Dynamic Coarsening for Geothermal Applications

Øystein Klemetsdal, Computational Geosciences, SINTEF Digital

ECCOMAS 2022, June 5 – 9, Oslo, Norway

Presentation outline

- Motivation
- Governing equations and discretization
- The MATLAB Reservoir Simulation Toolbox
- Dynamic coarsening
- Numerical examples
- Concluding remarks

Motivation

- Geothermal heat is an appealing resource for energy production and storage
  - Renewable ✓ Always on ✓ Available anywhere ✓ Low carbon footprint ✓
- Viability depends a number of factors (Glassley 2010; Stober and Bucher 2013)
  - Efficiency, storage capacity, operational and drilling costs, legal regulations, ...
- Assessment requires solid system knowledge (Andersson 2007)
  - Aquifer/aquiclude geology, groundwater chemistry, flow properties, ...

Motivation

- Geothermal heat is an appealing resource for energy production and storage
  - Renewable ✓ Always on ✓ Available anywhere ✓ Low carbon footprint ✓
- Viability depends a number of factors (Glassley 2010; Stober and Bucher 2013)
  - Efficiency, storage capacity, operational and drilling costs, legal regulations, ...
- Assessment requires solid system knowledge (Andersson 2007)
  - Aquifer/aquiclude geology, groundwater chemistry, flow properties, ...

Complexity and size typically renders numerical simulations the only viable option
(O'Sullivan, Pruess, and Lippmann 2000; K. S. Lee 2010; Stober and Bucher 2013)

Presentation outline

- Motivation
- Governing equations and discretization
- The MATLAB Reservoir Simulation Toolbox
- Dynamic coarsening
- Numerical examples
- Concluding remarks

Governing equations and discretization

Single-phase conservation of mass on residual form

$$R_f = \partial_t(\phi \rho_f) + \nabla \cdot (\rho_f \vec{v}_f) - \rho_f q_f = 0$$

- Velocity given by Darcy's law: $\vec{v}_f = -\frac{1}{\mu_f}\boldsymbol{K}(\nabla p - \rho_f g \nabla \vec{z})$

ϕ	Pore volume	$\boldsymbol{K}$	Permeability	$\boldsymbol{\Lambda}$	Thermal conductivity	$\vec{g}$	Gravity
ρ	Density	μ	Viscosity	u	Internal energy	h	Enthalpy
p	Pressure	T	Temperature	q	Sources/sinks	r/f	Fluid/rock

Governing equations and discretization

Single-phase conservation of mass on residual form

$$R_f = \partial_t M_f + \nabla \cdot \vec{V}_f - Q_f = 0$$

Mass — Flux — Sources/sinks

- Velocity given by Darcy's law: $\vec{v}_f = -\frac{1}{\mu_f}\boldsymbol{K}(\nabla p - \rho_f g \nabla \vec{z})$

ϕ	Pore volume	$\boldsymbol{K}$	Permeability	$\boldsymbol{\Lambda}$	Thermal conductivity	$\vec{g}$	Gravity
ρ	Density	μ	Viscosity	u	Internal energy	h	Enthalpy
p	Pressure	T	Temperature	q	Sources/sinks	r/f	Fluid/rock

Governing equations and discretization

Conservation of energy on residual form

$$R_e = \partial_t(\phi\rho_f u_f + [1-\phi]\rho_r u_r) + \nabla\cdot(\rho_f\vec{v}_f h_f + \vec{H}) - \rho_f q_f h_f = 0$$

- Conductive heat flux from Fourier's law: $\vec{H} = -(\boldsymbol{\Lambda}_f + \boldsymbol{\Lambda}_r)\nabla T$

ϕ	Pore volume	$\boldsymbol{K}$	Permeability	$\boldsymbol{\Lambda}$	Thermal conductivity	$\vec{g}$	Gravity
ρ	Density	μ	Viscosity	u	Internal energy	h	Enthalpy
p	Pressure	T	Temperature	q	Sources/sinks	r/f	Fluid/rock

Governing equations and discretization

Conservation of energy on residual form

Fluid energy — Rock energy

$$R_e = \partial_t(M_f u_f + M_r u_r) + \nabla \cdot (\vec{V}_f h_f + \vec{H}) - Q_f h_f = 0$$

Advective heat flux — Conductive heat flux

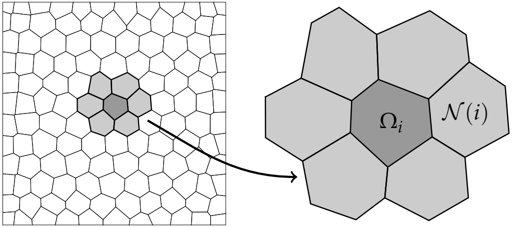
- Conductive heat flux from Fourier's law: $\vec{H} = -(\boldsymbol{\Lambda}_f + \boldsymbol{\Lambda}_r)\nabla T$

ϕ	Pore volume	$\boldsymbol{K}$	Permeability	$\boldsymbol{\Lambda}$	Thermal conductivity	$\vec{g}$	Gravity
ρ	Density	μ	Viscosity	u	Internal energy	h	Enthalpy
p	Pressure	T	Temperature	q	Sources/sinks	r/f	Fluid/rock

Governing equations and discretization

Finite volumes in space, implicit backward Euler in time

$$\boldsymbol{R}^{n+1} = \frac{1}{\Delta t^n}(\boldsymbol{M}^{n+1} - \boldsymbol{M}^n) + \texttt{div}(\boldsymbol{V}^{n+1}) - \boldsymbol{Q}^{n+1} = 0$$



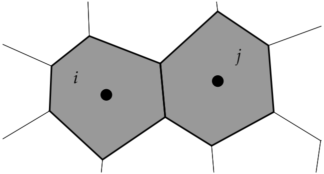
Governing equations and discretization

Finite volumes in space, implicit backward Euler in time

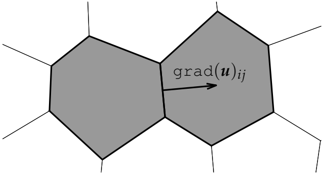
$$\boldsymbol{R}^{n+1} = \frac{1}{\Delta t^n}(\boldsymbol{M}^{n+1} - \boldsymbol{M}^n) + \texttt{div}(\boldsymbol{V}^{n+1}) - \boldsymbol{Q}^{n+1} = 0$$

$$\boldsymbol{V} = -\texttt{upw}(\boldsymbol{\rho}/\boldsymbol{\mu})[\texttt{Kgrad}(\boldsymbol{p}) - g\texttt{favg}(\boldsymbol{\rho})\texttt{Kgrad}(\boldsymbol{z})]$$

- Kgrad: discrete operator $\boldsymbol{K}\nabla$ (linear/nonlinear two-point, multipoint, mimetic, etc.)
  - In this work: linear two-point flux approximation (comparison: Ø. Klemetsdal et al. 2020)



Scalar field $\boldsymbol{u}$



Gradient $\texttt{grad}(\boldsymbol{u})_{ij} = u_j - u_i$

$$\texttt{Kgrad} = \boldsymbol{T}\texttt{grad}$$

$$\boldsymbol{T}_{ij} = \left(\boldsymbol{T}_{i,j}^{-1} + \boldsymbol{T}_{j,i}^{-1}\right)^{-1}$$

$$\boldsymbol{T}_{i,j} = |F_{ij}|\frac{\vec{c}_{i,j}\boldsymbol{K}_i\vec{n}_{i,j}}{|\vec{c}_{i,j}|^2}$$

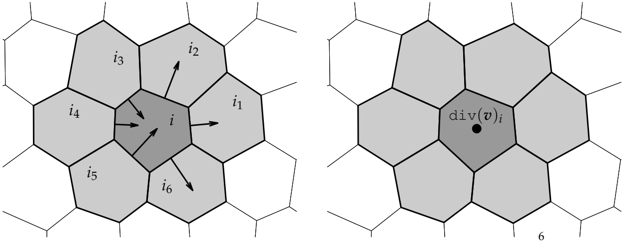
Governing equations and discretization

Finite volumes in space, implicit backward Euler in time

$$\boldsymbol{R}^{n+1} = \frac{1}{\Delta t^n}(\boldsymbol{M}^{n+1} - \boldsymbol{M}^n) + \texttt{div}(\boldsymbol{V}^{n+1}) - \boldsymbol{Q}^{n+1} = 0$$

$$\boldsymbol{V} = -\texttt{upw}(\boldsymbol{\rho}/\boldsymbol{\mu})[\texttt{Kgrad}(\boldsymbol{p}) - g\texttt{favg}(\boldsymbol{\rho})\texttt{Kgrad}(\boldsymbol{z})]$$

- `div`: discrete divergence operator



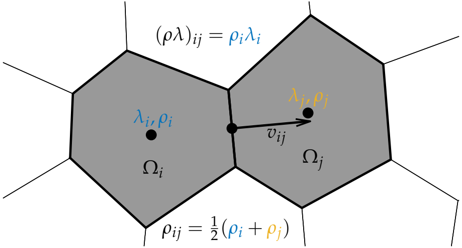
Interface flux field $\boldsymbol{v}$ Divergence $\texttt{div}(\boldsymbol{v})_i = \sum_{k=1}^{6} v_{ii_k}$

Governing equations and discretization

Finite volumes in space, implicit backward Euler in time

$$\boldsymbol{R}^{n+1} = \frac{1}{\Delta t^n}(\boldsymbol{M}^{n+1} - \boldsymbol{M}^n) + \texttt{div}(\boldsymbol{V}^{n+1}) - \boldsymbol{Q}^{n+1} = 0$$
$$\boldsymbol{V} = -\texttt{upw}(\boldsymbol{\rho}/\boldsymbol{\mu})[\texttt{Kgrad}(\boldsymbol{p}) - g\texttt{favg}(\boldsymbol{\rho})\texttt{Kgrad}(\boldsymbol{z})]$$

- upw: Upwind discretization (single-point here); favg: Face average operator



Governing equations and discretization

Finite volumes in space, implicit backward Euler in time

$$\boldsymbol{R}^{n+1} = \frac{1}{\Delta t^n}(\boldsymbol{M}^{n+1} - \boldsymbol{M}^n) + \texttt{div}(\boldsymbol{V}^{n+1}) - \boldsymbol{Q}^{n+1} = 0$$

Newton's method: make system $\boldsymbol{R}(\boldsymbol{x}) = \boldsymbol{0}$, linearize, neglect higher-order terms

$$\boldsymbol{x}^{k+1} = \boldsymbol{x}^k + \Delta\boldsymbol{x}, \qquad -\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{x}}\Delta\boldsymbol{x} = \boldsymbol{R}(\boldsymbol{x}^k)$$

Sequential implicit formulation

1. Form pressure equation as weighted sum of $\boldsymbol{R}_f$ and $\boldsymbol{R}_e$

$$\boldsymbol{R}_p = \omega_f \boldsymbol{R}_f + \omega_e \boldsymbol{R}_e, \ \partial_{\boldsymbol{x}}\big(\omega_f \boldsymbol{M}_f^{n+1}\big) + \partial_{\boldsymbol{x}}\big(\omega_e [\boldsymbol{M}_f \boldsymbol{u}_f + \boldsymbol{M}_r \boldsymbol{u}_r]^{n+1}\big) = \mathbf{0}, \ \boldsymbol{x} \neq \text{pressure}$$

2. Solve $\boldsymbol{R}_p = \mathbf{0}$ with fixed temperature and transport variables $\rightarrow$ pressure + intercell fluxes
3. Solve $\boldsymbol{R}_f = \mathbf{0}$ and $\boldsymbol{R}_e = \mathbf{0}$ with fixed pressure and intercell fluxes $\rightarrow$ temperature + transport

Sequential implicit formulation

1. Form pressure equation as weighted sum of $\boldsymbol{R}_f$ and $\boldsymbol{R}_e$

$$\boldsymbol{R}_p = \omega_f \boldsymbol{R}_f + \omega_e \boldsymbol{R}_e, \ \partial_{\mathbf{x}}\left(\omega_f \boldsymbol{M}_f^{n+1}\right) + \partial_{\mathbf{x}}\left(\omega_e [\boldsymbol{M}_f \boldsymbol{u}_f + \boldsymbol{M}_r \boldsymbol{u}_r]^{n+1}\right) = \mathbf{0}, \ \boldsymbol{x} \neq \text{pressure}$$

2. Solve $\boldsymbol{R}_p = \mathbf{0}$ with fixed temperature and transport variables $\rightarrow$ pressure + intercell fluxes
3. Solve $\boldsymbol{R}_f = \mathbf{0}$ and $\boldsymbol{R}_e = \mathbf{0}$ with fixed pressure and intercell fluxes $\rightarrow$ temperature + transport

Transport formulation: solve for **temperature** $\boldsymbol{T}$ and **total saturation** $\boldsymbol{S}_t$
$\rightarrow$ allow total saturation to be $\neq 1$, multiply densities by total saturation

$$\rho_f \rightarrow \boldsymbol{S}_t \rho_f, \quad \rho_r \rightarrow \boldsymbol{S}_t \rho_r$$

Presentation outline

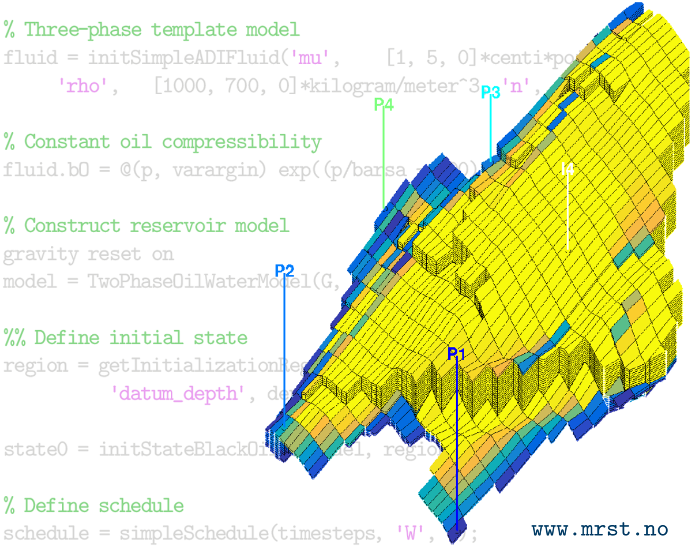
- Motivation
- Governing equations and discretization
- The MATLAB Reservoir Simulation Toolbox
- Dynamic coarsening
- Numerical examples
- Concluding remarks

MATLAB Reservoir Simulation Toolbox (MRST)

Transforming research on reservoir modelling

Unique prototyping platform:

- Standard data formats
- Data structures/library routines
- Fully unstructured grids
- Rapid prototyping:
  - Differentiation operators
  - Automatic differentiation
  - Object-oriented framework
  - State functions
- Industry-standard simulation



MATLAB Reservoir Simulation Toolbox (MRST)

Transforming research on reservoir modelling

Unique prototyping platform:

- Standard data formats
- Data structures/library routines
- Fully unstructured grids
- Rapid prototyping:
  - **Differentiation operators**
  - Automatic differentiation
  - Object-oriented framework
  - State functions
- Industry-standard simulation

Differentiation operators

Write discrete equations on form very close to continuous equations

$\nabla \cdot \vec{H}$ $\qquad \vec{H} = -(\lambda_f + \lambda_r)\nabla T$

```
div(H)      H = -(lambdaF + lambdaR).*grad(T)
```

MATLAB Reservoir Simulation Toolbox (MRST)

Transforming research on reservoir modelling

Unique prototyping platform:

- Standard data formats
- Data structures/library routines
- Fully unstructured grids
- Rapid prototyping:
  - Differentiation operators
  - **Automatic differentiation**
  - Object-oriented framework
  - State functions
- Industry-standard simulation

Differentiation operators

Write discrete equations on form very close to continuous equations

$\nabla \cdot \vec{H}$ $\qquad$ $\vec{H} = -(\lambda_f + \lambda_r)\nabla T$

```
div(H)      H = -(lambdaF + lambdaR).*grad(T)
```

Automatic differentiation

Combine chain rule and elementary differentiation rules by means of operator overloading to analytically evaluate all derivatives

$\rightarrow$ *Computing Jacobians amounts to writing down residual equations.*

```
[x,y] = initVariablesADI(1,2); z = 3*exp(-x*y)
```

```
x = ADI Properties:        y = ADI Properties:        z = ADI Properties:
    val: 1                     val: 2                     val: 0.4060
    jac: {[1]  [0]}            jac: {[0]  [1]}            jac: {[-0.8120]  [-0.4060]}
```

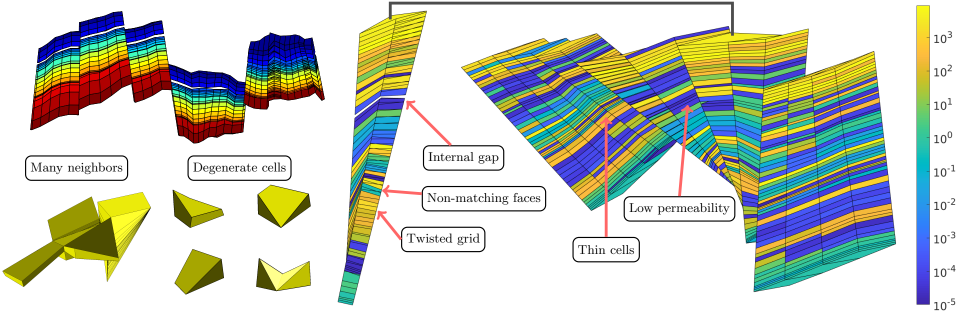
$\frac{\partial x}{\partial x}$ $\quad$ $\frac{\partial x}{\partial y}$ $\quad$ $\frac{\partial y}{\partial x}$ $\quad$ $\frac{\partial y}{\partial y}$ $\quad$ $\frac{\partial z}{\partial x}\Big|_{x=1,y=2}$ $\quad$ $\frac{\partial z}{\partial y}\Big|_{x=1,y=2}$

Presentation outline

- Motivation
- Governing equations and discretization
- The MATLAB Reservoir Simulation Toolbox
- Dynamic coarsening
- Numerical examples
- Concluding remarks

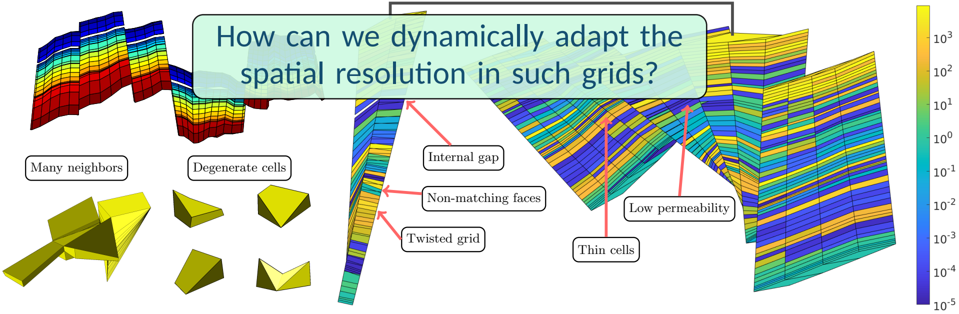
Reservoir simulation grids

- Subsurface reservoirs are complex: layers, faults, fractures, erosion, wells, ...
- Simulation models often upscaled $\rightarrow$ polyhedral cells with full-tensor permeability



Reservoir simulation grids

- Subsurface reservoirs are complex: layers, faults, fractures, erosion, wells, ...
- Simulation models often upscaled $\rightarrow$ polyhedral cells with full-tensor permeability



Dynamic coarsening

- Transport of geothermal heat chiefly confined to proximity of wells
- Difficult to determine appropriate grid resolution a priori
- Many geomodels are not suitable for conventional grid refinement methods

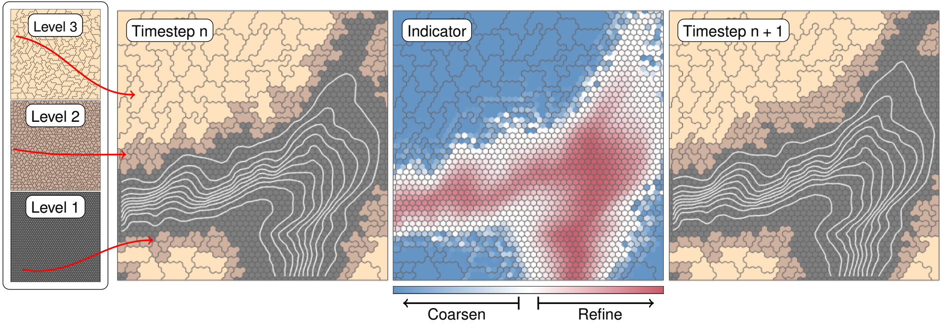
Dynamic coarsening

- Transport of geothermal heat chiefly confined to proximity of wells
- Difficult to determine appropriate grid resolution a priori
- Many geomodels are not suitable for conventional grid refinement methods
- Reservoir engineering applications:

$$\text{\# cells in simulation grid} \quad \ll \quad \text{\# cells in geocellular model}$$

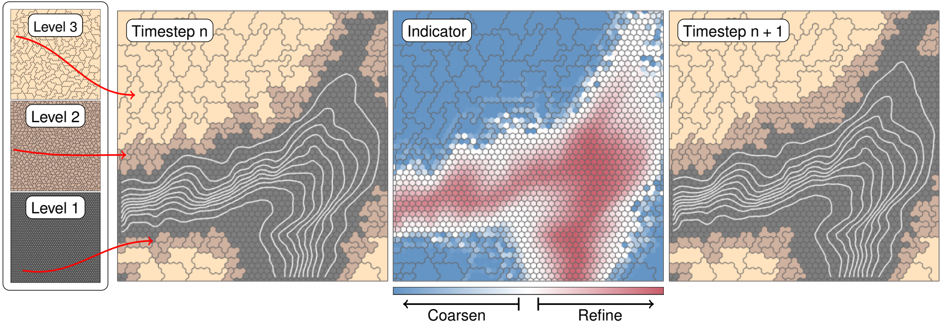
- State-of-the-art multiscale methods (attempt to) bridge gap for pressure problems (Jenny, S. H. Lee, and Tchelepi 2006; Møyner and Lie 2016; Lie et al. 2017, etc.)
- Here: attempt to bridge this gap for transport problems by **dynamic coarsening**

Dynamic coarsening



Quandalle and Besset 1983; Christensen et al. 2004; Batenburg et al. 2011; Hoteit and Chawathe 2016; Cusini and Hajibeygi 2018; Ø. S. Klemetsdal and Lie 2020; Ø. S. Klemetsdal, Møyner, et al. 2021

Dynamic coarsening



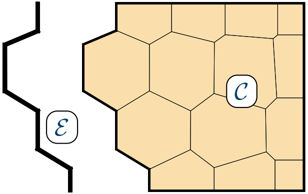
Keep track of which cells to refine/coarsen using coarsening indicator $\mathcal{I}(u) \in \mathbb{R}_{+}^{N}$

Coarse block comprising fine-scale cells $\mathcal{C}$

coarsen if $\mathcal{I}_i < \epsilon_c$ for **all** $i \in \mathcal{C}$, **refine** if $\mathcal{I}_i > \epsilon_r$ for **any** $i \in \mathcal{C}$

Dynamic coarsening – Mapping quantities

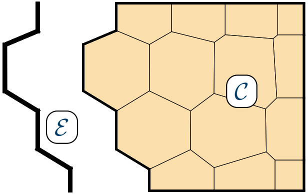
Mapping should be inexpensive and **energy conservative**



Dynamic coarsening – Mapping quantities

Mapping should be inexpensive and **energy conservative**

1. Accumulate coarse-block energy: $\sum_{i\in\mathcal{C}}(\boldsymbol{M}_f\boldsymbol{u}_f + \boldsymbol{M}_r\boldsymbol{u}_r)_i$

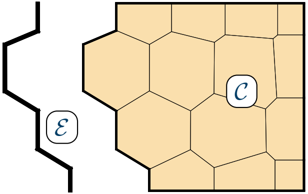


Dynamic coarsening – Mapping quantities

Mapping should be inexpensive and **energy conservative**

1. Accumulate coarse-block energy: $\sum_{i\in\mathcal{C}}(\boldsymbol{M}_f\boldsymbol{u}_f + \boldsymbol{M}_r\boldsymbol{u}_r)_i$
2. PV averaged pressures/temperatures, summed total fluxes

$$\boldsymbol{p}^a = \frac{1}{\boldsymbol{\Phi}^a}\sum_{j\in\mathcal{C}}(\boldsymbol{\Phi}\boldsymbol{p})_j, \quad \boldsymbol{T}^a = \frac{1}{\boldsymbol{\Phi}^a}\sum_{j\in\mathcal{C}}(\boldsymbol{\Phi}\boldsymbol{T})_j, \quad \boldsymbol{v}^a = \sum_{(m,n)\in\mathcal{E}}\boldsymbol{v}_{mn}$$

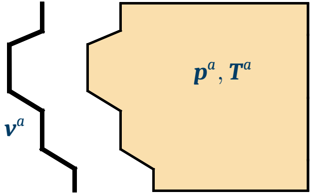


Dynamic coarsening – Mapping quantities

Mapping should be inexpensive and **energy conservative**

1. Accumulate coarse-block energy: $\sum_{i \in \mathcal{C}} (\boldsymbol{M}_f \boldsymbol{u}_f + \boldsymbol{M}_r \boldsymbol{u}_r)_i$
2. PV averaged pressures/temperatures, summed total fluxes

$$\boldsymbol{p}^a = \frac{1}{\boldsymbol{\Phi}^a} \sum_{j \in \mathcal{C}} (\boldsymbol{\Phi} \boldsymbol{p})_j, \quad \boldsymbol{T}^a = \frac{1}{\boldsymbol{\Phi}^a} \sum_{j \in \mathcal{C}} (\boldsymbol{\Phi} \boldsymbol{T})_j, \quad \boldsymbol{v}^a = \sum_{(m,n) \in \mathcal{E}} \boldsymbol{v}_{mn}$$



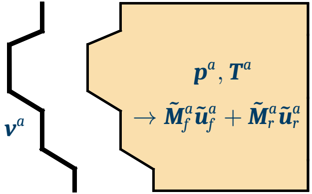
Dynamic coarsening – Mapping quantities

Mapping should be inexpensive and **energy conservative**

1. Accumulate coarse-block energy: $\sum_{i \in \mathcal{C}} (\boldsymbol{M}_f \boldsymbol{u}_f + \boldsymbol{M}_r \boldsymbol{u}_r)_i$
2. PV averaged pressures/temperatures, summed total fluxes

$$\boldsymbol{p}^a = \frac{1}{\boldsymbol{\Phi}^a} \sum_{j \in \mathcal{C}} (\boldsymbol{\Phi} \boldsymbol{p})_j, \quad \boldsymbol{T}^a = \frac{1}{\boldsymbol{\Phi}^a} \sum_{j \in \mathcal{C}} (\boldsymbol{\Phi} \boldsymbol{T})_j, \quad \boldsymbol{v}^a = \sum_{(m,n) \in \mathcal{E}} \boldsymbol{v}_{mn}$$

3. Compute energy in coarse block with adapted properties

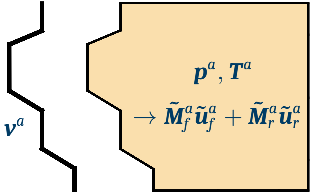


Dynamic coarsening – Mapping quantities

Mapping should be inexpensive and **energy conservative**

1. Accumulate coarse-block energy: $\sum_{i \in \mathcal{C}} (\boldsymbol{M}_f \boldsymbol{u}_f + \boldsymbol{M}_r \boldsymbol{u}_r)_i$
2. PV averaged pressures/temperatures, summed total fluxes

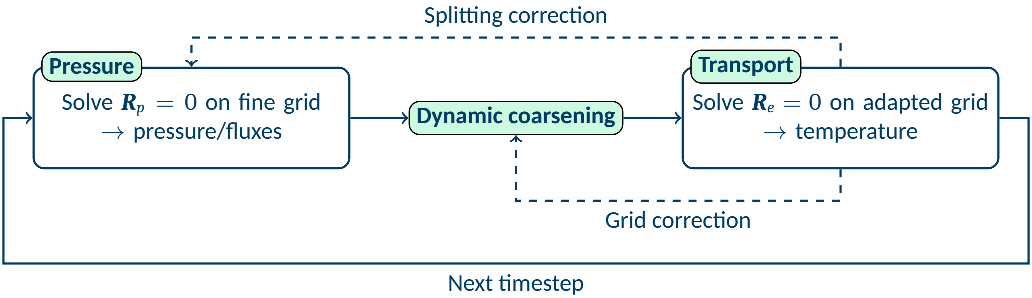
$$\boldsymbol{p}^a = \frac{1}{\boldsymbol{\Phi}^a} \sum_{j \in \mathcal{C}} (\boldsymbol{\Phi} \boldsymbol{p})_j, \quad \boldsymbol{T}^a = \frac{1}{\boldsymbol{\Phi}^a} \sum_{j \in \mathcal{C}} (\boldsymbol{\Phi} \boldsymbol{T})_j, \quad \boldsymbol{v}^a = \sum_{(m,n) \in \mathcal{E}} \boldsymbol{v}_{mn}$$



3. Compute energy in coarse block with adapted properties
4. Ensure conservation of energy through **energy discrepancy** $\rightarrow$ density correction

$$\boldsymbol{S}_t = \frac{\sum_{i \in \mathcal{C}} (\boldsymbol{M}_f \boldsymbol{u}_f + \boldsymbol{M}_r \boldsymbol{u}_r)_i}{\tilde{\boldsymbol{M}}_f^a \tilde{\boldsymbol{u}}_f^a + \tilde{\boldsymbol{M}}_r^a \tilde{\boldsymbol{u}}_r^a} = \frac{\text{accumulated energy from fine grid}}{\text{energy on adapted grid}}$$

Dynamic coarsening – Solution procedure



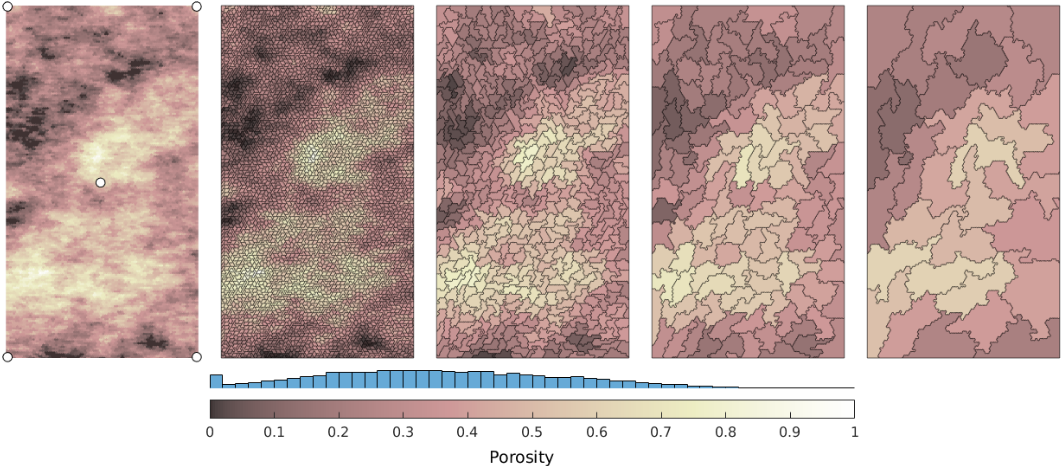
Presentation outline

- Motivation
- Governing equations and discretization
- The MATLAB Reservoir Simulation Toolbox
- Dynamic coarsening
- Numerical examples
- Concluding remarks

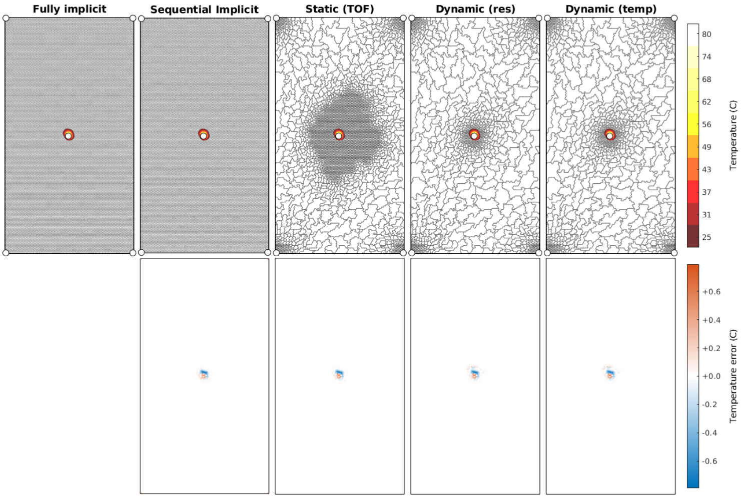
Example: SPE10 Model 2

- Heat storage in two different layers of SPE10 Model 2
- Three one-year cycles of storage in center well with pressure support in corner wells
    1. **Charge:** 3 months of injection at 80 °C, bhp = 70 bar
    2. **Rest:** 3 months with no driving forces
    3. **Discharge:** 3 months of extraction, bhp = 1500 bar
    4. **Rest:** 3 months with no driving forces
- Three coarsening approaches
    1. Static based on incompressible time-of-flight
    2. Dynamic with residual-based indicator
    3. Dynamic with temperature indicator

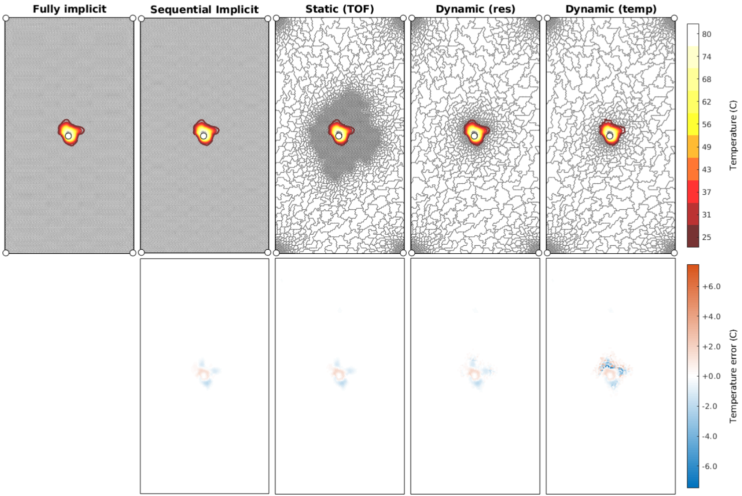
Example: SPE10 Model 2 – Tarbert (layer 10)



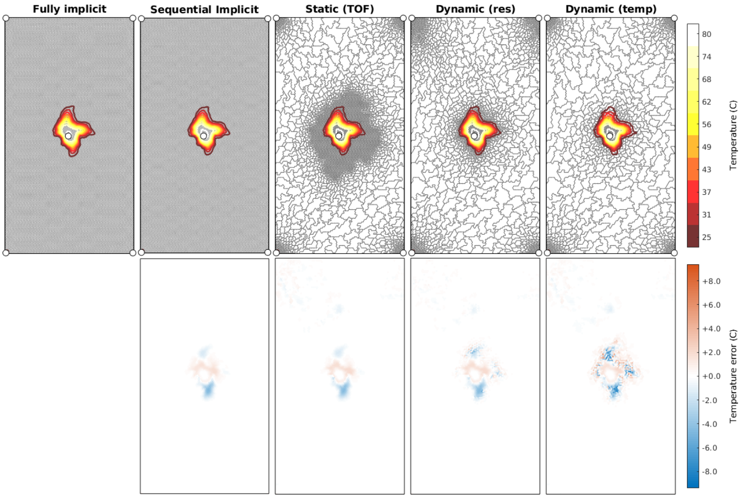
Example: SPE10 Model 2 – Tarbert (layer 10)



Example: SPE10 Model 2 – Tarbert (layer 10)

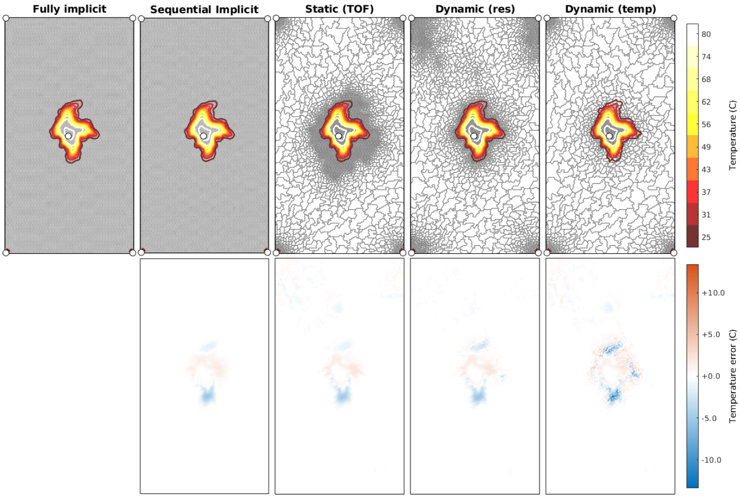


Example: SPE10 Model 2 – Tarbert (layer 10)

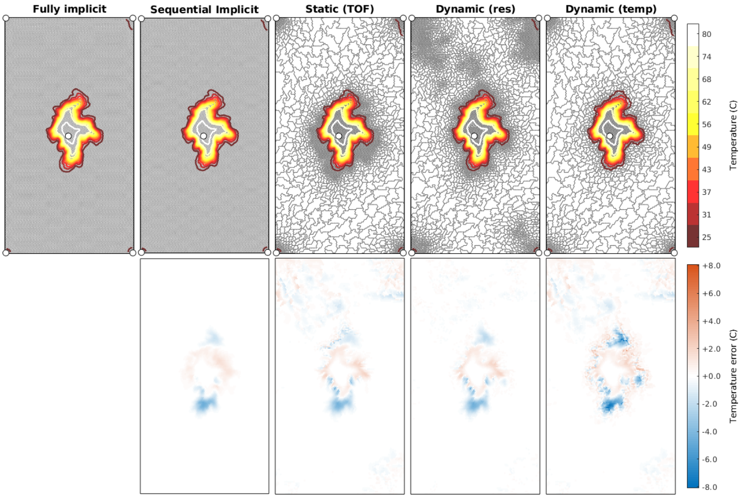


SINTEF

Example: SPE10 Model 2 – Tarbert (layer 10)

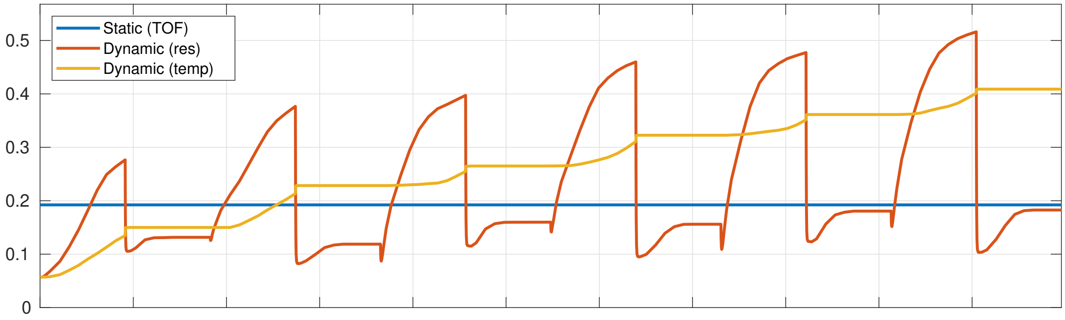


Example: SPE10 Model 2 – Tarbert (layer 10)



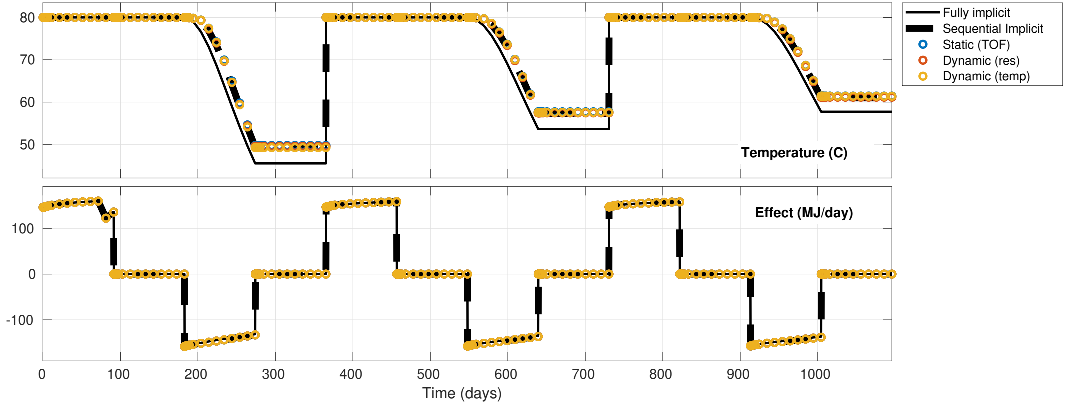
Example: SPE10 Model 2 – Tarbert (layer 10)

Dynamic grid relative cell count

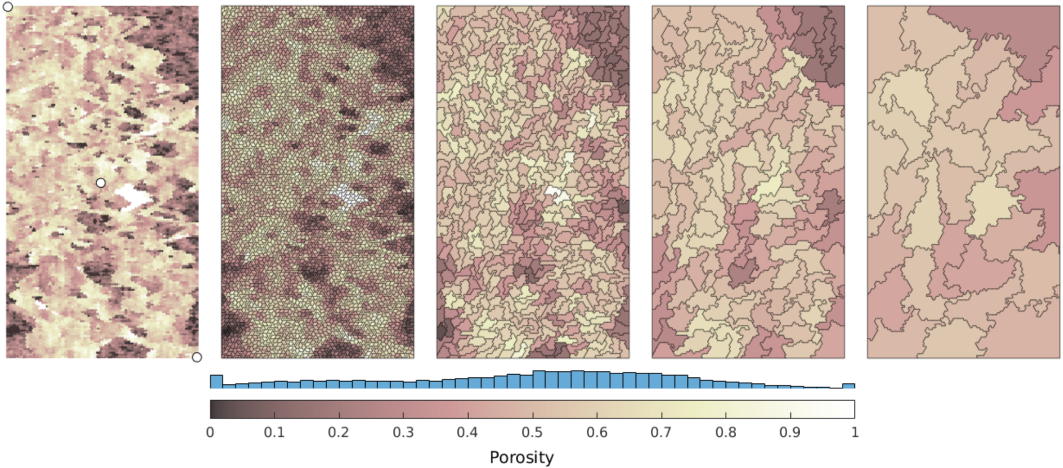


Example: SPE10 Model 2 – Tarbert (layer 10)

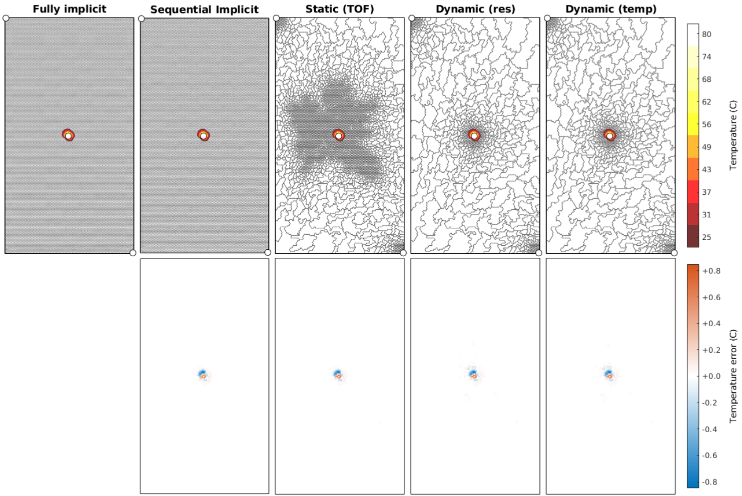
Injection well output



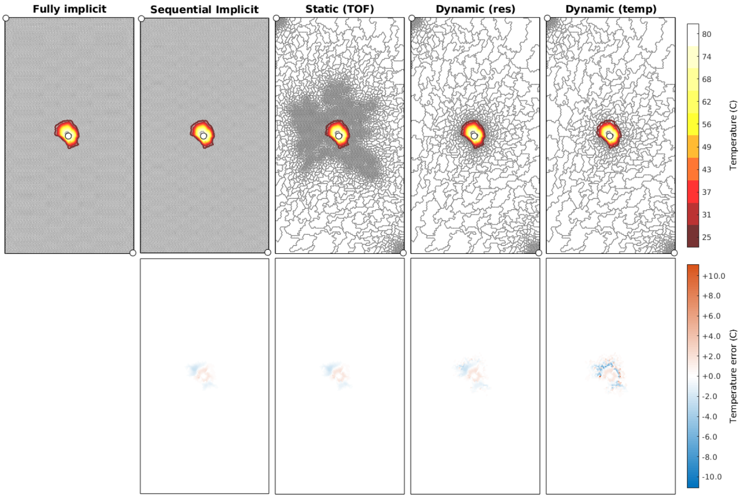
Example: SPE10 Model 2 – Upper Ness (layer 85)



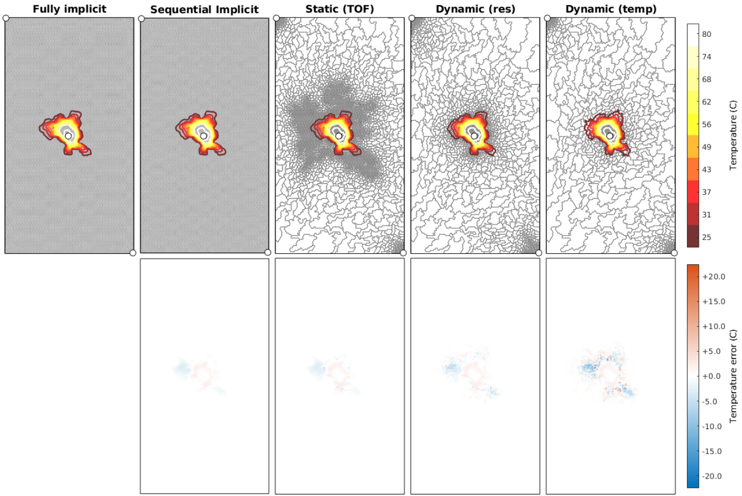
Example: SPE10 Model 2 – Upper Ness (layer 85)



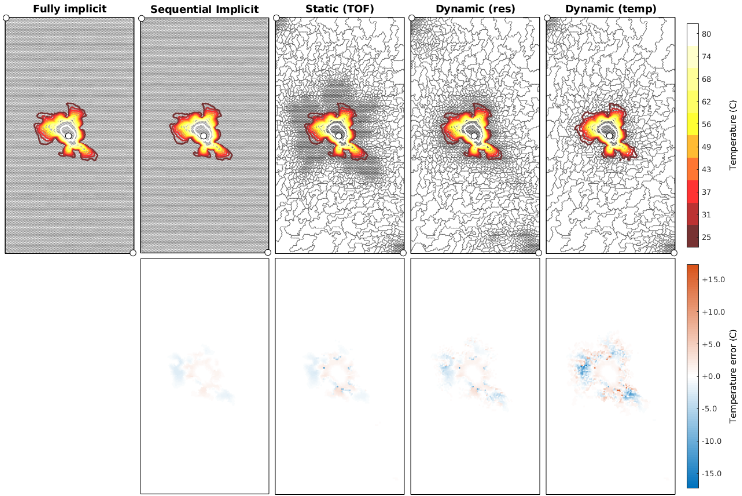
Example: SPE10 Model 2 – Upper Ness (layer 85)



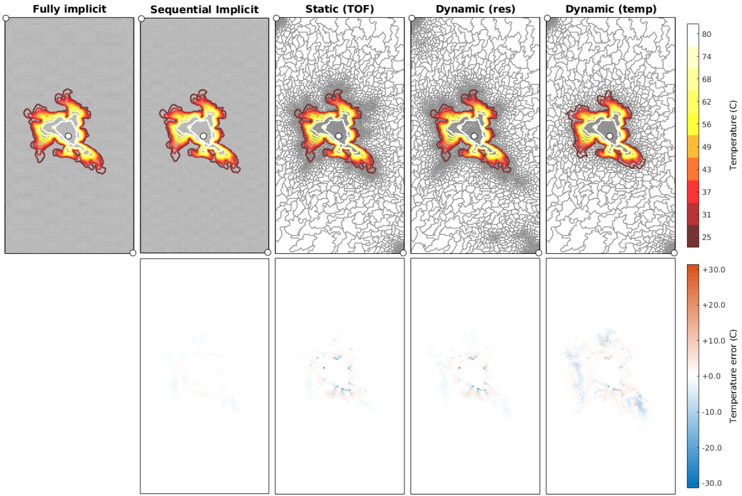
Example: SPE10 Model 2 – Upper Ness (layer 85)



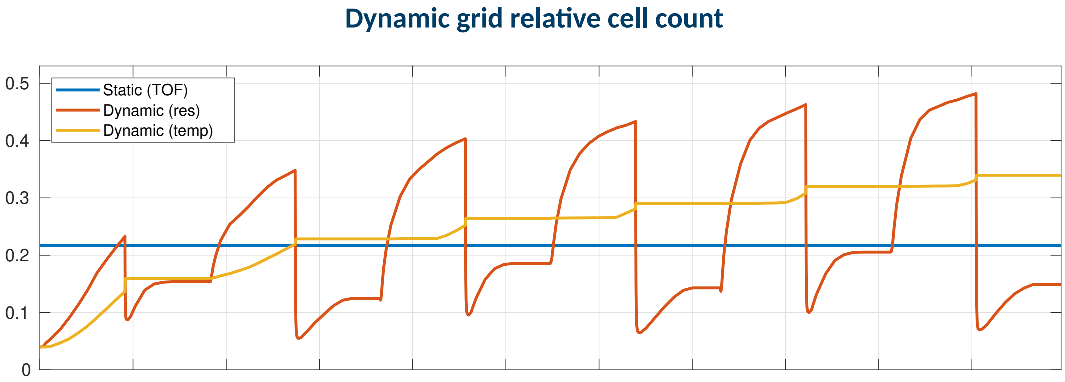
Example: SPE10 Model 2 – Upper Ness (layer 85)



Example: SPE10 Model 2 – Upper Ness (layer 85)

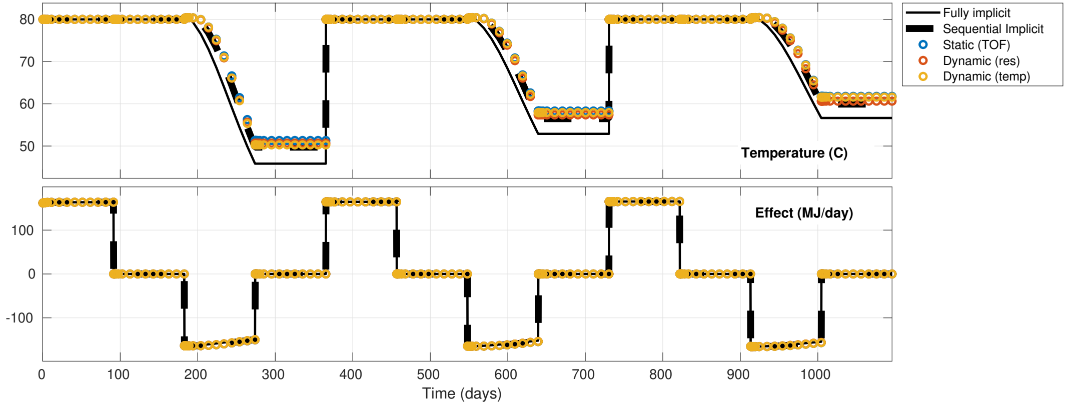


Example: SPE10 Model 2 – Upper Ness (layer 85)



Example: SPE10 Model 2 – Upper Ness (layer 85)

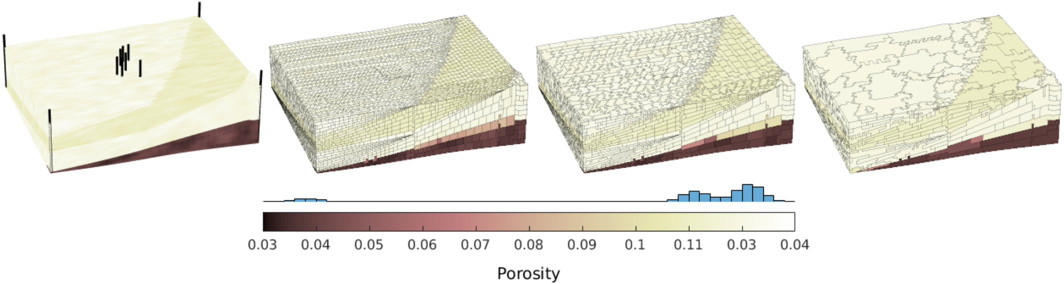
Injection well output



Example: SPE10 Model 2

- Very close match with fine-scale results for all indicators and coarsening strategies
  - Between 49% and 96% reduction in # transport problem dofs
- Point-wise large temperature differences
- Energy discrepancy correction ensures conservation of energy between scales

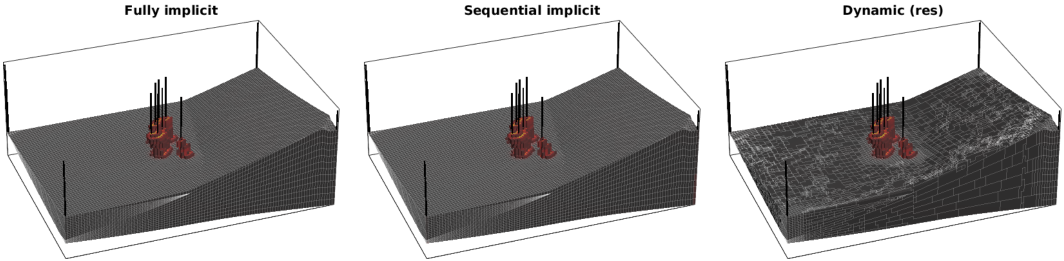
Example: Real(istic) Model



- Model of real geothermal storage site, provided by Ruden AS
- Corner-point grid with four geological layers
- Group of center wells inject at 73 °C over four months, pressure support in corner wells
- Dynamic coarsening with residual-based indicator

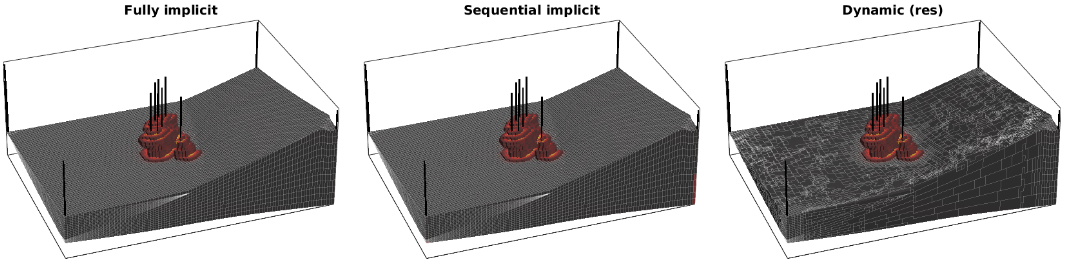
Example: Real(istic) Model

Reservoir temperature



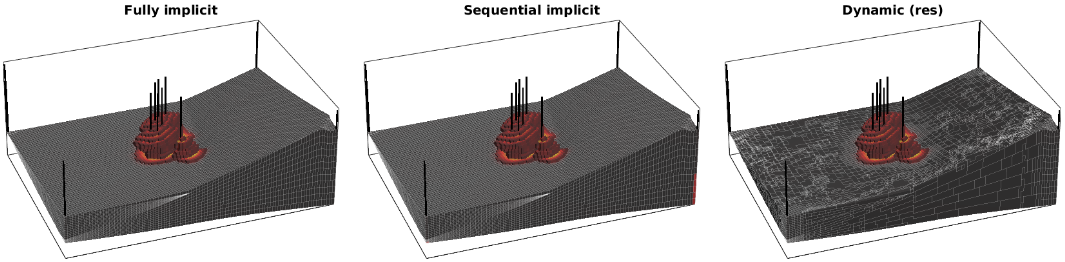
Example: Real(istic) Model

Reservoir temperature



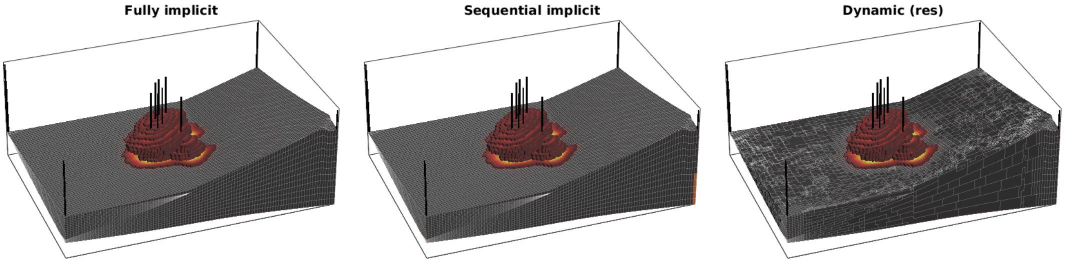
Example: Real(istic) Model

Reservoir temperature



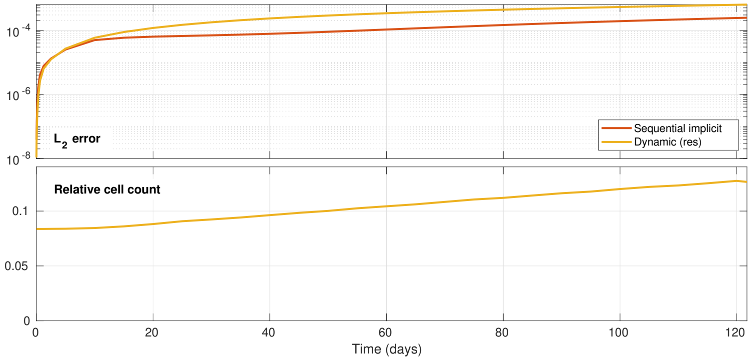
Example: Real(istic) Model

Reservoir temperature



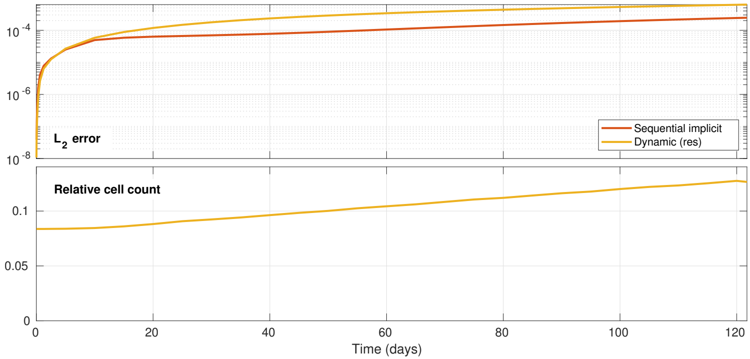
Example: Real(istic) Model

Relative L_2 energy difference from fully implicit and dynamic grid relative cell count



Example: Real(istic) Model

Relative L_2 energy difference from fully implicit and dynamic grid relative cell count



Less than 10^{-3} maximum relative L_2 difference with at least **87% reduction** in # transport problem dofs

Presentation outline

- Motivation
- Governing equations and discretization
- The MATLAB Reservoir Simulation Toolbox
- Dynamic coarsening
- Numerical examples
- Concluding remarks

Concluding Remarks

Conclusions

- Highly flexible **dynamic coarsening** method for geothermal simulations in MRST
  - Sequential splitting of flow and transport/energy
  - Applicable to unstructured, polytopal grids
  - Energy discrepancy correction ensures **conservation of energy**
- Method demonstrated on two examples (low/moderate enthalpy)
  - **Significant reduction** in # dofs in the transport subproblem
  - Very good match with fine-scale solution

Concluding Remarks

Further work

- Optimize implementation and investigate actual CPU speedup
- Test method for high-enthalpy systems (phase changes)
- Solve each subproblem at its appropriate timescale
  - Multiple transport steps for each pressure step
- Combine with a posteriori estimators for error control (Ahmed et al. 2021)

Concluding Remarks

Related talks

MS83B (16:30) *Using MRST for modeling and optimization of operational strategies for a geothermal storage plant in Asker, Norway*

MS50B (16:30) *Optimized graph-based methods for subsurface flow simulations*

Acknowledgments

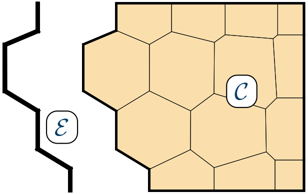
Thanks to Marine Collignon (University of Geneva),
Olav Møyner and Knut-Andreas Lie (SINTEF Digital) for fruitful discussions
Thanks to Ruden AS for allowing use of the field model in this work

SINTEF

Technology for a better society

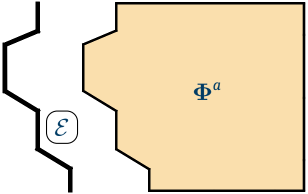
Extra slides

Dynamic coarsening – Mapping parameters



Dynamic coarsening – Mapping parameters

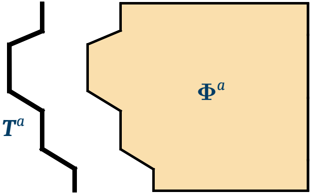
- Accumulate pore volumes: $\mathbf{\Phi}^a = \sum_{i\in\mathcal{C}} \mathbf{\Phi}_i$



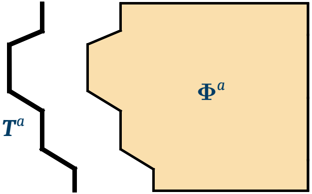
Dynamic coarsening – Mapping parameters

SINTEF

- Accumulate pore volumes: $\mathbf{\Phi}^a = \sum_{i \in \mathcal{C}} \mathbf{\Phi}_i$
- Compute transmissibilities (multiple options):
  1. Accumulation: $\boldsymbol{T}^a = \sum_{(m,n) \in \mathcal{E}} \boldsymbol{T}_{mn}$
  2. Upscale permeability and coarse geometry $\rightarrow$ compute $\boldsymbol{T}^a$
  3. Compute representative transmissibility given flux $\boldsymbol{T}^a_{mn} = \boldsymbol{v}_{mn} / (\boldsymbol{p}_m - \boldsymbol{p}_n)$



Dynamic coarsening – Mapping parameters



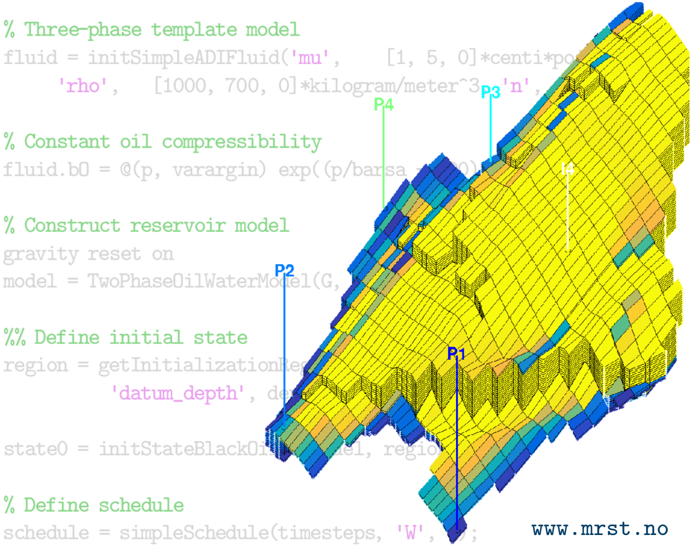
- Accumulate pore volumes: $\Phi^a = \sum_{i \in \mathcal{C}} \Phi_i$
- Compute transmissibilities (multiple options):
  1. Accumulation: $T^a = \sum_{(m,n) \in \mathcal{E}} T_{mn}$
  2. Upscale permeability and coarse geometry $\rightarrow$ compute T^a
  3. Compute representative transmissibility given flux $T^a_{mn} = v_{mn}/(p_m - p_n)$
- These parameters can be computed in a preprocessing step (except transmissibility option 3)
  $\rightarrow$ Adapting the grid amounts to looking up precomputed parameters

MATLAB Reservoir Simulation Toolbox (MRST)

Transforming research on reservoir modelling

Unique prototyping platform:

- Standard data formats
- Data structures/library routines
- Fully unstructured grids
- Rapid prototyping:
  - Differentiation operators
  - Automatic differentiation
  - Object-oriented framework
  - State functions
- Industry-standard simulation



www.mrst.no

MATLAB Reservoir Simulation Toolbox (MRST)

Transforming research on reservoir modelling

Unique prototyping platform:

- Standard data formats
- Data structures/library routines
- Fully unstructured grids
- Rapid prototyping:
  - **Differentiation operators**
  - Automatic differentiation
  - Object-oriented framework
  - State functions
- Industry-standard simulation

Differentiation operators

Writing discrete equations on form very close to continuous equations

$\nabla \cdot \vec{H}$ $\qquad$ $\vec{H} = -(\lambda_f + \lambda_r)\nabla T$

```
div(H)      H = -(lambdaF + lambdaR).*grad(T)
```

MATLAB Reservoir Simulation Toolbox (MRST)

Transforming research on reservoir modelling

Unique prototyping platform:

- Standard data formats
- Data structures/library routines
- Fully unstructured grids
- Rapid prototyping:
  - Differentiation operators
  - **Automatic differentiation**
  - Object-oriented framework
  - State functions
- Industry-standard simulation

Differentiation operators

Writing discrete equations on form very close to continuous equations

$\nabla \cdot \vec{H}$	$\vec{H} = -(\lambda_f + \lambda_r)\nabla T$
`div(H)`	`H = -(lambdaF + lambdaR).*grad(T)`

Automatic differentiation

Combine chain rule and elementary differentiation rules by means of operator overloading to analytically evaluate all derivatives

$\rightarrow$ *Computing Jacobians amounts to writing down residual equations.*

```
[x,y] = initVariablesADI(1,2); z = 3*exp(-x*y)
```

```
x = ADI Properties:        y = ADI Properties:        z = ADI Properties:
    val: 1                     val: 2                     val: 0.4060
    jac: {[1]  [0]}            jac: {[0]  [1]}            jac: {[-0.8120]  [-0.4060]}
```

$\frac{\partial x}{\partial x}$ $\frac{\partial x}{\partial y}$ $\frac{\partial y}{\partial x}$ $\frac{\partial y}{\partial y}$ $\left.\frac{\partial z}{\partial x}\right|_{x=1,y=2}$ $\left.\frac{\partial z}{\partial y}\right|_{x=1,y=2}$

MATLAB Reservoir Simulation Toolbox (MRST)

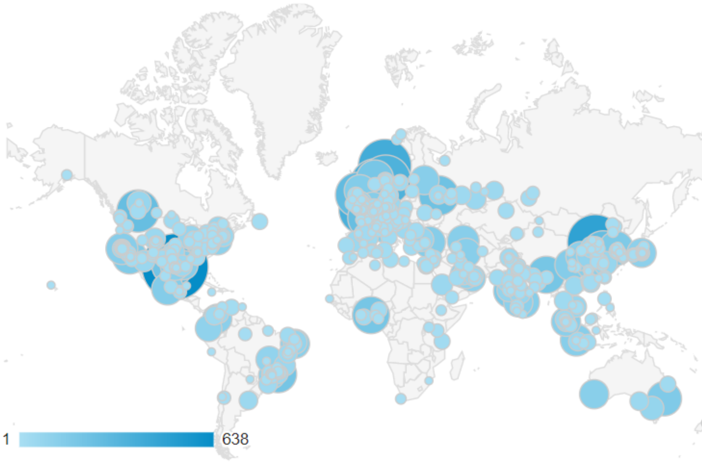
Transforming research on reservoir modelling

Large international user base:

- downloads from the whole world
- 124 master theses
- 56 PhD theses
- 400 journal papers (not by us)
- 144 proceedings papers

Numbers are from Google Scholar notifications

Used both by academia and industry



Google Analytics: access pattern for `www.mrst.no`
Period: 1 July 2018 to 31 December 2019
Unique downloads: 5 516 (103 countries and 838 cities)