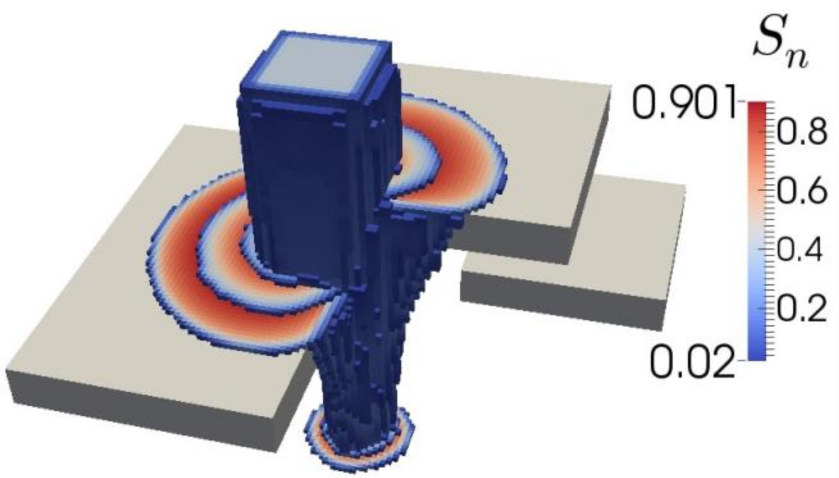
Grid Adaptation with DuMuX

Timo Koch, Martin Schneider
LH2 Stuttgart

Motivation

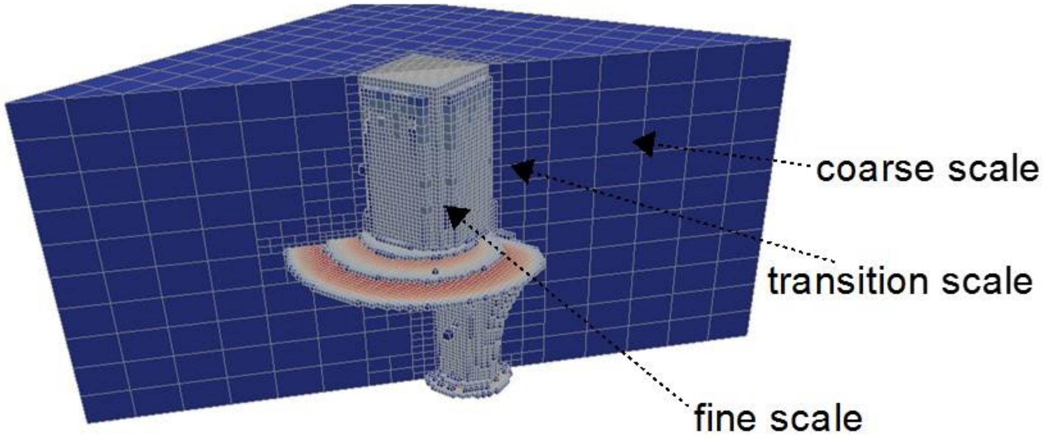
Infiltration process:

Injection in media with lenses



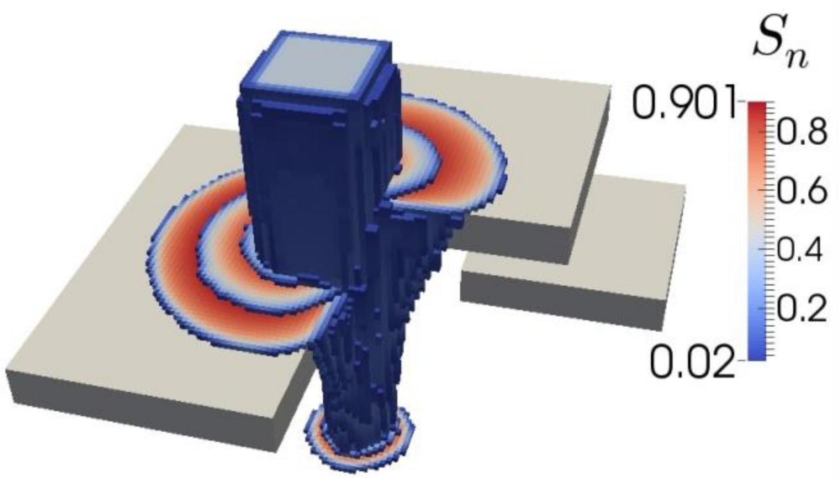
Decoupled Discretization:

- Adaptive Grid: 4.0e4 instead of 1.2e6 cells
- MPFA/TPFA with Adaptive Grid ~ hours
- TPFA with Non-Adaptive Grid ~ days
- MPFA with Non-Adaptive Grid ~ week

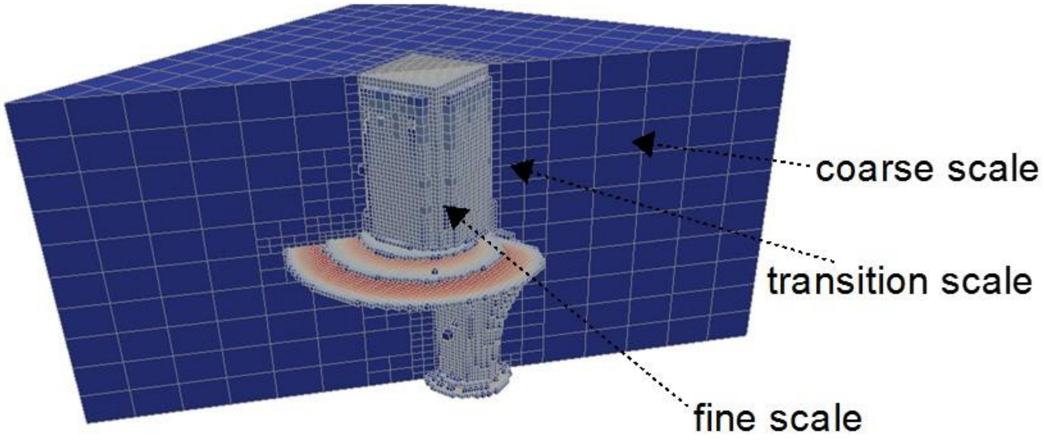


Motivation

Infiltration process:
Injection in media with lenses



→ **Increase efficiency, same accuracy**

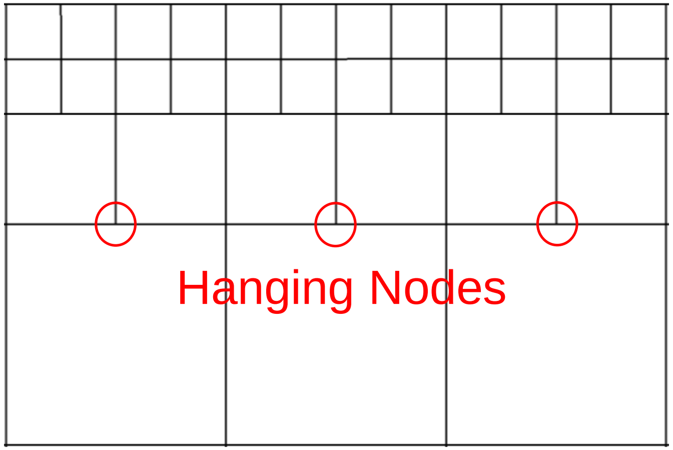


Grid Adaptation

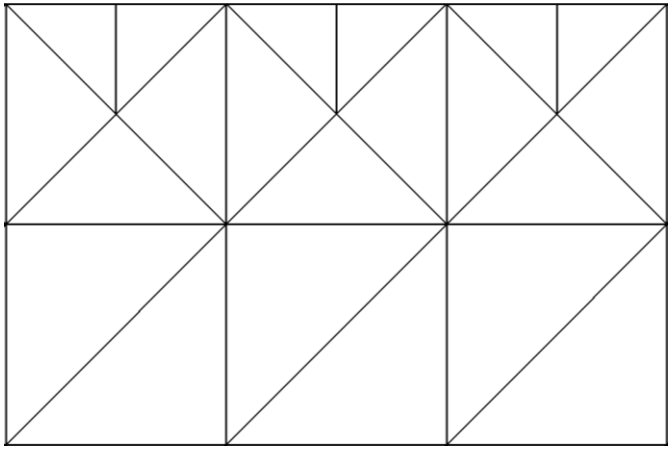
Dune grids supporting adaptation:
Alu, UG, Foam, …

No local adaptation possible:
Yasp, …

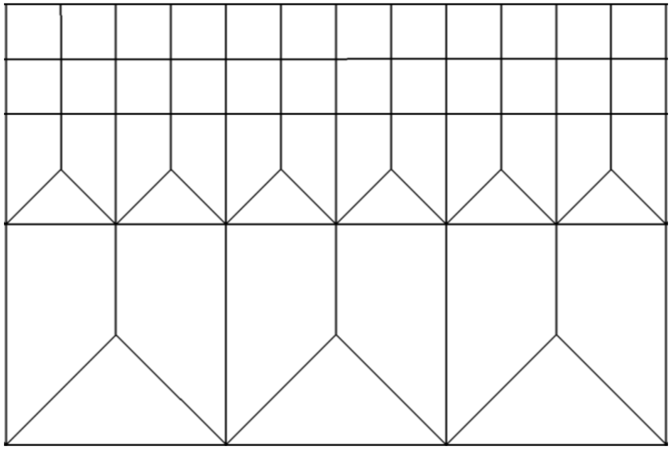
Nonconforming: (Alu)



Conforming: (Alu)

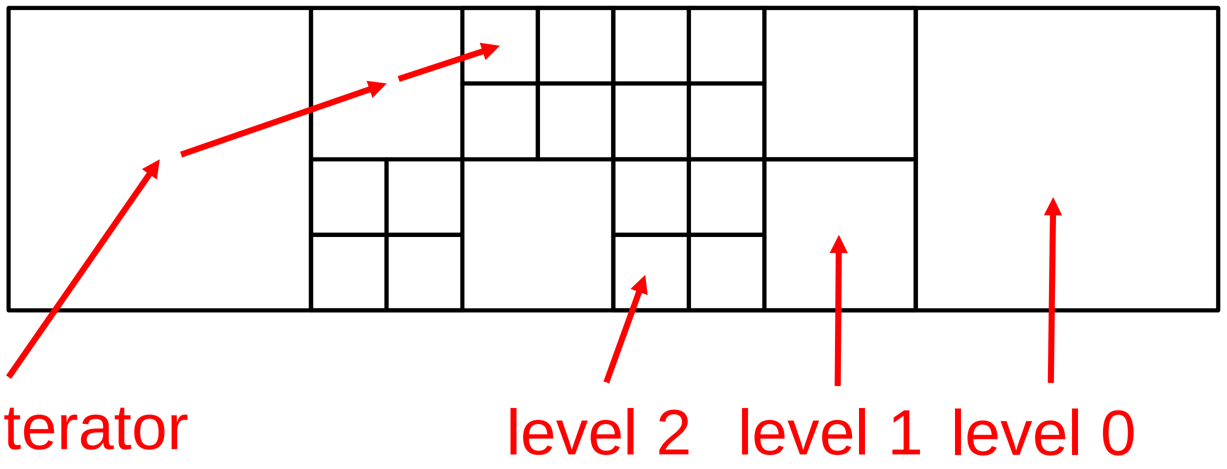


Conforming: (UG)



dune-grid LeafGridView

LeafGridView:



Leaf Iterator

level 2 level 1 level 0

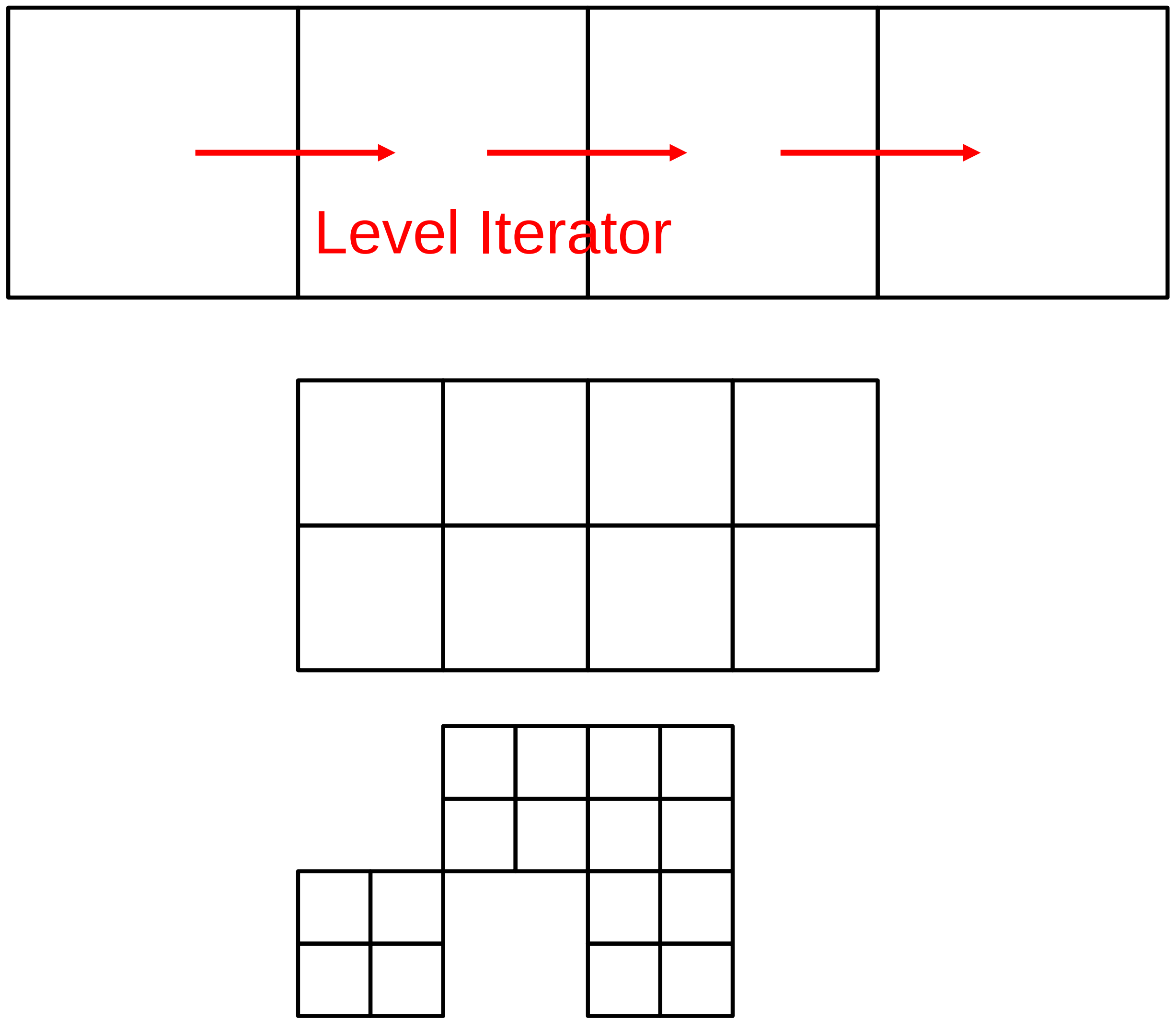
Mostly used in DuMuX

dune-grid Level View

Level 0:

Level 1:

Level 2:



Grid Adaptation

Most of the work is done by dune-grid

Basic routine for adaptation process:

1. Determine cells which should be coarsened or refined
2. Mark grid cells
3. Store primary variables
4. Adapt grid
5. Reconstruct primary variables

Grid Adaptation

Most of the work is done by dune-grid

Basic routine for adaptation process:

1. Determine cells which should be coarsened or refined
2. Mark grid cells *grid.mark(±1, entity)*
3. Store primary variables
4. Adapt grid *grid.adapt()*
5. Reconstruct primary variables

dune-grid

Grid Adaptation

Most of the work is done by dune-grid

Basic routine for adaptation process:

1. Determine cells which should be coarsened or refined
2. Mark grid cells *grid.mark(±1, entity)*
3. Store primary variables
4. Adapt grid *grid.adapt()*
5. Reconstruct primary variables

dune-grid
basic routines in implicit/adaptive

Grid Adaptation

Most of the work is done by dune-grid

Basic routine for adaptation process:

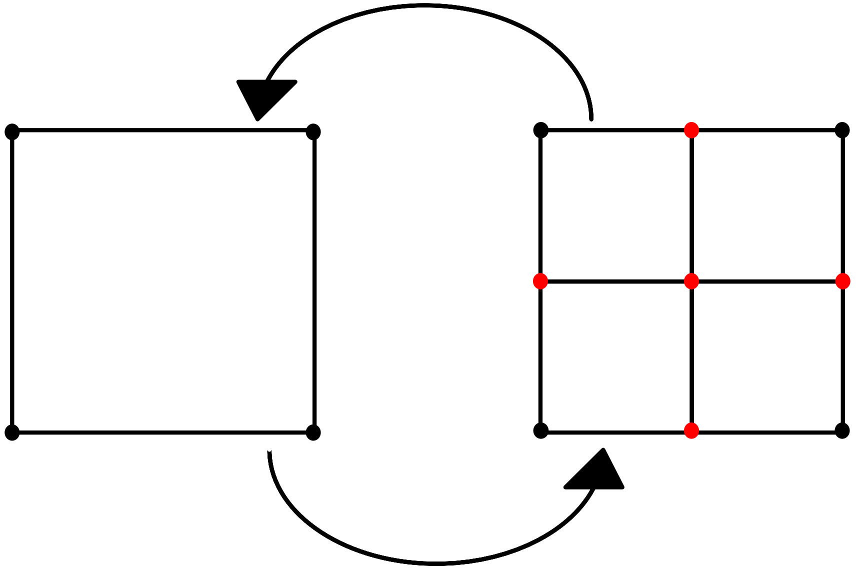
1. Determine cells which should be coarsened or refined
2. Mark grid cells *grid.mark(±1, entity)*
3. Store primary variables
4. Adapt grid *grid.adapt()*
5. Reconstruct primary variables

dune-grid
basic routines in implicit/adaptive
user: model specific, indicator for refinement / coarsening

Reconstruction of primary variables

Box method:

Nothing to do

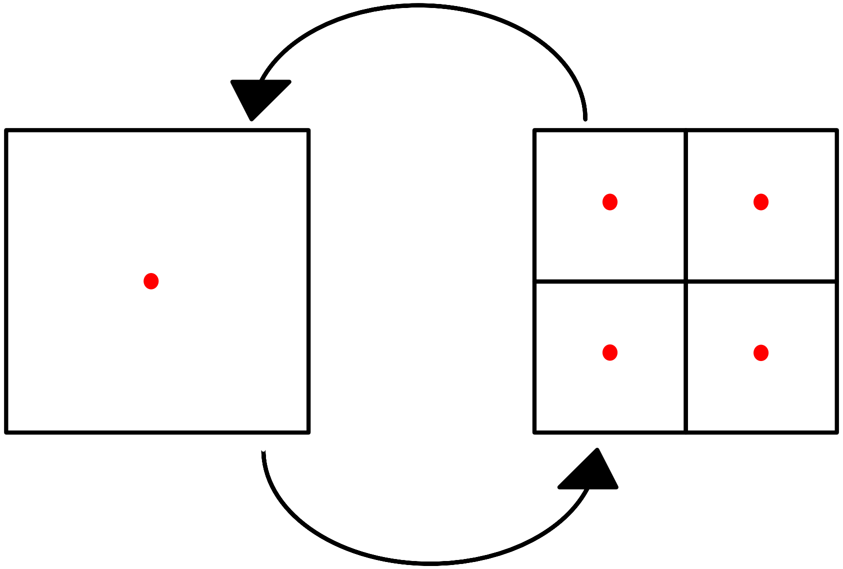


Calculate new values with Ansatz functions

Reconstruction of primary variables

CC FV method:

Averaging of values



Same value as father

Indicator

Two types of indicators:

- *Initialization Indicator* (refine at boundaries / source)
- *Runtime Indicator* (refine, coarse cells during runtime)

Example: Indicator for Saturation

$$\mathcal{I}(e_i) = \max_{j \in \mathcal{N}(i)} |S_i - S_j|$$

refine: $\mathcal{I}(e_i) > \mathrm{TOL}_{ref} |S^{\max} - S^{\min}|$

coarse: $\mathcal{I}(e_i) < \mathrm{TOL}_{coar} |S^{\max} - S^{\min}|$

Basic GridAdapt Properties

General Properties

- *AdaptiveGrid (true or false)*
- *AdaptionIndicator, AdaptionInitializationIndicator*

Initialization Indicator

- *RefineAtDirichletBC, RefineAtFluxBC, RefineAtSource*

Runtime Indicator

- *MinLevel, MaxLevel*
- *RefineTolerance, CoarsenTolerance*

Adaptation during runtime

Using adaptivity for the implicit model:

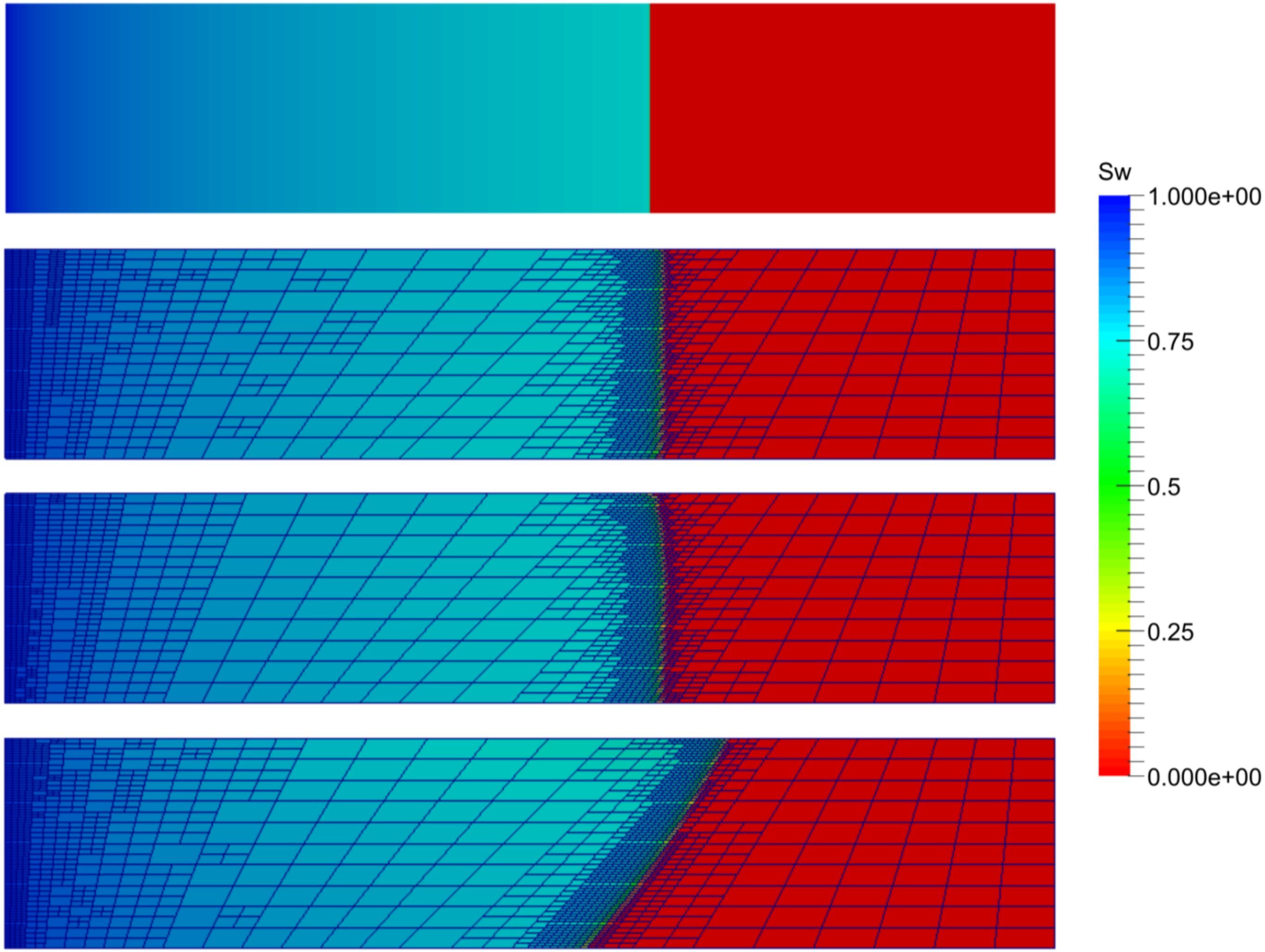
- Main routine is called in: *problem.preTimeStep()*

- Construction of new jacobian matrix in: *model.updateBegin()*

Important Facts

- Box method can not be used for nonconforming adaptation
- Cell Centered TPFA method may produce huge errors at hanging nodes
- This could influence convergence of Newton solver
- Adaptivity is not yet implemented for all solvers

Accuracy



Exact:

MPFA:

NLTPFA:

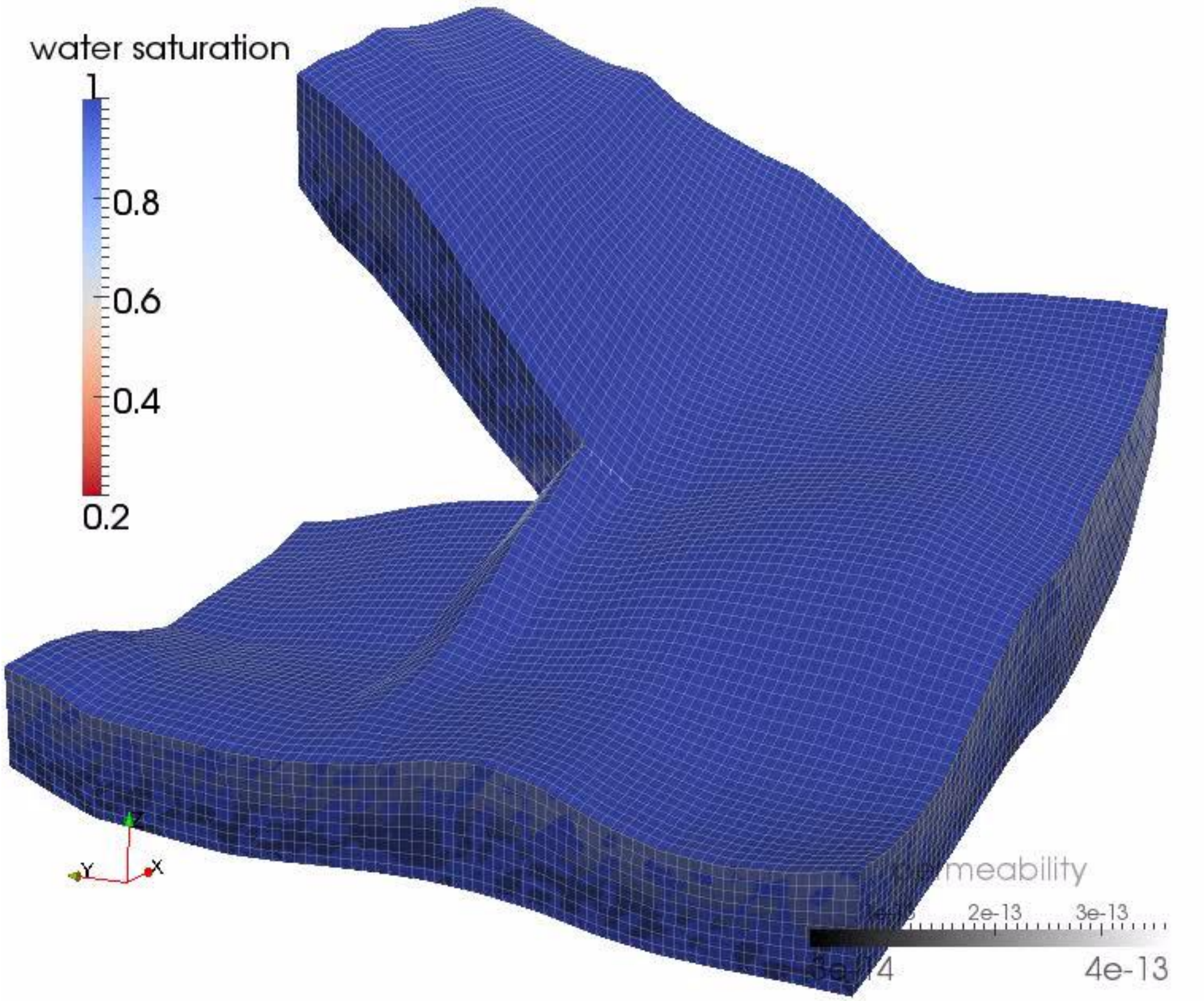
TPFA:

www.hydrosys.uni-stuttgart.de

Complex Example

CO_2 injection into

Johanson formation



(B. Faigle: Adaptive Multi-Physics)