol{u} + \mu (\boldsymbol{\nabla} \boldsymbol{u})^T + \lambda \mathrm{tr}(\boldsymbol{\nabla} \boldsymbol{u}) \mathbf{I} \right] \, \mathrm{d}\Gamma_o + \int_{\Omega_o} \rho_o \boldsymbol{b} \, \mathrm{d}\Omega_o$$

$$\int_{\Omega_o} \rho_o \frac{\partial^2 \boldsymbol{u}}{\partial t^2} \, \mathrm{d}\Omega_o = \oint_{\Gamma_o} \boldsymbol{n}_o \cdot \left[\mu \boldsymbol{\nabla} \boldsymbol{u} + \mu (\boldsymbol{\nabla} \boldsymbol{u})^T + \lambda \mathrm{tr}(\boldsymbol{\nabla} \boldsymbol{u}) \mathbf{I} \right] \, \mathrm{d}\Gamma_o + \int_{\Omega_o} \rho_o \boldsymbol{b} \, \mathrm{d}\Omega_o$$

We will consider a selection of differing solution methodologies:

- **implicit - segregated**
  - *primitive variable*: total displacement
  - *face gradient discretisation*: interpolate gradients from cell-centres
  - *stabilisation*: Rhie-Chow or JST stabilisation
- **implicit - block-coupled**
  - *primitive variable*: total displacement
  - *face gradient discretisation*: calculate normal/tangential gradient at face
  - *stabilisation*: none
- **explicit**
  - *primitive variable*: total displacement
  - *face gradient discretisation*: interpolate gradients from cell-centres
  - *stabilisation*: JST stabilisation

- **implicit - segregated**
  - *primitive variable*: total displacement
  - *face gradient discretisation*: interpolate gradients from cell-centres
  - *stabilisation*: Rhie-Chow or JST stabilisation

```
do
{
    solve
    (
        rho*fvm::d2dt2(U)
     == fvm::laplacian(mu, U)
      + fvc::div(mu*gradU.T() + lambda*tr(gradU)*I)
      + rho*gravity
      + stabilisation
    );

    gradU = fvc::grad(U);
    stabilisation = alpha*(fvc::laplacian(mu, U) - fvc::div(mu*gradU)); // Rhie-Chow
    //stabilisation = -alpha*fvc::laplacian(mesh.magSf(), fvc::laplacian(mu, U)); // JST
}
while (!converged);
```

Typical values for the stabilisation scale factor (`alpha`) are 1.0 for Rhie-Chow and 0.001 for JST.

- **implicit - segregated**
  - *primitive variable*: total displacement
  - *face gradient discretisation*: interpolate gradients from cell-centres
  - *stabilisation*: Rhie-Chow or JST stabilisation

```
do
{
    solve                                                    Jasak & Weller "over-relaxed" approach
    (
        rho*fvm::d2dt2(U)
     == fvm::laplacian(2*mu + lambda, U)
      - fvc::laplacian(mu + lambda, U)
      + fvc::div(mu*gradU.T() + lambda*tr(gradU)*I)
      + rho*gravity
      + stabilisation
    );

    gradU = fvc::grad(U);
    stabilisation = alpha*(fvc::laplacian(mu, U) - fvc::div(mu*gradU)); // Rhie-Chow
    //stabilisation = -alpha*fvc::laplacian(mesh.magSf(), fvc::laplacian(mu, U)); // JST
}
while (!converged);
```

H. Jasak and H. G. Weller, Application of the finite volume method and unstructured meshes to linear elasticity, Int. J. Numer. Meth. Engng. 2000; 48:267–287

$$\int_{\Omega_o} \rho_o \frac{\partial^2 \boldsymbol{u}}{\partial t^2} \, \mathrm{d}\Omega_o = \oint_{\Gamma_o} \boldsymbol{n}_o \cdot \left[\mu \boldsymbol{\nabla} \boldsymbol{u} + \mu (\boldsymbol{\nabla} \boldsymbol{u})^T + \lambda \mathrm{tr}(\boldsymbol{\nabla} \boldsymbol{u}) \mathbf{I} \right] \, \mathrm{d}\Gamma_o + \int_{\Omega_o} \rho_o \boldsymbol{b} \, \mathrm{d}\Omega_o$$

We will consider a selection of differing solution methodologies:

- **implicit - segregated**
  - *primitive variable*: total displacement
  - *face gradient discretisation*: interpolate gradients from cell-centres
  - *stabilisation*: Rhie-Chow or JST stabilisation
- **implicit - block-coupled**
  - *primitive variable*: total displacement
  - *face gradient discretisation*: calculate norma/tangential gradient at face
  - *stabilisation*: none
- **explicit**
  - *primitive variable*: total displacement
  - *face gradient discretisation*: interpolate gradients from cell-centres
  - *stabilisation*: JST stabilisation

- **implicit - block-coupled**
  - *primitive variable*: total displacement
  - *face gradient discretisation*: calculate norma/tangential gradient at face
  - *stabilisation*: none

```
solve
 (
     rho*fvm::d2dt2(U)
  == BlockFvm::laplacian(mu, U)
   + BlockFvm::laplacianTranspose(mu, U)
   + BlockFvm::laplacianTrace(lambda, U)
   + rho*gravity
 );

 ...   // actual code has a few more steps
```

P. Cardiff, Ž. Tuković, P. De Jaeger, M. Clancy, A. Ivanković: A Lagrangian cell-centred finite volume method for metal forming simulation. International Journal for Numerical Methods in Engineering, 2016, DOI: 10.1002/nme.5345.

$$\int_{\Omega_o} \rho_o \frac{\partial^2 \boldsymbol{u}}{\partial t^2} \, \mathrm{d}\Omega_o = \oint_{\Gamma_o} \boldsymbol{n}_o \cdot \left[\mu \boldsymbol{\nabla} \boldsymbol{u} + \mu (\boldsymbol{\nabla} \boldsymbol{u})^T + \lambda \mathrm{tr}(\boldsymbol{\nabla} \boldsymbol{u}) \mathbf{I} \right] \, \mathrm{d}\Gamma_o + \int_{\Omega_o} \rho_o \boldsymbol{b} \, \mathrm{d}\Omega_o$$

We will consider a selection of differing solution methodologies:

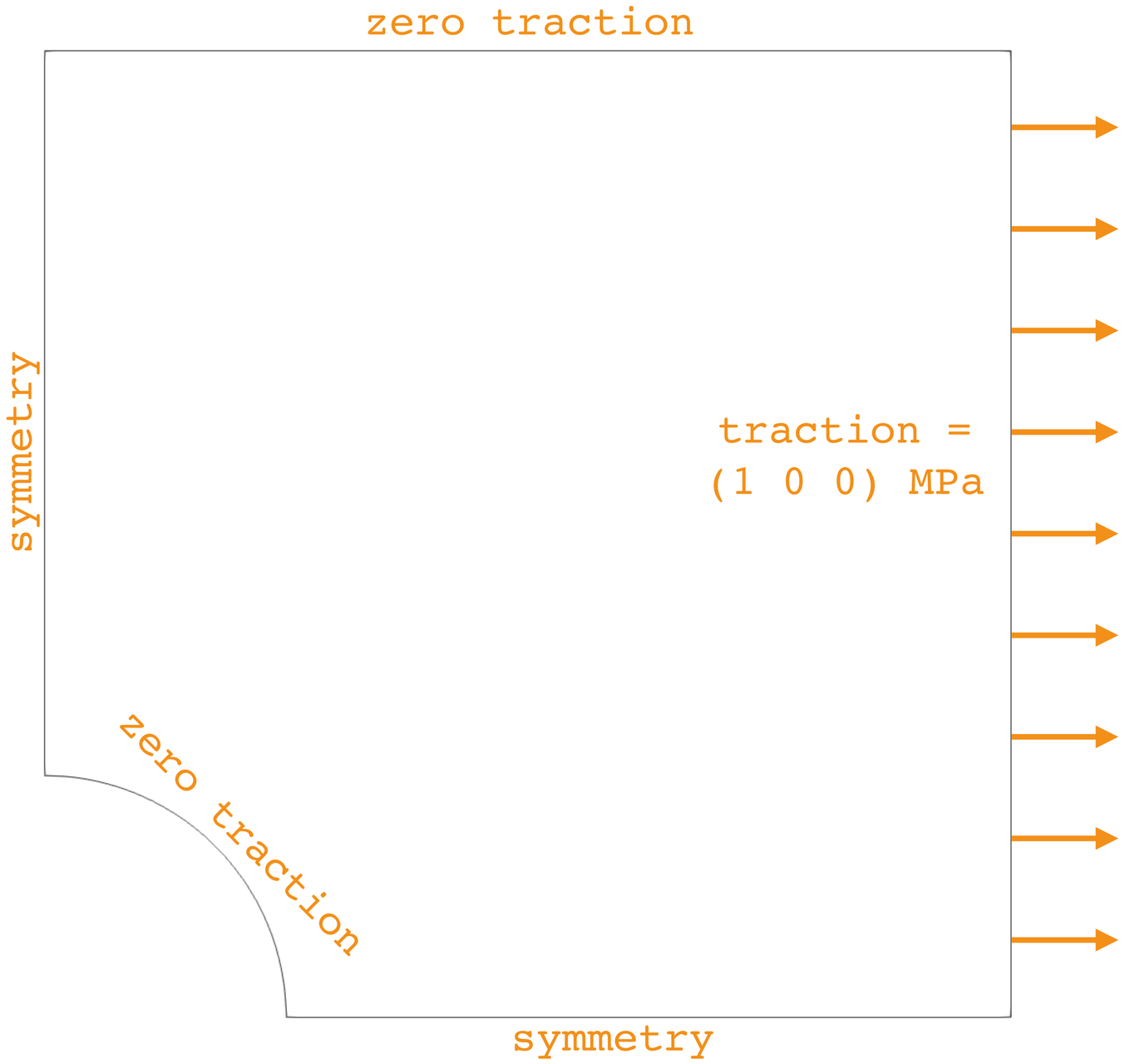
- **implicit - segregated**
  - *primitive variable*: total displacement
  - *face gradient discretisation*: interpolate gradients from cell-centres
  - *stabilisation*: Rhie-Chow or JST stabilisation
- **implicit - block-coupled**
  - *primitive variable*: total displacement
  - *face gradient discretisation*: calculate norma/tangential gradient at face
  - *stabilisation*: none
- **explicit**
  - *primitive variable*: total displacement
  - *face gradient discretisation*: interpolate gradients from cell-centres
  - *stabilisation*: JST stabilisation

- **explicit**
  - *primitive variable*: total displacement
  - *face gradient discretisation*: interpolate gradients from cell-centres
  - *stabilisation*: JST stabilisation

```
// Update velocity
V = V.oldTime() + runTime.deltaT()*a.oldTime();

// Update displacement
U = U.oldTime() + runTime.deltaT()*V;

// Enforce displacement boundary conditions
U.correctBoundaryConditions();

// Update stress
sigma = 2*mu*symm(fvc::grad(U)) + lambda*tr(fvc::grad(U))*I;

// Calculate JST stabilisation
stabilisation = -alpha*fvc::laplacian(mesh.magSf(), fvc::laplacian(mu, V))

// Update acceleration
a = (fvc::div(sigma) + stabilisation)/rho + gravity;
```

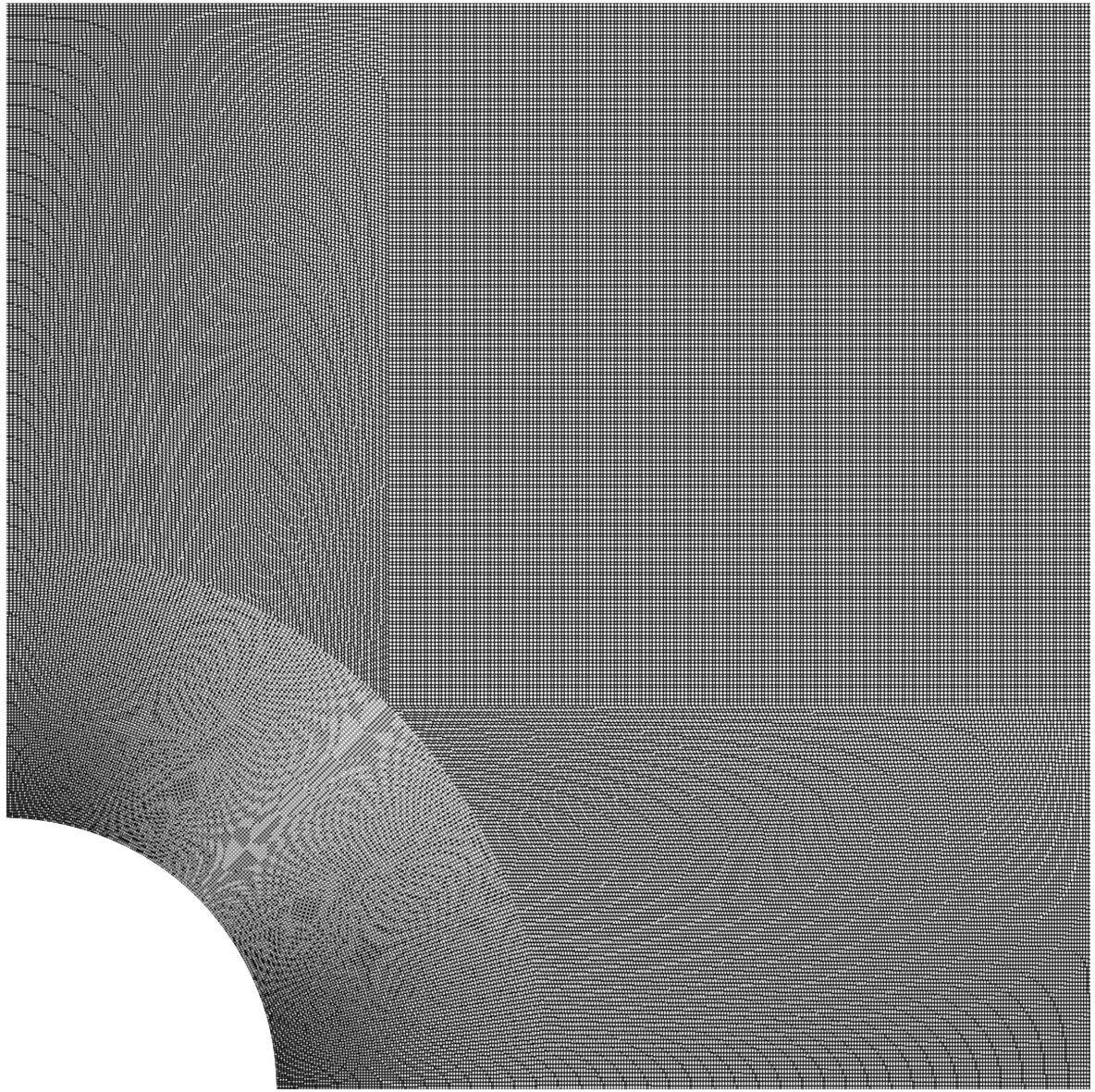
Example Cases

plateHole


zero traction
symmetry
traction =
(1 0 0) MPa
zero traction
symmetry

plateHole

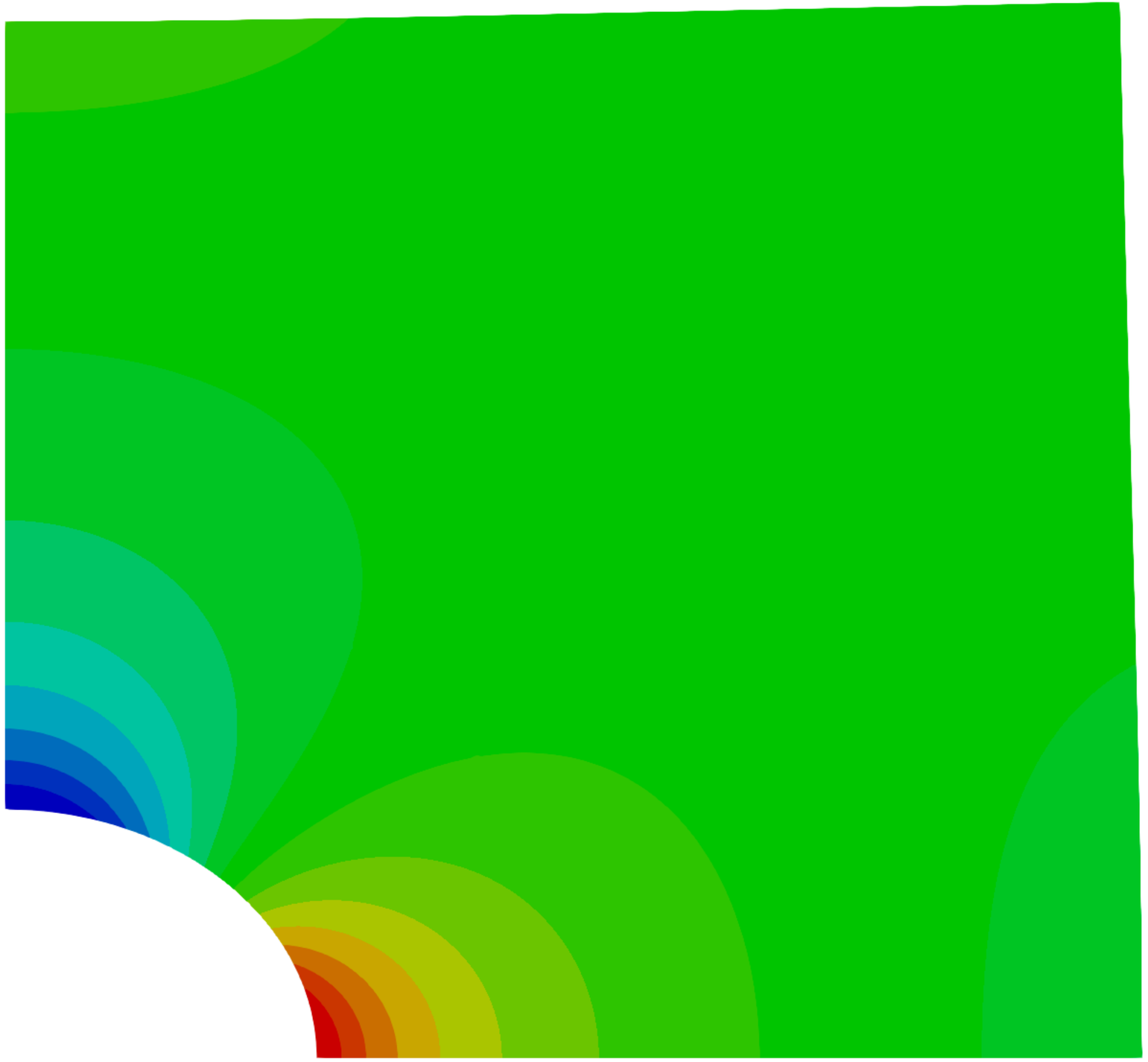
100,000 hexahedral cells



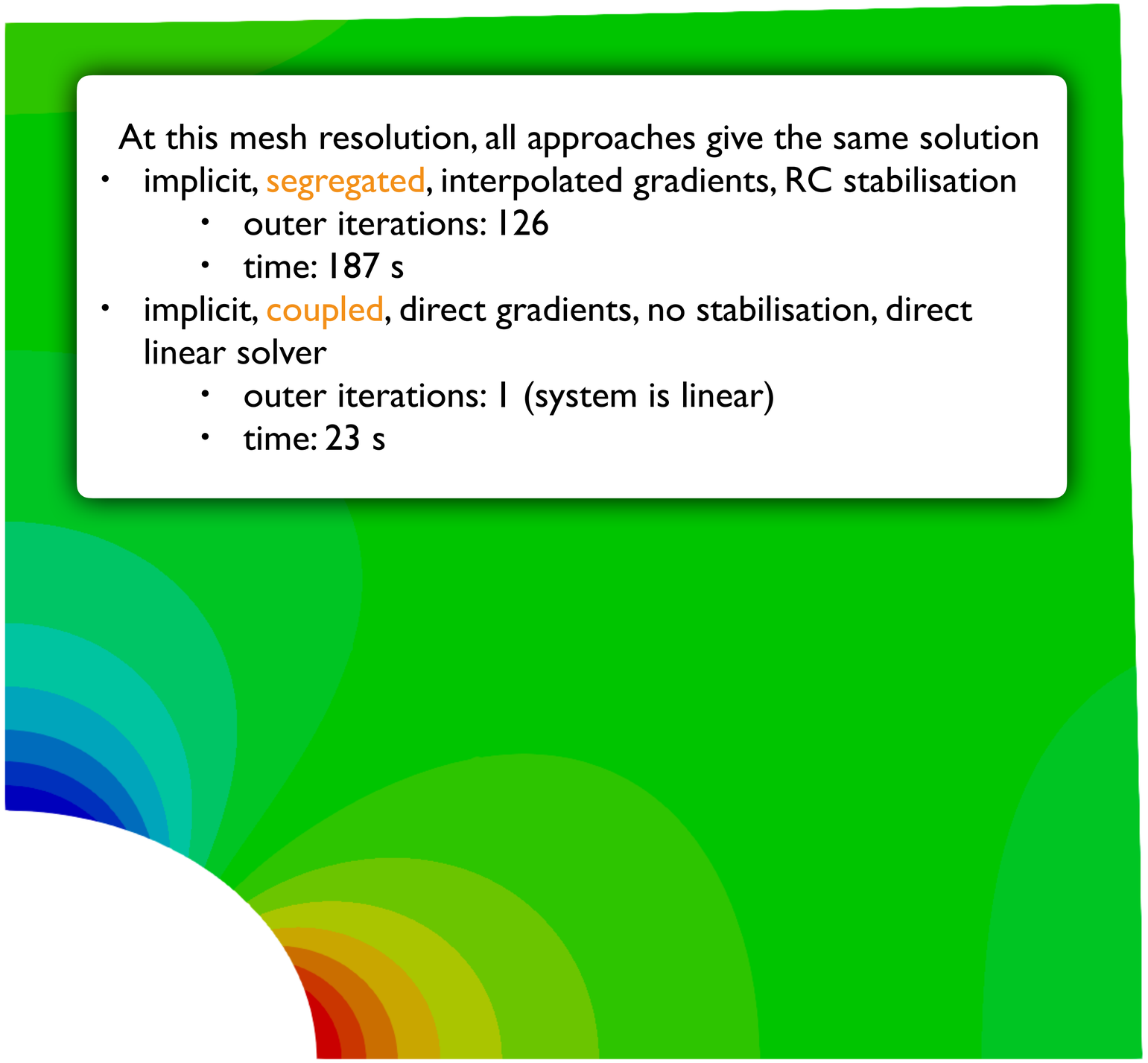
plateHole

- steady-state : 1 time-step, no inertial effects
  - implicit, segregated, interpolated gradients, Rhie-Chow stabilisation
  - implicit, coupled, direct gradients, no stabilisation, direct linear solver

- transient : end-time = 0.01 s, deltaT = 1.066e-05 s
  - explicit, interpolated gradients, JST stabilisation
  - implicit, segregated, interpolated gradients, Rhie-Chow stabilisation
  - implicit, segregated, interpolated gradients, RC stabilisation, ×10 deltaT
  - implicit, coupled, direct gradients, no stabilisation, iterative linear solver

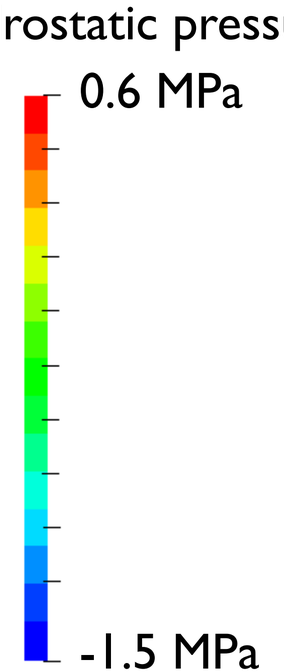


deformation called by
1e+05 for visualisation

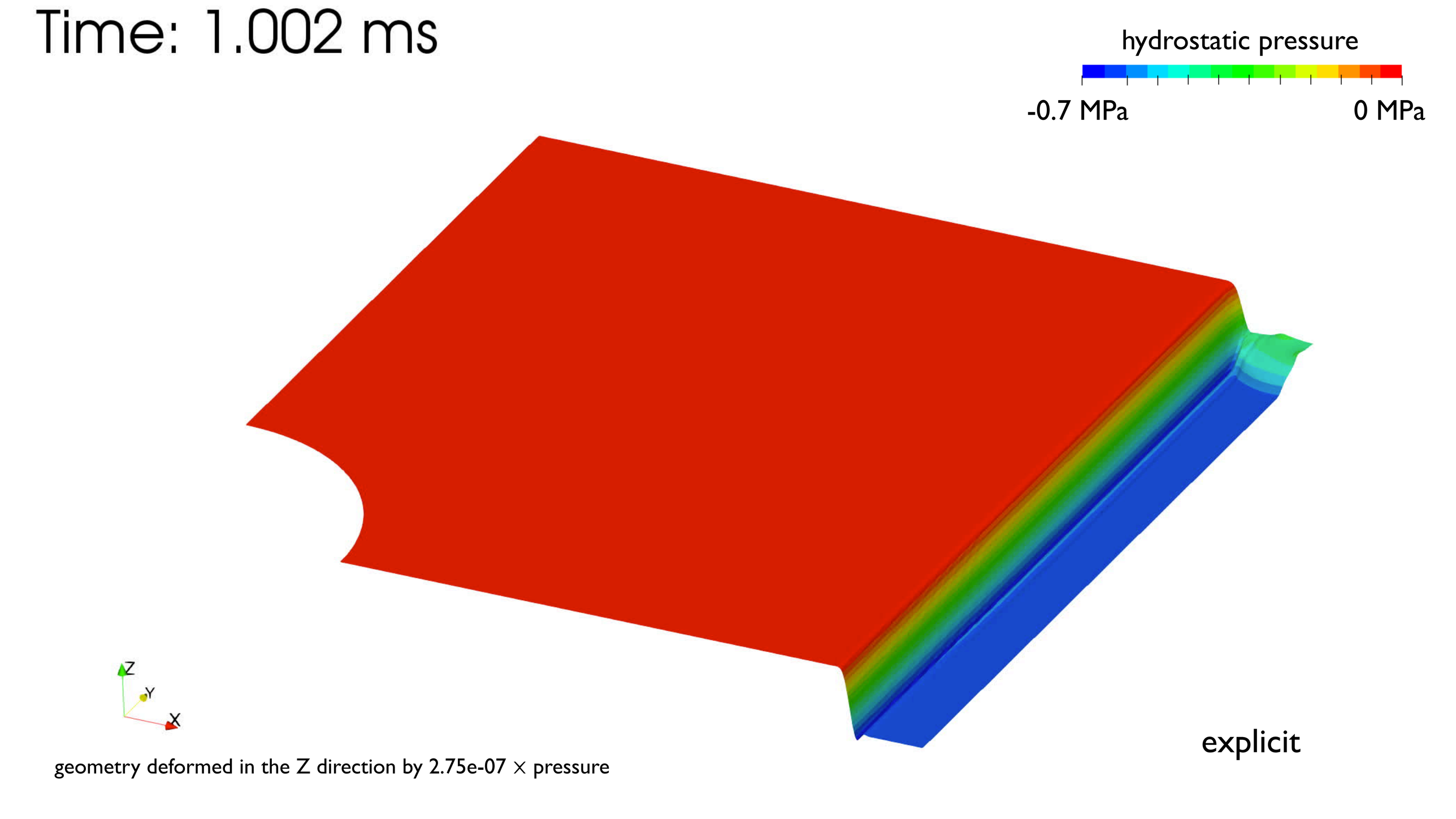


At this mesh resolution, all approaches give the same solution

- implicit, segregated, interpolated gradients, RC stabilisation
  - outer iterations: 126
  - time: 187 s
- implicit, coupled, direct gradients, no stabilisation, direct linear solver
  - outer iterations: 1 (system is linear)
  - time: 23 s



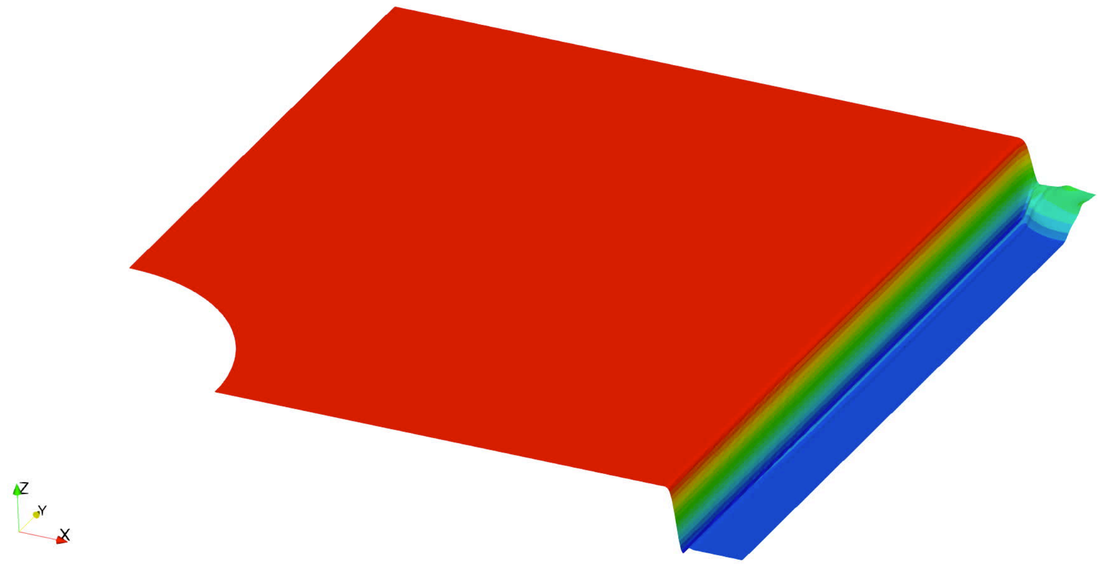
deformation called by 1e+04 for visualisation


Time: 1.002 ms
hydrostatic pressure
-0.7 MPa
0 MPa
Z
Y
X
explicit
geometry deformed in the Z direction by 2.75e-07 × pressure

hydrostatic pressure

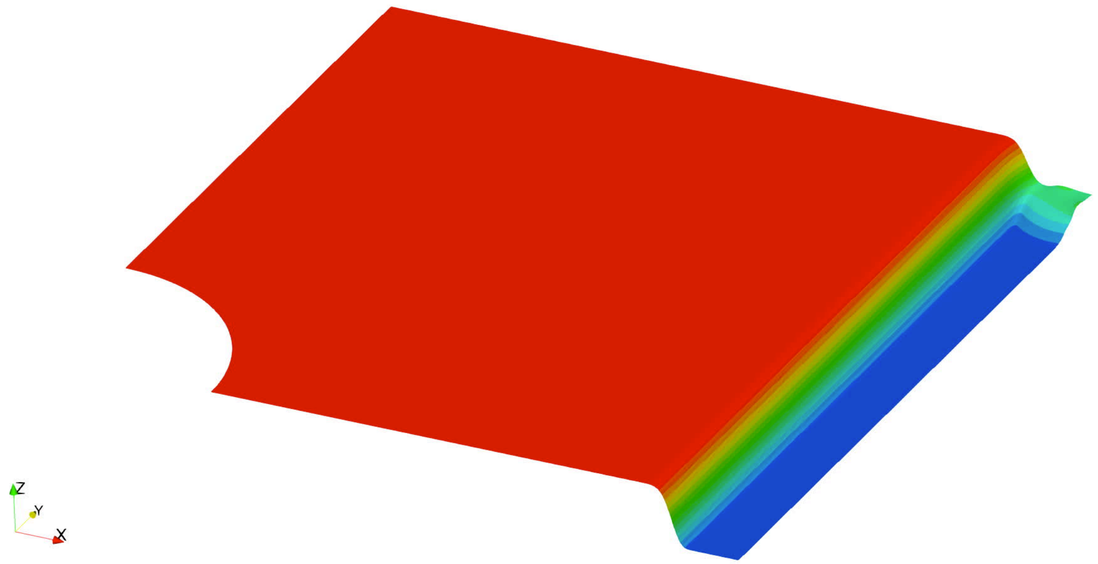
-0.7 MPa 0 MPa

Time: 1.002 ms



explicit

Time: 1.002 ms

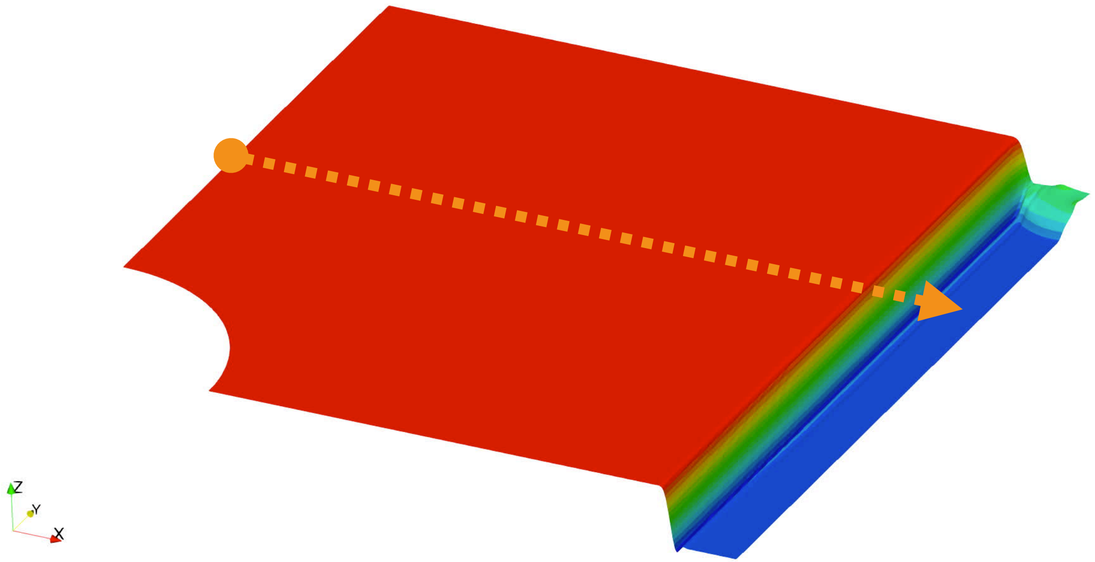


implicit, segregated

hydrostatic pressure

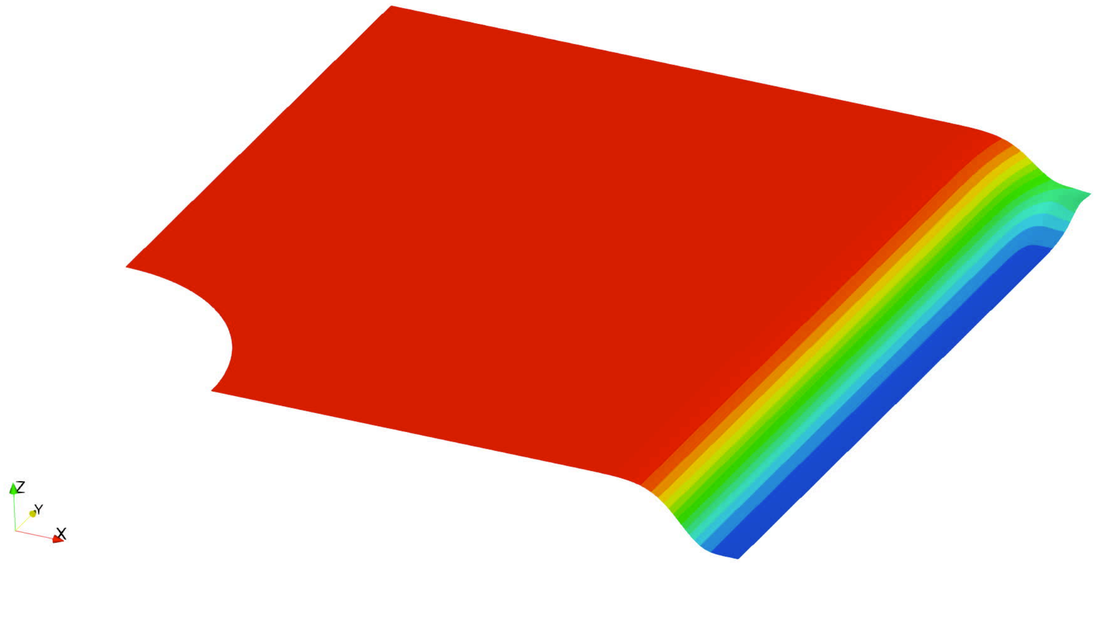
-0.7 MPa 0 MPa

Time: 1.002 ms



explicit

Time: 0.959 ms



implicit, segregated
×10 deltaT

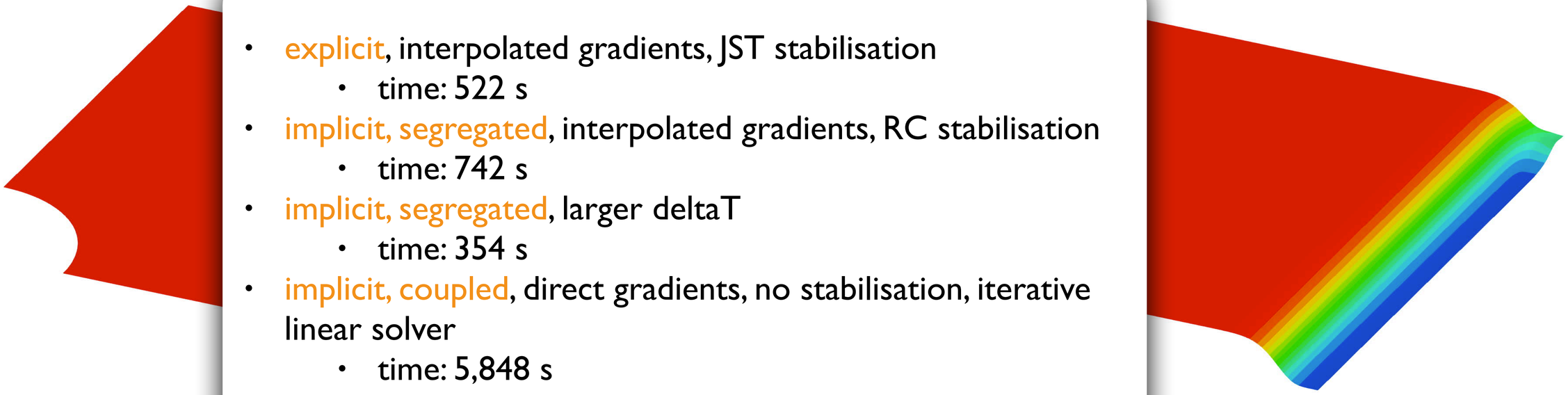

explicit
implicit
implicit - larger deltaT
0
-50000
-100000
-150000
-200000
-250000
-300000
-350000
-400000
-450000
-500000
-550000
-600000
-650000
-700000
-750000
0
0.05
0.1
0.15
0.2
0.25
0.3
0.35
0.4
0.45
0.5
0.55
0.6
0.65
0.7
0.75
0.8
0.85
0.9
0.95
1
1.05
1.1
1.15
1.2
1.25
1.3
1.35
1.4
1.45
1.5
1.55
1.6
1.65
1.7
1.75
1.8
1.85
1.9
1.95
2

hydrostatic pressure

-0.7 MPa 0 MPa

Time: 1.002 ms

Time: 0.959 ms



- explicit, interpolated gradients, JST stabilisation
  - time: 522 s
- implicit, segregated, interpolated gradients, RC stabilisation
  - time: 742 s
- implicit, segregated, larger deltaT
  - time: 354 s
- implicit, coupled, direct gradients, no stabilisation, iterative linear solver
  - time: 5,848 s

explicit

implicit, segregated
×10 deltaT

Summary & Conclusions

Summary & Conclusions

- There have been significant developments in the finite volume methods for solid mechanics over the past 30 years
- Classification of the differing finite volume methodologies can be performed in a number of ways, including by: grid arrangement, implicit vs explicit, and by solution stabilisation
- A variety of finite volume solution methodologies for solid mechanics have been presented (but there are many more variants)
- In-depth comparison of the differing approaches is still required, in terms of accuracy, efficiency and robustness for a variety of cases.
- The finite element method is not the only approach suitable for solid mechanics

April 2019

Online International Meeting for Users of OpenFOAM
Published on YouTube

On the analysis of OpenFOAM finite volume solution methodologies for linear elasticity

Philip Cardiff

School of Mechanical and Materials Engineering
University College Dublin