Towards buoyancy driven flows with FEM

December 08, 2017 | Marc Fehling | Civil Security & Traffic (CST)

Introduction

- Marc Fehling
- born 1990
- M. Sc. Physics, RUB, 2015
- since November 2015:
  PhD student at JSC,
  Division Civil Security & Traffic

Table of contents

Motivation: Fire safety in modern architecture

- Modern architecture and large–scale projects (like BER) may not fit in 'building code'.
- Individual considerations necessary:
  - Model experiments, CFD investigations, ...

Figure: Interior of BMW World, Munich.

Modeling fires

Processes to model:

- Fluid dynamics.
  - Navier–Stokes equations.
  - Turbulence modeling.
- Radiation.
  - Discrete Transfer Radiation.
  - Discrete Ordinates.
  - Monte–Carlo.
- Combustion
  - Mixture fraction.
  - Finite rate kinetics.
- Pyrolysis.

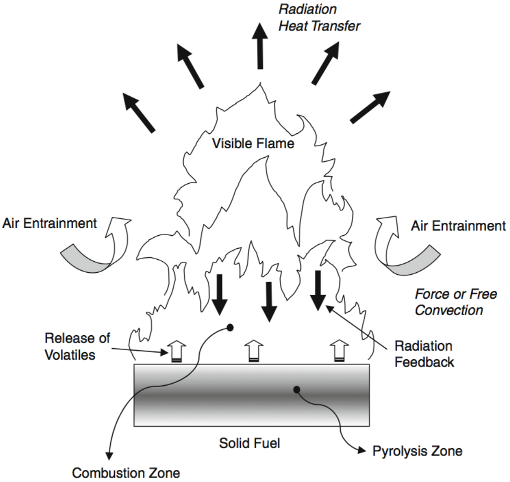


Figure: Burning solid fuel in air with physical processes involved. [1]

Modeling: Equations for smoke propagation

- Smoke propagation with incompressible Navier–Stokes (INS) equations:

$$\nabla \cdot \mathbf{u} = 0$$

$$\rho_0 \left[\partial_t \mathbf{u} + (\mathbf{u} \cdot \nabla)\mathbf{u}\right] + \nabla p - \nabla(2\,\mu\,\epsilon_{ij}(\mathbf{u})) = \mathbf{f}(T)$$

$$\rho_0 \left[\partial_t T + (\mathbf{u} \cdot \nabla) T\right] - 2\,\frac{\mu}{c_\mathrm{p}}\,\epsilon_{ij}(\mathbf{u}) : \nabla \mathbf{u} - \nabla \cdot \left(\frac{\mu}{\mathrm{Pr}}\,\nabla T\right) = \gamma$$

  with strain rate tensor $\epsilon_{ij}(\mathbf{u}) = \frac{1}{2}\left[\nabla \mathbf{u} + (\nabla \mathbf{u})^T\right]$.

- Turbulence model: Smagorinsky–Lilly LES [2]:

$$\nu = \nu_{mol} + \nu_{turb} \qquad \text{with} \qquad \nu_{turb} = (C_s h)^2 \left|\left|\epsilon_{ij}(\mathbf{u})\right|\right|_2 \ .$$

JuFire

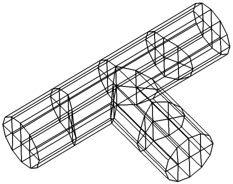
- Investigate applicability of Finite Element Methods (FEM) for fire simulation.
- Use open–source library deal.II [3].
  - 'Toolbox' for the creation of FEM codes.

Differential **E**quations **A**nalysis **L**ibrary

JuFire: Features

Implemented:

- Unstructured grids.
- Continuous Galerkin Methods.
- Adaptive mesh refinement (AMR).
- MPI parallelization.
- Utilization of `CAD` models as manifolds for mesh refinement.

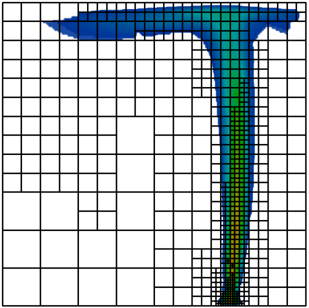


Current field of activity:

- Buoyancy driven flows.

Future work:

- Discontinuous Galerkin methods.
- p–adaptivity.



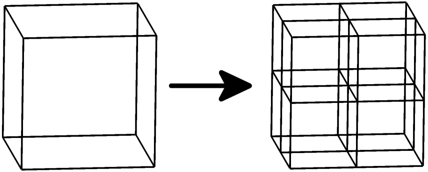
Adaptive mesh refinement (AMR)

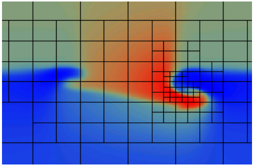
- Solution of differential equations with *FEM* requires:
  - Discretization of space in cells of length h.
  - Shape functions with polynomial degree p.
- But: The more accurate the solution by choosing h and p, the longer the computation.

Adaptive mesh refinement (AMR)

- Solution of differential equations with *FEM* requires:
  - Discretization of space in cells of length *h*.
  - Shape functions with polynomial degree *p*.
- But: The more accurate the solution by choosing *h* and *p*, the longer the computation.

→ **Adaptive refinement** as a 'compromise'.
  - Adjustment of parameters locally where necessary.
  - Variable mesh precision at runtime.





J.Dreher, RUB

AMR: Algorithm

How to determine refinement criteria?

Set up decision criterion for refinement/coarsening.
Our choice:

- Estimate error on each cell. Common: $||\nabla u||$
- Normalize values with respect to all cells.

AMR: Algorithm

How to determine refinement criteria?

Set up decision criterion for refinement/coarsening.
Our choice:

- Estimate error on each cell. Common: $||\nabla u||$
- Normalize values with respect to all cells.
- Flag top/lower part for refinement/coarsening at each step.
- Define max/min refinement levels as upper/lower bounds.

AMR: Algorithm

How to determine refinement criteria?

Set up decision criterion for refinement/coarsening.
Our choice:

- Estimate error on each cell. Common: $||\nabla u||$
- Normalize values with respect to all cells.
- Flag top/lower part for refinement/coarsening at each step.
- Define max/min refinement levels as upper/lower bounds.
- Cells are not allowed to differ by more than one level of refinement.

AMR: Example

- Demonstration of adaptive mesh refinement via moving vortex test case as a shape–preserving potential stream.

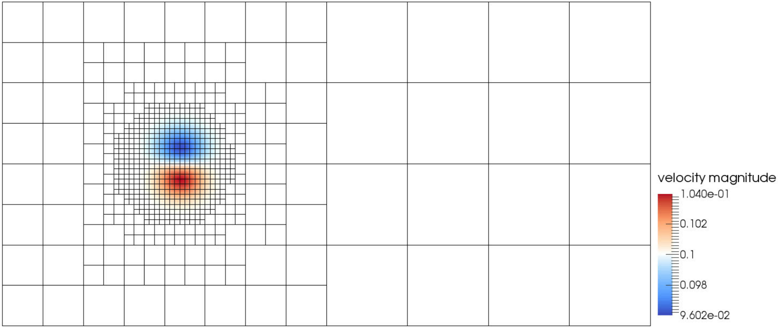


Figure: Video of velocity magnitude of moving vortex, overlayed with corresponding mesh.

AMR: Benefits

- Comparison of runtime and accuracy between uniform and adaptively refined meshes with the moving vortex example.

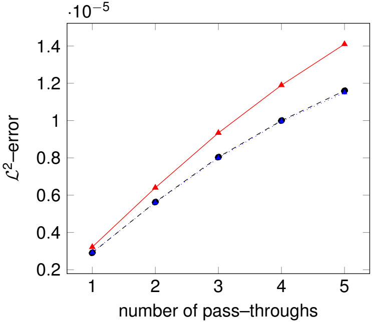


Figure: Global $\mathcal{L}^2$–errors at each periodic pass–through of the vortex.

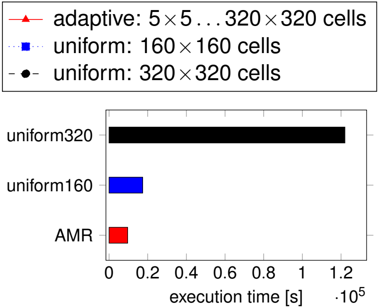


Figure: Execution time of the vortex simulation, run in serial on a common desktop computer.

AMR: Errors

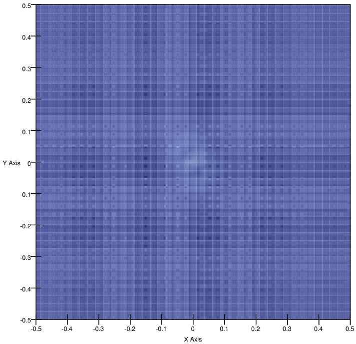


Figure: Static mesh.

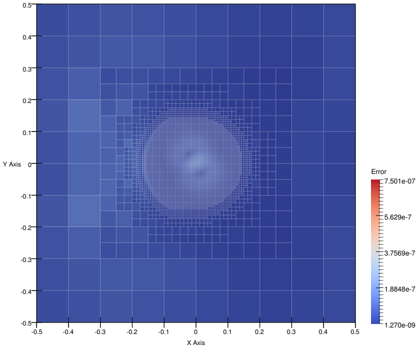


Figure: Adaptive mesh.

1 pass–through.

AMR: Errors

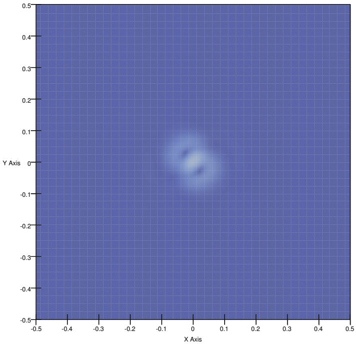


Figure: Static mesh.

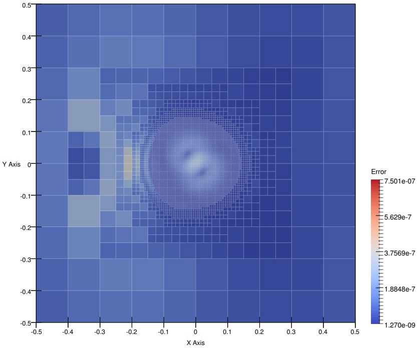


Figure: Adaptive mesh.

2 pass–throughs.

Member of the Helmholtz-Association

AMR: Errors

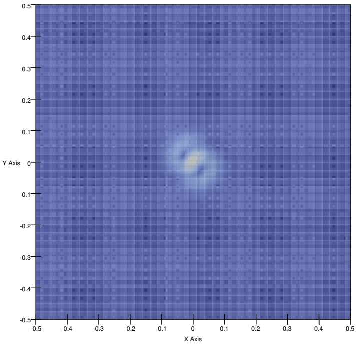


Figure: Static mesh.

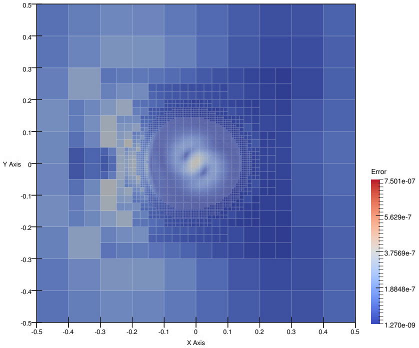


Figure: Adaptive mesh.

3 pass–throughs.

AMR: Errors

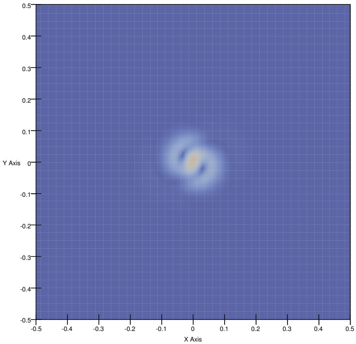


Figure: Static mesh.

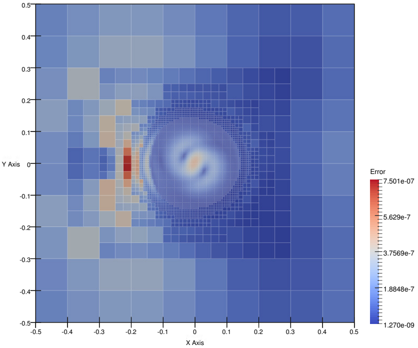


Figure: Adaptive mesh.

4 pass–throughs.

AMR: Errors

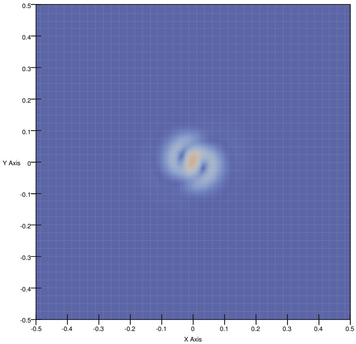


Figure: Static mesh.

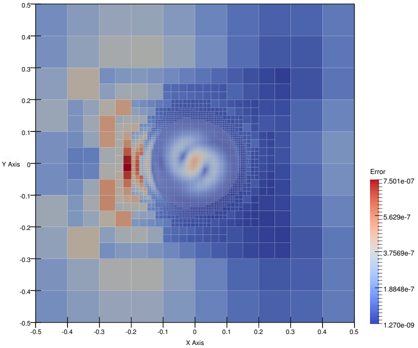


Figure: Adaptive mesh.

5 pass–throughs.

Buoyancy driven flows – Work in progress

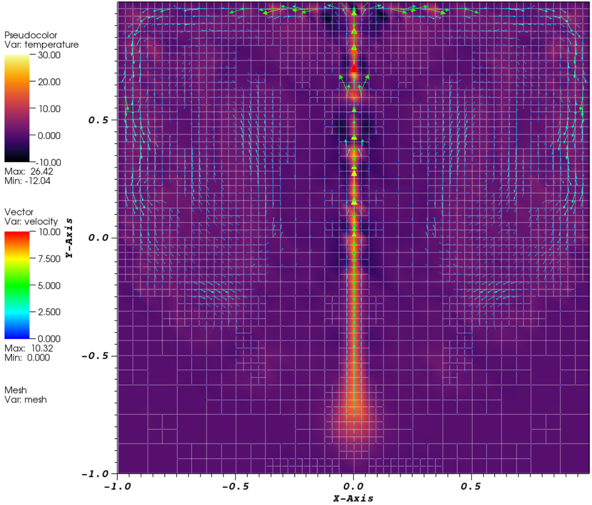


Figure: Video of velocity & temperature with constant heat source.

Buoyancy driven flows – Possible error source

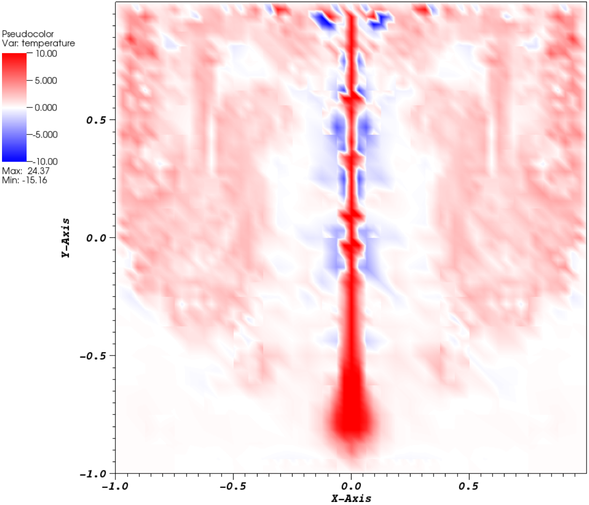


Figure: Temperature with constant heat source.

Heat equation with FEM – Possible error source

$$\partial_t u - \nu \nabla^2 u = 0$$

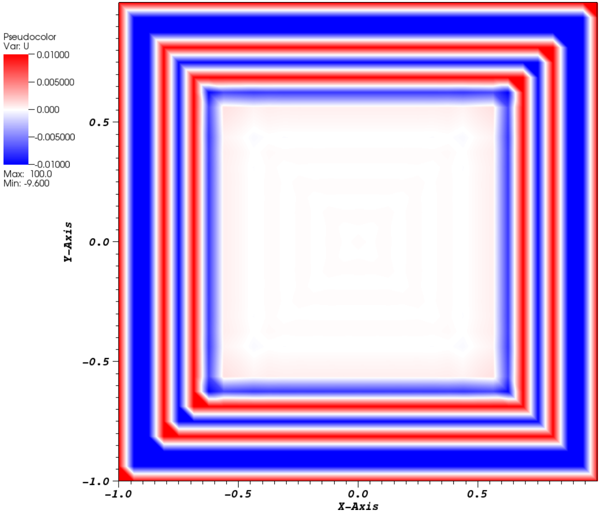


Figure: Solution of heat equation at advanced time.

Flow in exemplary underground – Work in progress

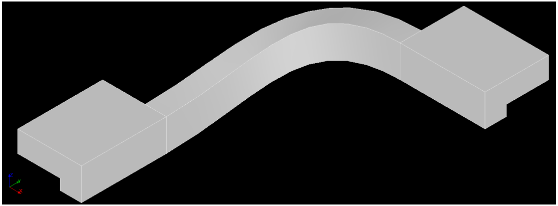


Figure: CAD model of exemplary underground station.

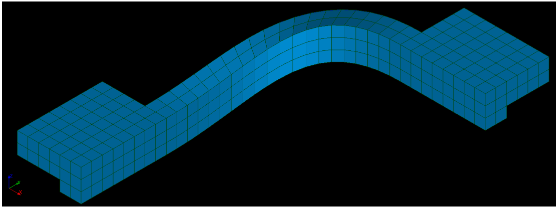


Figure: Initial mesh for simulation.

Summary & Outlook

Summary:

- Adaptive Mesh Refinement works satisfactorily (for now).
- Flaws in solving heat equation need to be cleared.

Future work:

- Implementation of models.
  - Radiation, combustion, pyrolysis, ...
- Extension of numerical methods.
  - DG methods, p-adaptivity, ...
- Comparison with other fire solvers.
- Validation using experiments.

Thank you for your kind attention!

JÜLICH FORSCHUNGSZENTRUM

References I

Guan Heng Yeoh and Kwok Kit Yuen.
Computational Fluid Dynamics in Fire Engineering – Theory, Modelling and Practice.
Butterworth–Heinemann, 1st edition, 2009.

Joseph Smagorinsky.
General circulation experiments with the primitive equations.
Monthly Weather Review, 91(3):99–164, March 1963.

References II

W. Bangerth, D. Davydov, T. Heister, L. Heltai, G. Kanschat, M. Kronbichler, M. Maier, B. Turcksin, and D. Wells.
The `deal.II` library, version 8.4.
Journal of Numerical Mathematics, 24, 2016.

Randy McDermott.
A non–trivial analytical solution to the 2–D incompressible Navier–Stokes equations, August 2003.
[cited 26/09/2016].

Member of the Helmholtz-Association

JÜLICH FORSCHUNGSZENTRUM

References III

- Cedric Taylor and P. Hood.
  A numerical solution of the Navier–Stokes equations using the finite element technique.
  Computers & Fluids, 1:73–100, 1973.

- Bernardino Roig.
  One–step Taylor Galerkin methods for convection–diffusion problems.
  Journal of Computational and Applied Mathematics, 204:95–101, 2007.

References IV

N.D. Heavner and L.G Rebholz.
Locally chosen grad–div stabilization parameters for finite elemnt discretizations of incompressible flow problems.
SIAM Undergraduate Research Online (SIURO), 7.

Towards buoyancy driven flows with FEM: Addendum

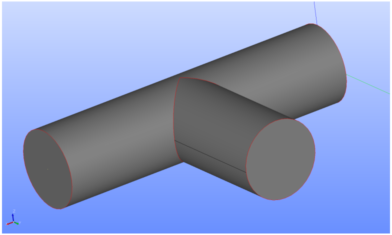
December 08, 2017 | Marc Fehling | Civil Security & Traffic (CST)

JuFire: Algorithm

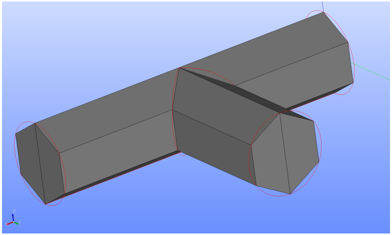
- Time evolution with backward differentiation formula (BDF)
- Taylor–Hood elements [5]: $(\mathbf{u}, p) \in Q_{k+1}^{\dim} \times Q_k$
- Decoupling of $(\mathbf{u}, p)$ by projection scheme
  - Leads to the pressure–Poisson equation
- Stabilization of momentum equation
  - Taylor–Galerkin stabilization [6] for Taylor–Hood elements
    - Additional diffusion in flow direction
  - Grad–div stabilization [7] to enforce $\nabla \cdot \mathbf{u} = 0$
- Neumann series for fast matrix assembly
- Boussinesq approximation for buoyancy force density

Unstructured Grids: Example: T–pipe

- Why T–pipe?
  - Compound bodies.
  - Crooked areas (→ cylinder casing area).
  - Discontinuous edges (→ 'welding seam').
- Develop procedure for the creation of appropriate initial meshes.

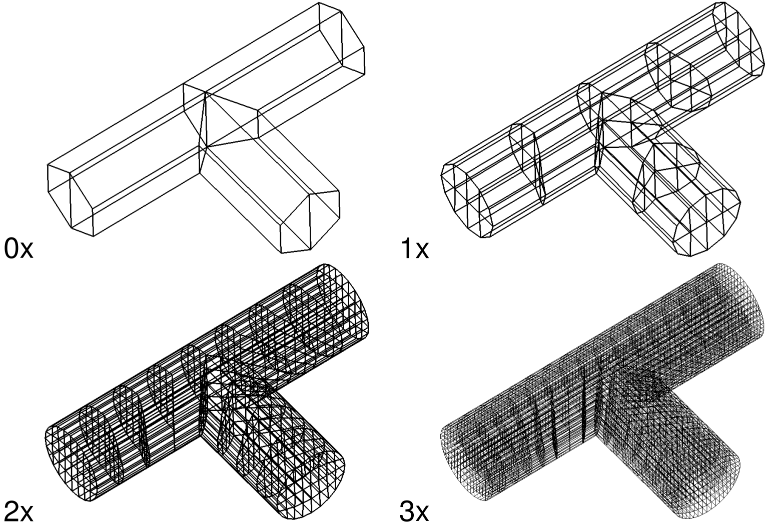


CAD model.



Initial mesh.

Iterations of global refinement:



Numerical methods

Spatial discretization methods for computational fluid dynamics and software packages using it:

- Finite Difference Method (FDM).
  - Open source: NIST Fire Dynamics Simulator (FDS), ...
- Finite Volume Method (FVM).
  - Open source: FireFOAM (OpenFOAM), ...
- Finite Element Method (FEM).
- Lattice–Boltzmann Method (LBM).

FEM: Features

Why use FEM?

FEM: Features

Why use FEM?

- Unstructured grids.
  - Allow for better domain representation without aliasing.

Member of the Helmholtz-Association

JÜLICH FORSCHUNGSZENTRUM

FEM: Features

Why use FEM?

- Unstructured grids.
  - Allow for better domain representation without aliasing.
- hp–adaptivity.
  - Dynamic resolution of numerical grid.
  - Adaptive polynomial degree of basis functions.

    $\rightarrow$ Increase accuracy where the action is happening!

FEM: Features

Why use FEM?

- Unstructured grids.
  - Allow for better domain representation without aliasing.
- hp–adaptivity.
  - Dynamic resolution of numerical grid.
  - Adaptive polynomial degree of basis functions.

    $\rightarrow$ Increase accuracy where the action is happening!
- Discontinuous Galerkin (DG) methods.
  - Allow discontinuities across cell borders.
  - Continuous Galerkin (CG) methods unstable for advection like problems, thus require stabilization.

FEM: Formulation

- Solve variational equation from 'weak' formulation of the differential equation with bilinear form $a(u, v)$:

$$\exists u \in V : \forall v \in V : a(u, v) = f(v)$$

- Choose subspace V_h with basis w_i, out of which the approximate solution $u_\mathrm{h} = \sum u_i w_i \in V_\mathrm{h}$ will be constructed:

$$a(u_\mathrm{h}, w_\mathrm{j}) = \sum a(w_i, w_\mathrm{j})\ u_i = f(w_\mathrm{j}) \quad \rightarrow \quad AU = F$$

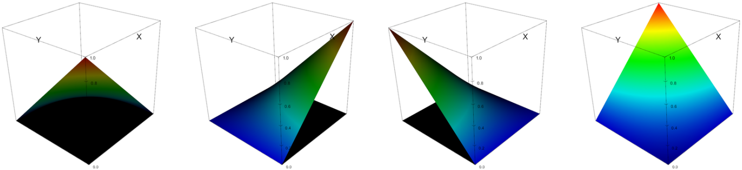


Figure: Q_1 elements in 2D (source: deal.II)

JÜLICH FORSCHUNGSZENTRUM

Verification: Richardson Extrapolation

- How to get convergence rates of time **and** space dependent problems?
- Cancel out redundant error with Richardson extrapolation. Constant ratio r for successive refinement (here: space).

$$\left.\begin{aligned} f_1 - f_0 &= c_t\,\Delta t^{p_t} + c_h\,\Delta h^{p_h} \\ f_2 - f_0 &= c_t\,\Delta t^{p_t} + c_h\,(r\Delta h)^{p_h} \\ f_3 - f_0 &= c_t\,\Delta t^{p_t} + c_h\,(r^2\Delta h)^{p_h} \end{aligned}\right\} \quad \ln\left(\frac{f_3 - f_2}{f_2 - f_1}\right) = p_h\,\ln(r)$$

	Space	Time
FDS	1.9244	2.0721
JuFire	3.0894	0.9715

Table: Examplary convergence rates for a McDermott testcase [4].

HPC on JURECA: Parallelization

- MPI parallelization with Trilinos and p4est through deal.II backends.

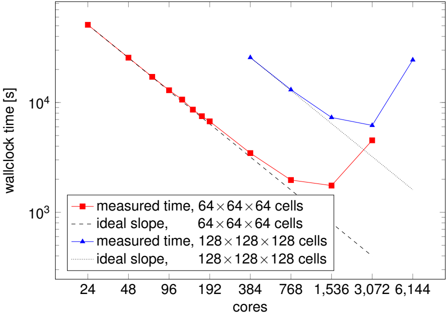


Figure: Strong scaling on JURECA for fixed 3D problems with 262,144 and 2,097,152 cells, respectively.

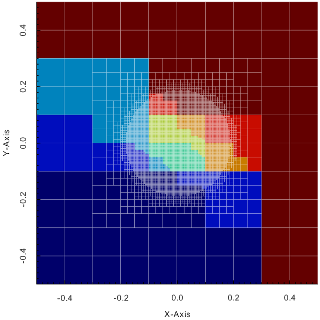


Figure: Examplary domain decomposition with p4est in an AMR case.