Field-Scale Petroleum Reservoir Simulation

Chensong Zhang

NCMIS & LSEC, Chinese Academy of Sciences
University of Chinese Academy of Sciences

China University of Petroleum, Qingdao — June 28, 2017

== version-2017.06.28.02 ==

Outline

中国科学院

CHINESE ACADEMY OF SCIENCES

Our Team and Collaborators

- Penn State Univ: Jinchao Xu, Changhe Qiao, Hongxuan Zhang, ...
- LSEC, CAS: Chensong Zhang, Ronghong Fan, ...
- Xiangtan Univ: Shi Shu, Chunsheng Feng, Xiaoqiang Yue, Zhiyang Zhou, ...
- Kunming Univ of Sci & Tech: Zheng Li
- Tufts Univ: Xiaozhe Hu
- Sichuan Univ: Shiquan Zhang
- SYSU, Guangzhou: Yuesheng Xu, Wenchao Guan, Yongdong Zhang, ...
- CoCreative Center: David Zhu, Zhongjian Zhang, ...
- RIPED, PetroChina: Shuhong Wu, Baohua Wang, Guanren Huan, ...
- CNOOC: Xiansong Zhang, Chunyang Lin, ...
- Monix Energy & China Univ of Petroleum: Wei Liu, ...
- China Univ of Petroleum: Ruizhong Jiang, ...

Financial support from CNOOC-2010-ZHKY-ZX-008, PetroChina-12HT10500002654, and CAS-QYZDB-SSW-SYS018!

§1. Modeling Aspects

1 Petroleum Reservoir Simulation

2 Phase Behavior: Black Oil

3 Classical Black Oil Model

4 Phase Behavior: Compositional

5 General Compositional Model

6 Enhanced Oil Recovery

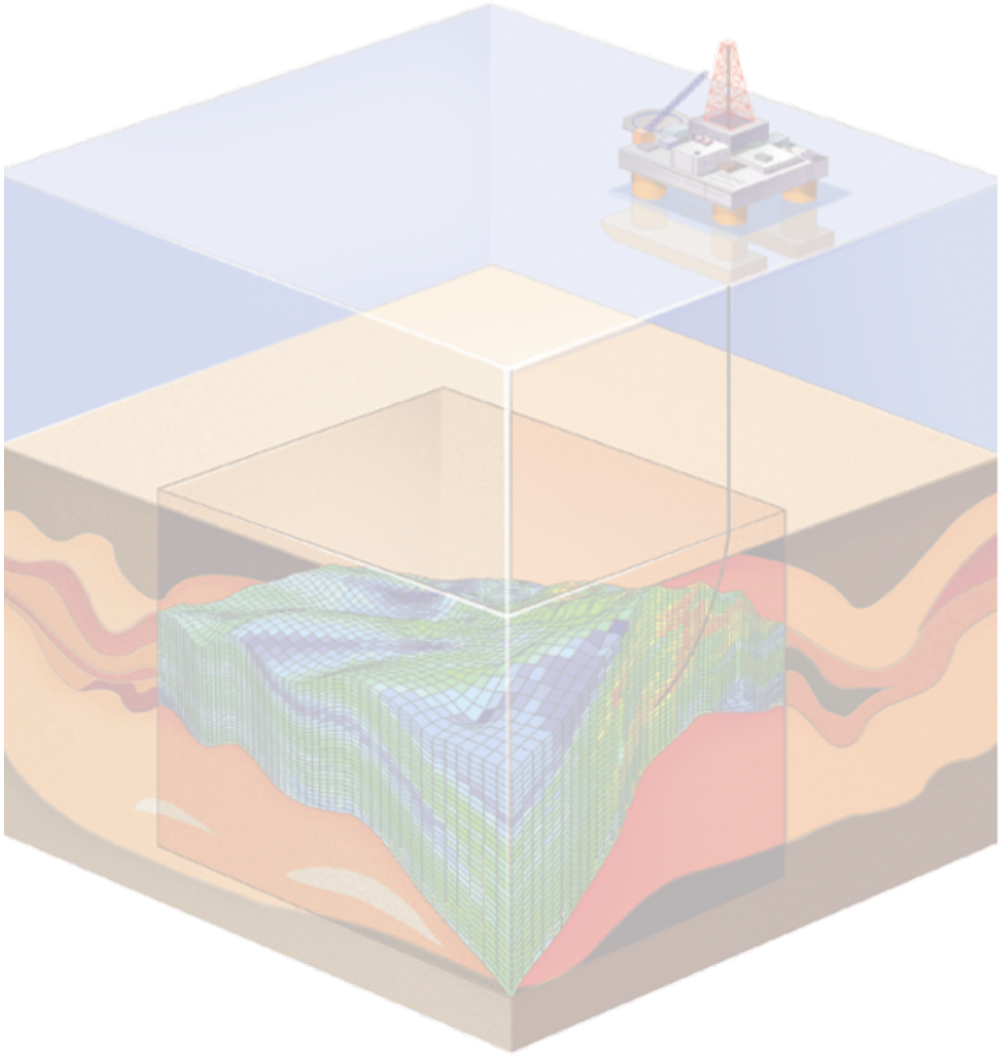
7 Unconventional Oil/Gas

8 Models of Interest

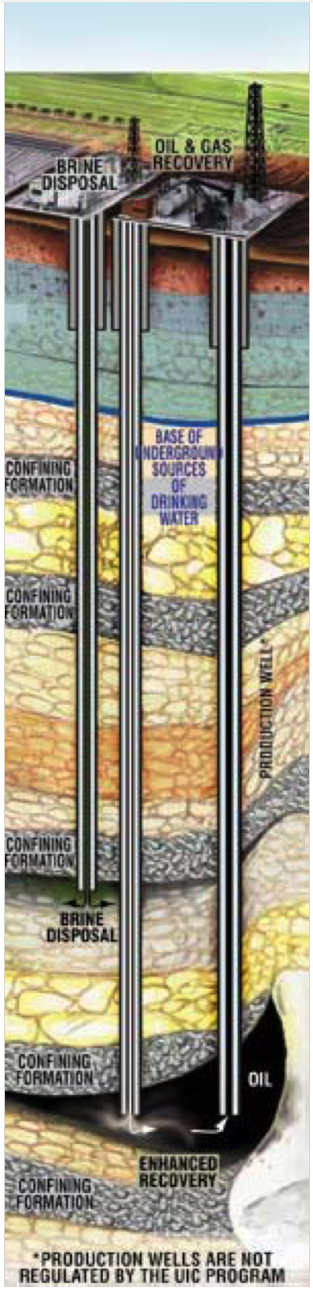
9 Modeling Complex Wells

10 Quantifying Uncertainty

11 Forward Uncertainty Propagation



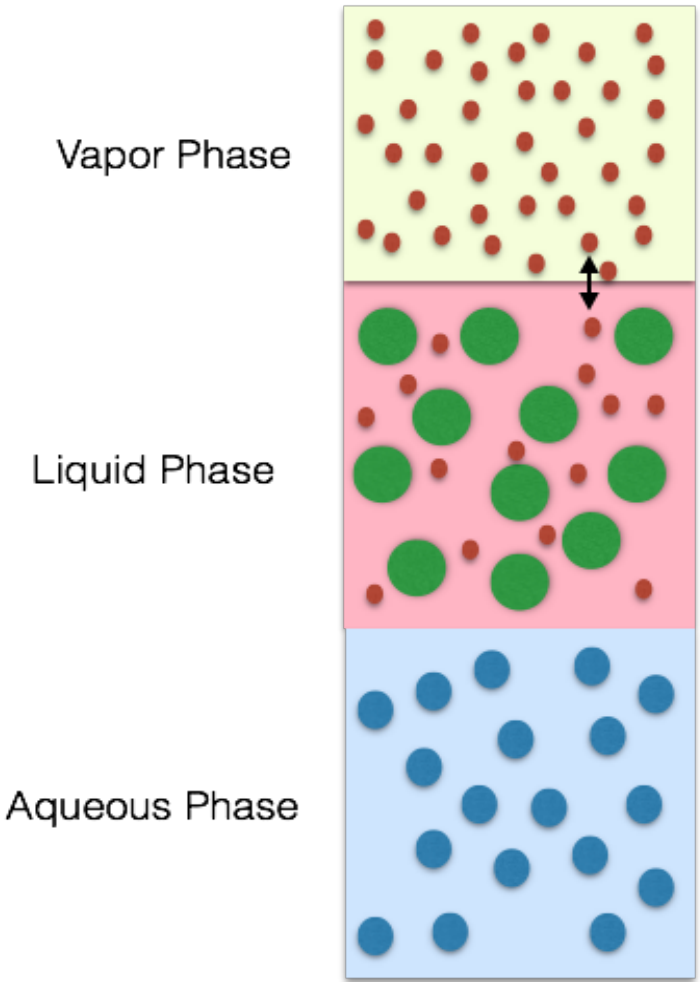
Petroleum Reservoir Simulation



Upstream Oil Industry: Finding and developing hydrocarbon deposits

1. Finding nearly horizontal and major fault surfaces
2. Determining detailed stratigraphic layers, faults, pinch-outs, ...
3. Generating reservoir characterization model ($10^6 \sim 10^8$ cells)
4. Describing reservoir heterogeneity at multiple scales
5. Upscaling reservoir grids and properties ($10^4 \sim 10^6$ cells)
6. Finding fluid properties: PVT, relative permeability, ...
7. Reservoir initialization
8. Dynamic simulation (developing process)
9. History matching
10. Calibrating model parameters

Phase Behavior: Black Oil



Black oil fluid

- The black oil model is based on simple interpolation of PVT properties as a function of pressure
- Water is modeled explicitly together with two hydrocarbon components, an oil phase and a gas phase
- At standard pressure and temperature, hydrocarbon components are divided into a gas component and an oil component in a stock tank
- No mass transfer occurs between the water phase and the oil/gas phases

[Chen, Huan, Ma, 2006]

Classical Black Oil Model

1. Mass conservation (saturated & under-saturated):

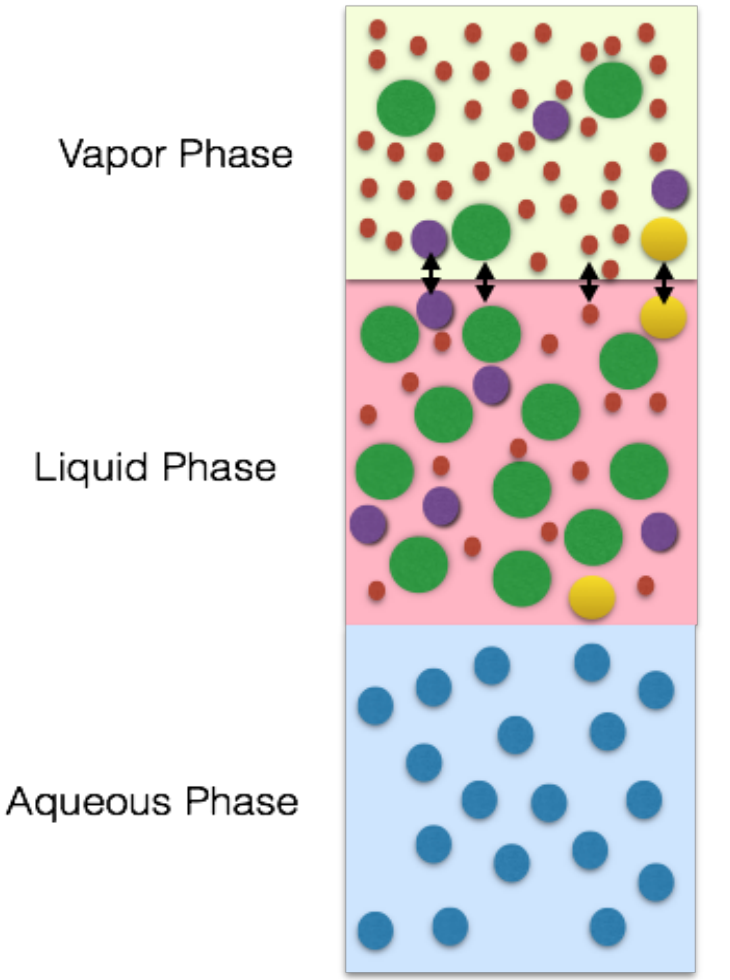
$$\frac{\partial}{\partial t}\left(\phi\rho_w S_w\right) = -\nabla\cdot\left(\rho_w \mathbf{u}_w\right) + Q_W$$
$$\frac{\partial}{\partial t}\left(\phi\rho_{oO} S_o\right) = -\nabla\cdot\left(\rho_{oO} \mathbf{u}_o\right) + Q_O$$
$$\frac{\partial}{\partial t}\left(\phi\rho_g S_g + \phi\rho_{oG} S_o\right) = -\nabla\cdot\left(\rho_g \mathbf{u}_g + \rho_{oG}\mathbf{u}_o\right) + Q_G$$

2. Darcy's Law and other constitutive equations:

$$\mathbf{u}_j = -\frac{k k_{rj}}{\mu_j}(\nabla P_j - \rho_j \mathfrak{g}\nabla z), \quad j = o, g, w$$
$$S_o + S_g + S_w = 1$$
$$P_{cow} = P_o - P_w, \quad P_{cog} = P_g - P_o$$

3. Well constraints + B.C. + I.C.

Phase Behavior: Compositional



EOS compositional fluid

- In reservoirs containing light oil, the hydrocarbon composition affects fluid properties a lot
- A compositional model is based on a thermodynamically-consistent model such as an equation of state (EOS)
- Each hydrocarbon component (arbitrary number) is handled separately
- More unknowns than the black oil model: ξ_j is the molar density of phase j; x_{ij} is the molar fraction of comp i in phase j; N_i is the overall molar density of comp i

[Chen, Huan, Ma, 2006]

General Compositional Model

$$\frac{\partial}{\partial t}\left(\phi \sum_{j=1}^{n_p} x_{ij}\xi_j S_j\right) + \nabla \cdot \mathbf{F}_i - \sum_{j=1}^{n_p} S_j r_{ij} = Q_i, \qquad i = 1 : n_c$$

$$\mathbf{F}_i = \sum_{j=1}^{n_p} \left(x_{ij}\xi_j \mathbf{u}_j - S_j \mathbf{D}_j \nabla(\xi_j x_{ij})\right), \qquad i = 1 : n_c$$

$$\mathbf{u}_j = -\frac{k k_{rj}}{\mu_j}(\nabla P_j - \gamma_j \nabla z), \qquad j = 1 : n_p$$

$$\sum_{j=1}^{n_p} S_j = 1,$$

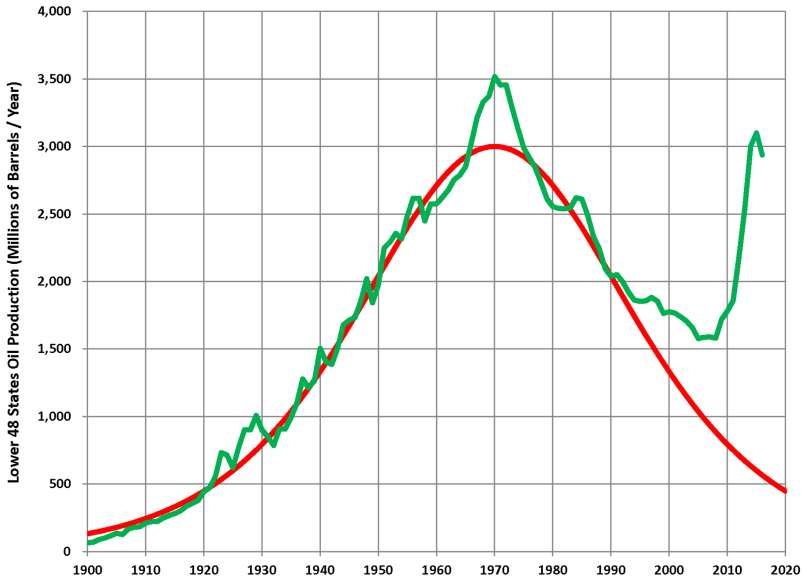
$$\sum_{i=1}^{n_c} x_{ij} = 1, \qquad j = 1 : n_p$$

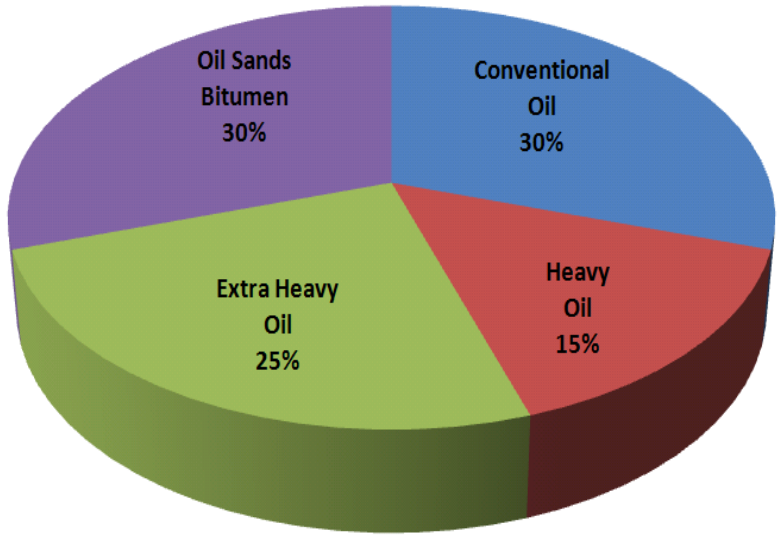
$$f_{ij} = f_{i1}, \qquad i = 1 : n_c,\ j = 2 : n_p$$

[Collins, Nghiem, Li, Grabenstetter, 1992; Qiao, Li, Johns, Xu, 2014, 2015; ...]

Enhanced Oil Recovery

Hubbert peak oil theory, 1956



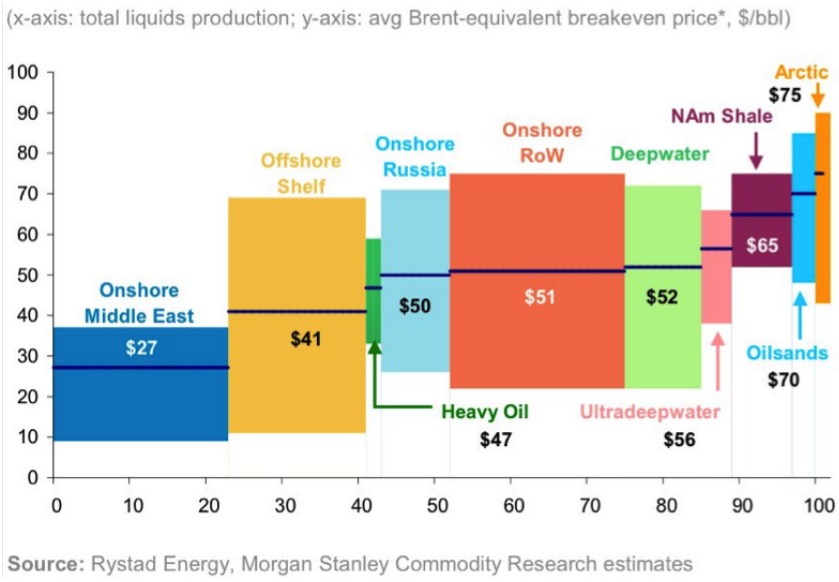


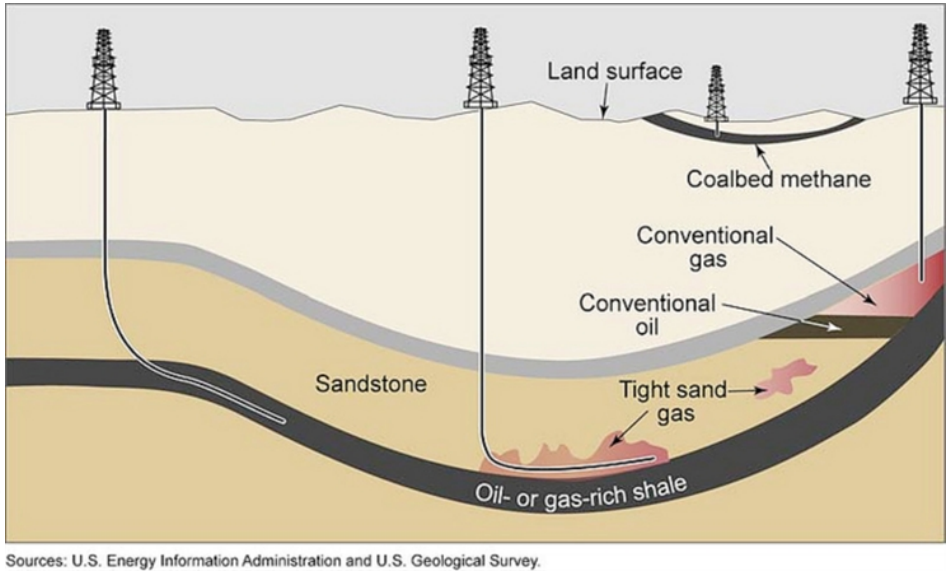
EOR techniques: recovery ratio 20%–40% $\Longrightarrow$ 30%–60%

- ✓ Gas injection: miscible flooding, commonly used
- ☞ Thermal injection: steam, fire, ... $\Longrightarrow$ Energy equation, ...
- ☞ Chemical injection: polymer, surfactant, microbial, ...
- ☞ More complicated well models

Environment Impact: Produce brine with toxic and radioactive substances!

Unconventional Oil/Gas





Unconventional oils: not strictly defined, technologies and economy

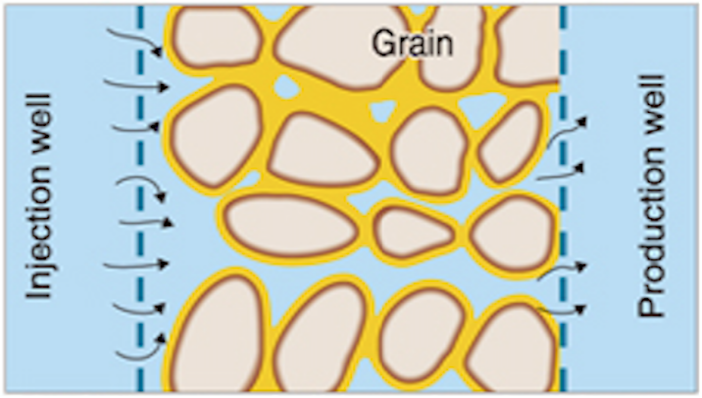
✏ Examples: Tight oil/gas, shale oil/gas, heavy oil, oil sands, gas hydrate

☞ Different models for different development conditions / technologies:

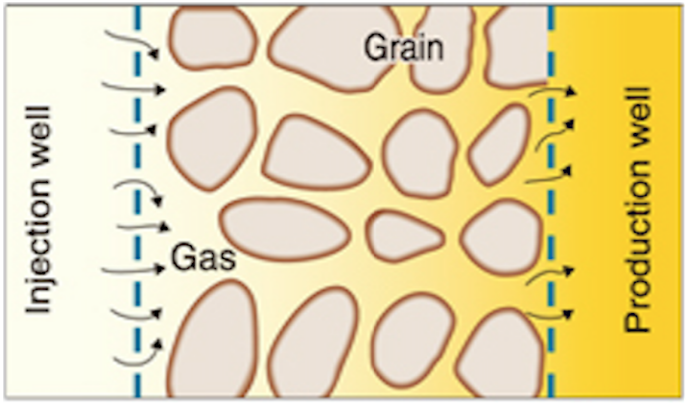
- Non-Darcy and non-Newtonian effects
- Multiscale fractures, adsorption/desorption
- Knudsen diffusion
- Fluid-structure interactions: fractures formation and propagation

[Garipov, Karimi-Fard, Tchelepi, 2016]

Models of Interest



Immiscible displacement, water flooding



Miscible displacement, gas flooding

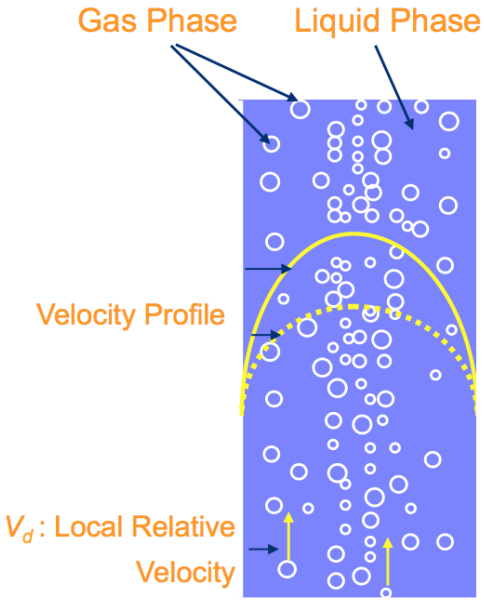
- ✓ Two-phase flow model: WO, OG
- ✓ Black oil model (three-phase flow)
- ✓ Volatile oil model
- ☞ Compositional model
  - Non-isothermal flow: energy conservation
  - Chemical flooding: polymer, foam, surfactant, alkaline, ...
  - Fractured reservoirs: DPDP, MINC, DFM, ...
- ✏ Darcy-NS coupling (Brinkman)
- ✏ Flow-Geomechanic coupling (Biot)

Modeling Complex Wells

Peaceman's well model

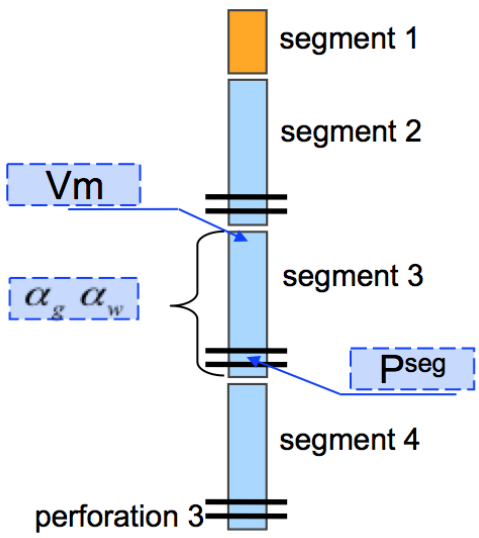
$$Q_j^{(w,p)} = \mathrm{WI}_j^{(w,p)} \left(P_{\mathrm{bh}}^{(w)} - P_j - \rho_j \mathfrak{g}(z_{\mathrm{bh}}^{(w)} - z) \right)$$

- Suitable for vertical wells
- 1D radial flow
- Steady-state
- Single-phase
- Polygonal partition: Palagi's well model



Simulating modern complex wells

- Horizontal wells
- Complex wells



Treatment of complex wells

- Multi-segment well model
- Drift-flux model
- Frictional resistance model

Quantifying Uncertainty

Uncertainty v.s. Error

- Lack of knowledge? Types: aleatoric (statistical) and epistemic (systematic)
- Sources of uncertainty: model, measurements, initial/boundary conditions
- "All models are wrong, but some are useful" [George Box] $\Longrightarrow$ V&V
- Where uncertainty makes a big difference (compared with error)

Uncertainty quantification: SIAM/ASA-joint conference on UQ 2012

- Predict model responses with quantified and reduced uncertainties
  - Identification and characterization
  - ☞ Forward propagation (UP): Surrogate model, MC, GP, PCE, gPCE, KLE, ...
  - Inverse propagation
  - Sensitivity analysis
- Difficulties when applied in petroleum reservoir simulation
  - Curse of dimensionality
  - Identifiability issue: Combinations of uncertainties yield the same prediction
  - "... an uncertain input parameter will lead not only to an uncertain solution but to an uncertain error ..." [Trucano, 2004]

Forward Uncertainty Propagation

Model problem and uncertainty propagation

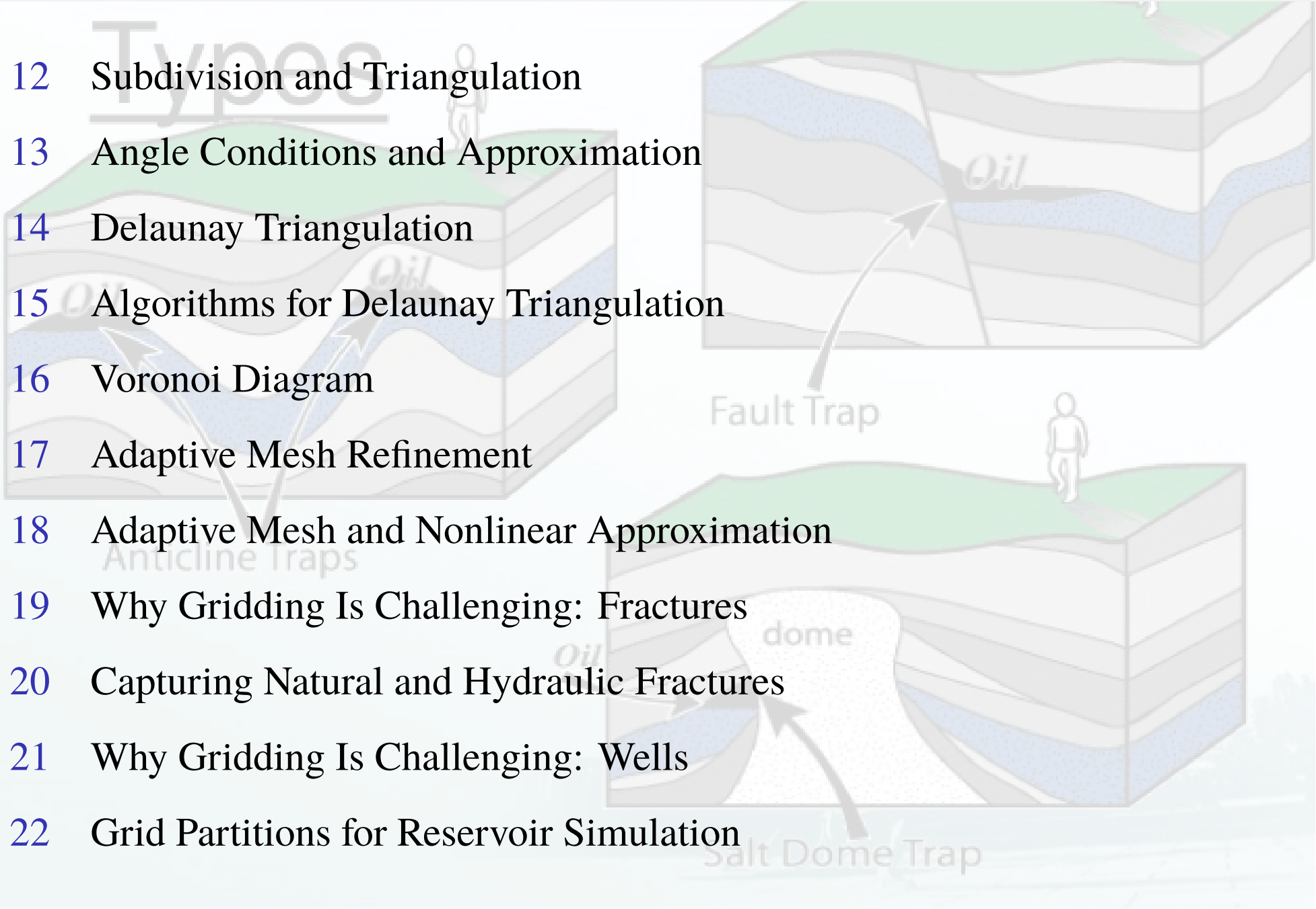
$$y = \mathcal{F}(x) \implies Y = \mathcal{F}(X), \text{ where } X \text{ is a random variable}$$

- Polynomial Chaos Expansion: represent a random variable of interest as a polynomial expansion of another random variable ξ with distribution ρ
- $\{\psi_j\}$ are the orthogonal polynomials w.r.t. $(\cdot,\cdot)_\rho$
  - We have $\psi_0 = 1$ and $E(\psi_j) = 0$, $j = 1, 2, \ldots$
  - Variance of ψ_j is $(\psi_j, \psi_j)_\rho$ and covariance $(\psi_i, \psi_j)_\rho = 0$ if $i \neq j$
  - A few possible choices, for example:
    Uniform $[-1, 1]$ $\Rightarrow$ Legendre; Gamma $[0, \infty)$ $\Rightarrow$ Laguerre; Normal $\Rightarrow$ Hermite

Non-intrusive UP with PCE

- Suppose $X \approx \sum_{j=0}^{m} x_j \psi_j(\xi)$ and $Y \approx \sum_{j=0}^{m} y_j \psi_j(\xi)$
- $\sum_{j=0}^{m} y_j \psi_j(\xi) = \mathcal{F}\big(\sum_{j=0}^{m} x_j \psi_j(\xi)\big) \implies y_k = \dfrac{\big(\mathcal{F}\big(\sum_{j=0}^{m} x_j \psi_j(\xi)\big), \psi_k\big)_\rho}{(\psi_k, \psi_k)_\rho}$
- Need to compute the integral: $\int_\Omega \mathcal{F}(\sum_{j=0}^{m} x_j \psi_j(\xi)) \psi_k(\xi) \rho(\xi)\, d\xi$

§2. Gridding Techniques

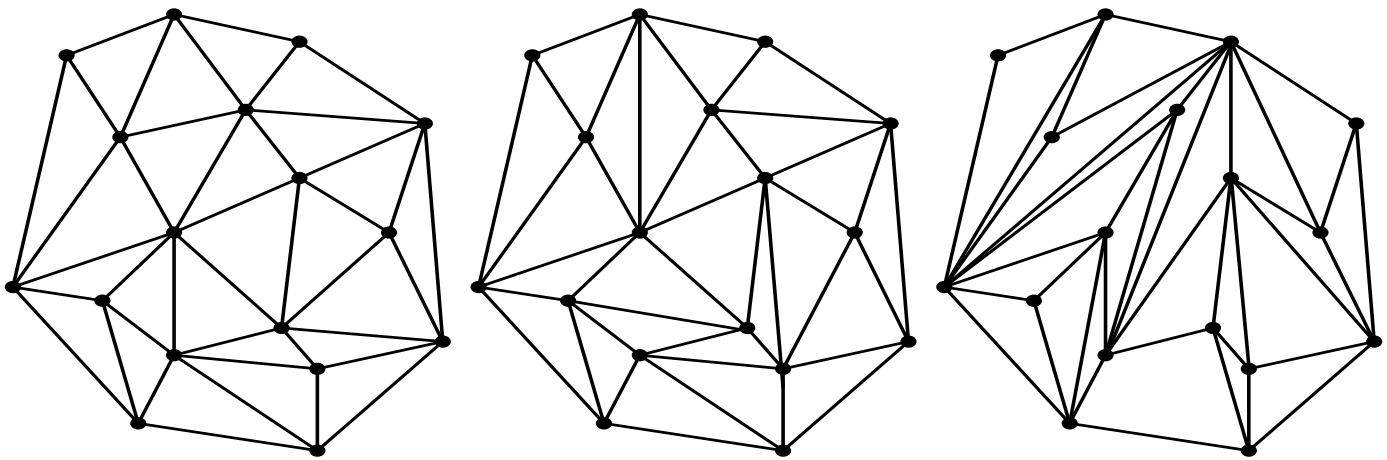


Subdivision and Triangulation

Subdivision and (conforming) triangulation

- Subdivision (partition) of Ω: $\cup_i \tau_i = \overline{\Omega}$ and $\text{int}\,\tau_i \cap \text{int}\,\tau_j = \emptyset$ (if $i \neq j$)
- Triangulation: A subdivision in which no vertex lies in the interior of any edge
- Find a triangulation $\mathcal{T}(\mathbb{P})$ of a set of sites (points) $\mathbb{P} := \{p_1, \ldots, p_n\}$
- An important problem in computational geometry with MANY applications

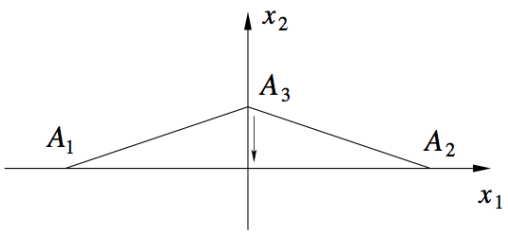
What is a "good" triangulation?



- We need to give mathematical conditions on "good" and "bad"
- We need to give algorithms to generate a good triangulation

Angle Conditions and Approximation

- Minimal angle condition: $\exists\, \alpha_{\min} > 0,\ \alpha_\tau \geq \alpha_{\min},\ \ \forall\, \tau \in \mathcal{T}_h,\ h \to 0$
  - P_2-FEM for Poisson $\|u - u_h\|_1 \lesssim h^2 / \sin \alpha_{\min}$ [Zlámal, 1968; Zenisek, 1969]
  - Similar estimate for the fourth-order clamped plane problem
  - Inscribed ball condition or $|\tau| \geq Ch^d$ [Ciarlet, 1978; Lin, Lin, 2003]
  - Small angles $\sim$ ill-conditioned linear systems [Xu, 1989; Shewchuk, 2002]
- Maximal angle condition: $\exists\, \alpha_{\max} < \pi,\ \alpha_\tau \leq \alpha_{\max},\ \ \forall\, \tau \in \mathcal{T}_h,\ h \to 0$
  - Minimal angle cond. $\Rightarrow$ maximal angle cond. $\Rightarrow$ essential for convergence
  - Interpolation error $\|u - I_h u\|_{1,\infty} \lesssim h|u|_{2,\infty}$ [Synge, 1957]
  - Sufficient for convergence of P_1-FEM [Feng, 1965; Babuška, Aziz, 1976]



$$A_1 = (-h, 0), A_2 = (h, 0), A_3 = (0, h^5)$$

$$u(x) = x_1^2,\ \|u - I_h u\|_1^2 \geq h^{-6} \cdot \tfrac{1}{2}(2h)h^5 = 1$$

Large interpolation error [Strang, Fix, 1973]

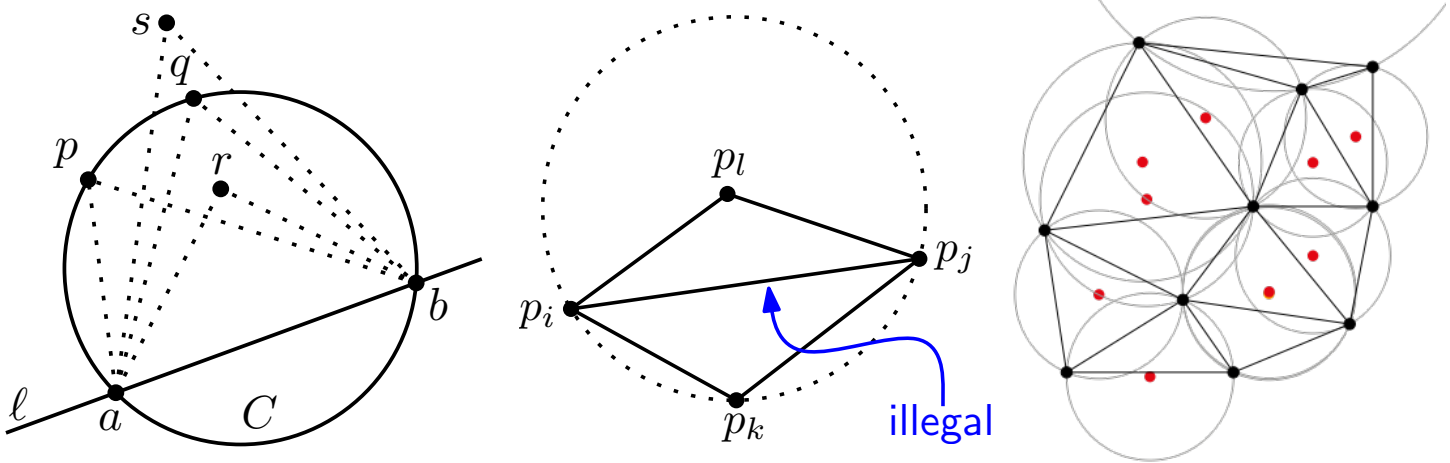
- Nonobtuse condition: $\alpha_\tau \leq \pi/2,\ \ \forall\, \tau \in \mathcal{T}_h$
  - Obtuse triangles can destroy the discrete maximum principle $f \geq 0 \Rightarrow u_h \geq 0$
  - Nonobtuse simplicial triangulations yields diagonally dominant stiffness matrices

Delaunay Triangulation

Delaunay triangulation

- Many possible choices; but which one is better? How to check?

Delaunay triangulation: a triangulation $\mathcal{T}(\mathbb{P})$ such that no point in $\mathbb{P}$ is inside the circum-hypersphere of any simplex

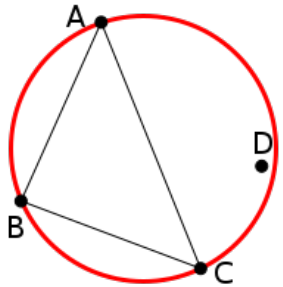


Properties of Delaunay triangulation

- Maximize the minimal angles
- The Delaunay triangulation contains at most $\mathcal{O}(n^{\lceil d/2 \rceil})$ simplexes
- The union of all simplexes in the triangulation is the convex hull of the points

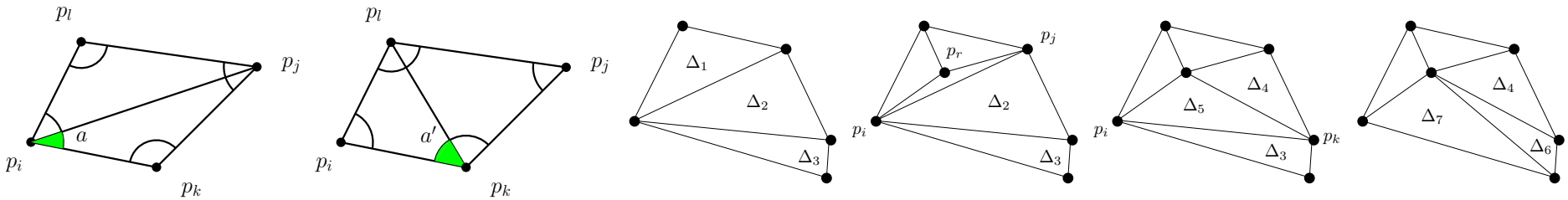
Algorithms for Delaunay Triangulation

Empty circle (sphere) condition



$$\Longleftrightarrow \quad \begin{vmatrix} A_1 & A_2 & A_1^2+A_2^2 & 1 \\ B_1 & B_2 & B_1^2+B_2^2 & 1 \\ C_1 & C_2 & C_1^2+C_2^2 & 1 \\ D_1 & D_2 & D_1^2+D_2^2 & 1 \end{vmatrix} > 0$$

Lawson's flip algorithm

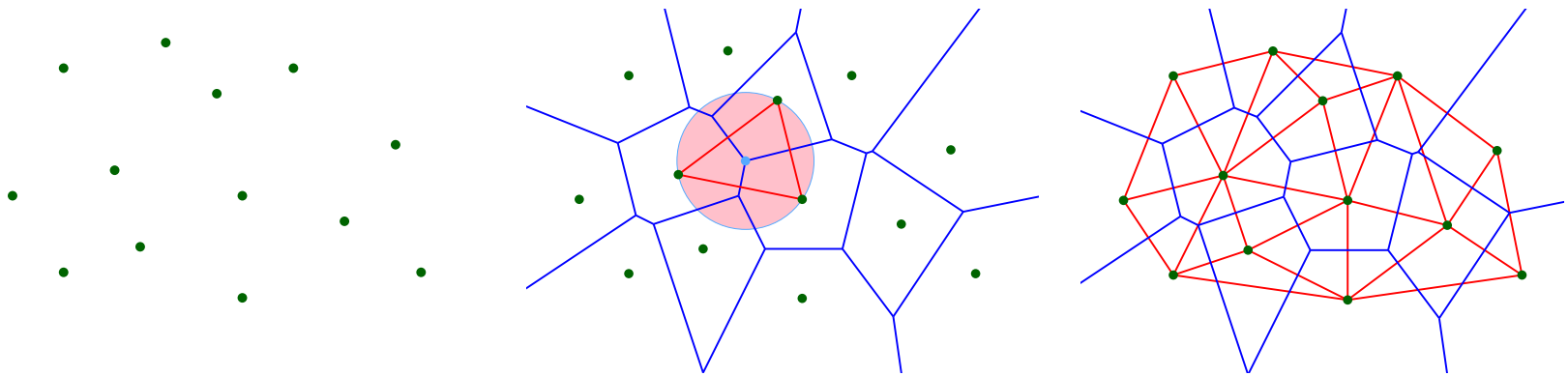


- Lawson flip algorithm terminates in finite steps
- Provides a constructive proof for the existence of Delaunay triangulation
- Sequential algorithms: [Su, Drysdale, 1996]
  - Incremental algorithms
  - Divide-and-conquer algorithms
  - Fortune's sweepline algorithms
  - Convex hull based algorithms: lift-and-project

Voronoi Diagram

Voronoi Diagram

- Voronoi cell of $p_k = \{x \in \mathbb{R}^d \ : \ \|x - p_k\| \leq \|x - p_j\|, \ \forall\, j \neq k\}$
- An edge of Voronoi diagram is equidistant to the two nearest sites
- Dual graph of the Delaunay triangulation



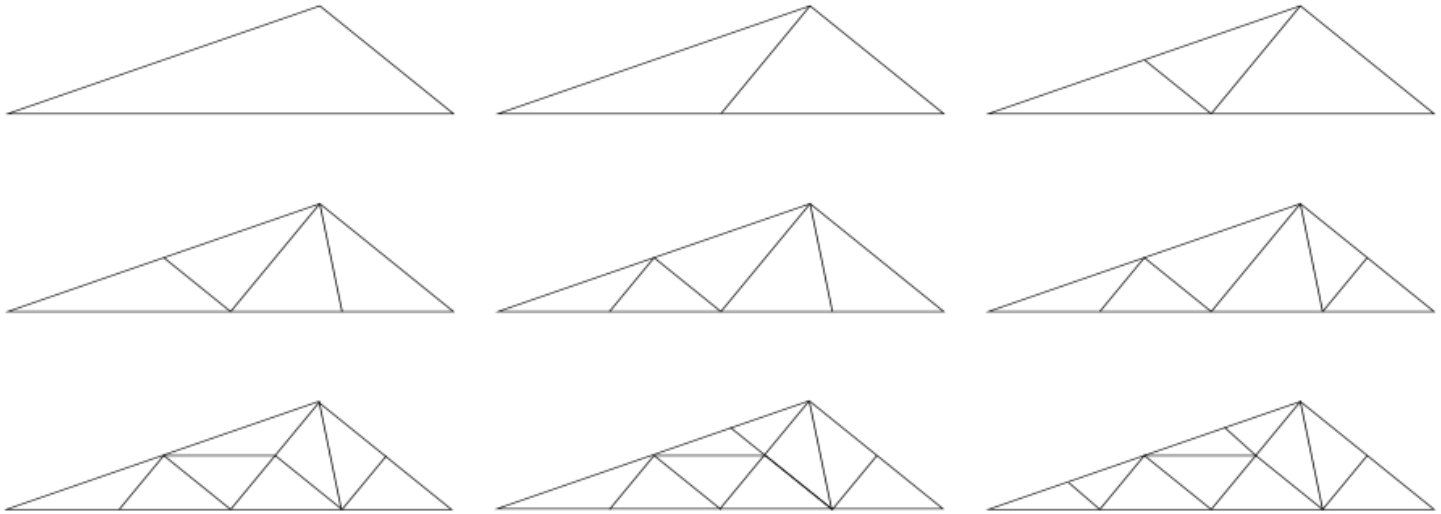
Generating Voronoi diagram

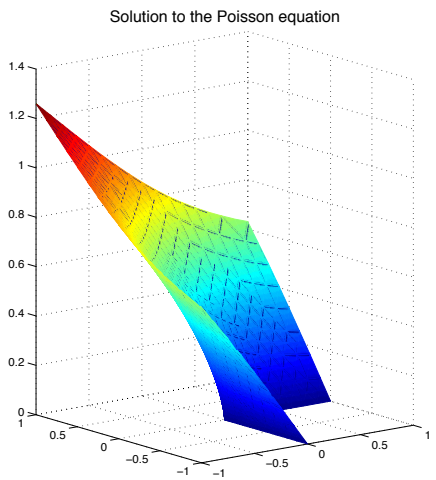
- Bowyer-Watson algorithm via Delaunay triangulation: $\mathcal{O}(n \log n)$ to $\mathcal{O}(n^2)$
- Fortune's algorithm: $\mathcal{O}(n \log n)$
- Lloyd's algorithm and k-means clustering

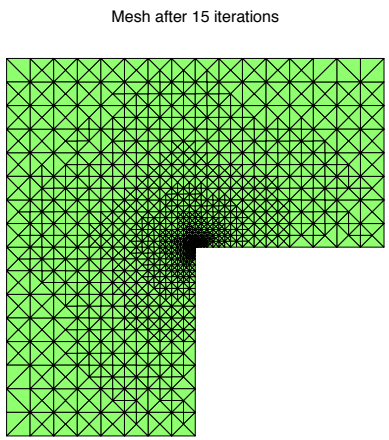
Dynamic demo of Voronoi diagram. https://bl.ocks.org/mbostock/4060366

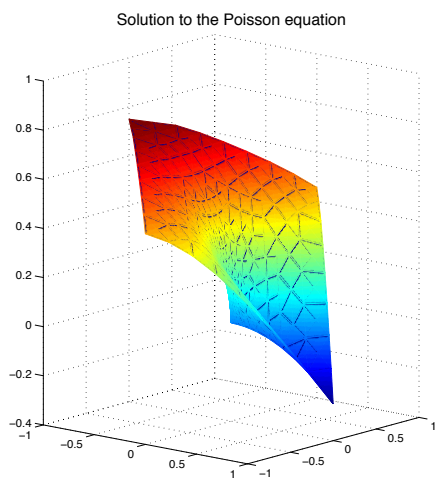
Adaptive Mesh Refinement

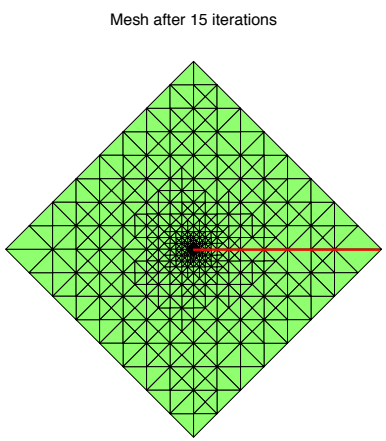
Red-green refinement, longest edge bisection, and newest vertex bisection





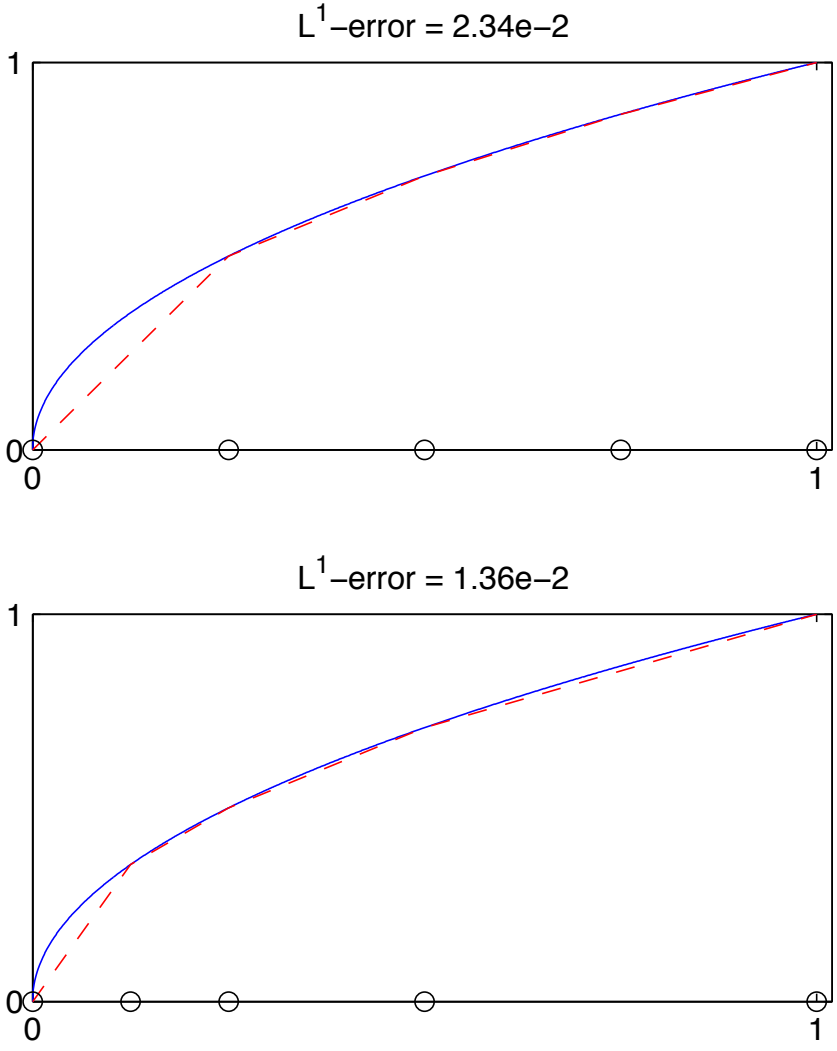


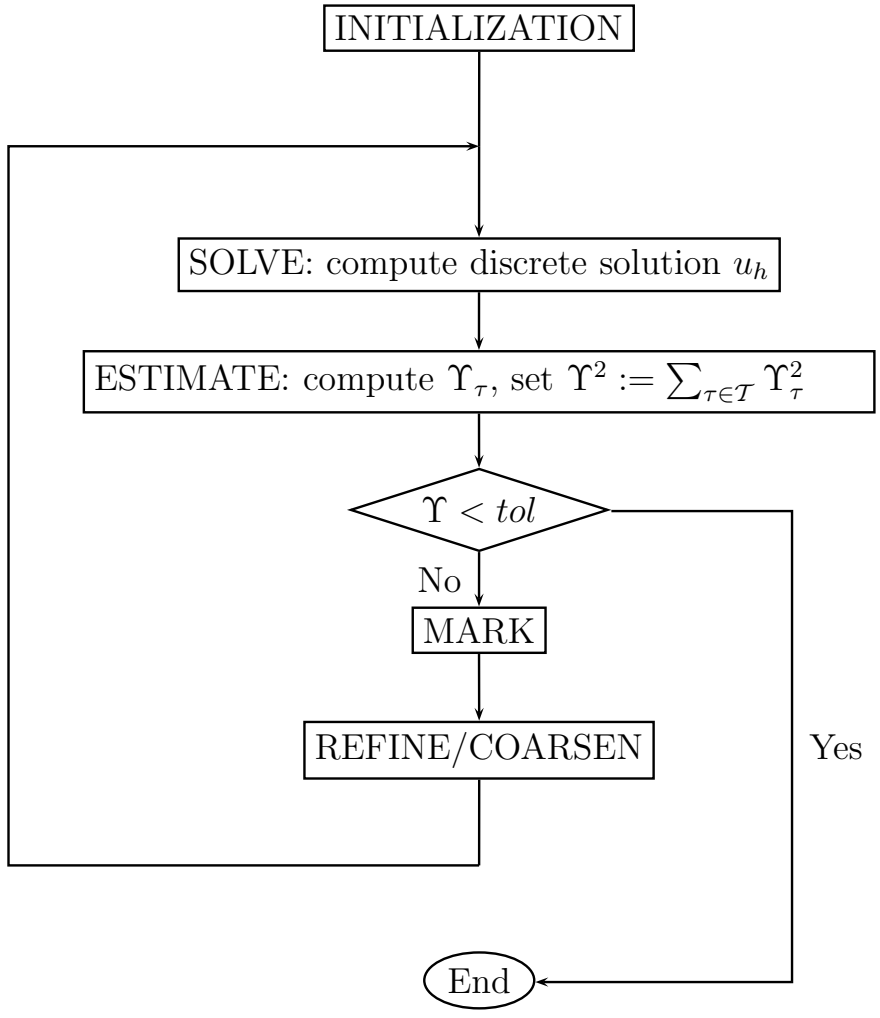




[Chen, Zhang, 2010]

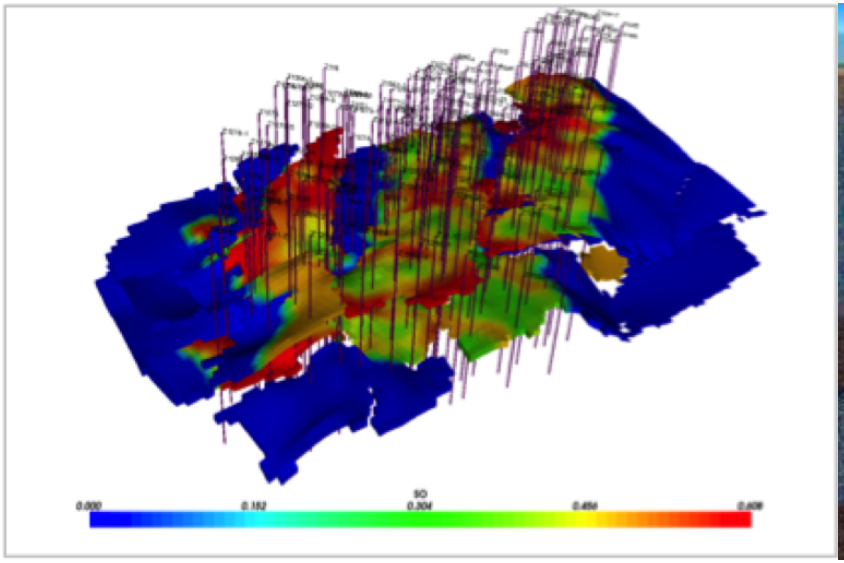
Adaptive Mesh and Nonlinear Approximation

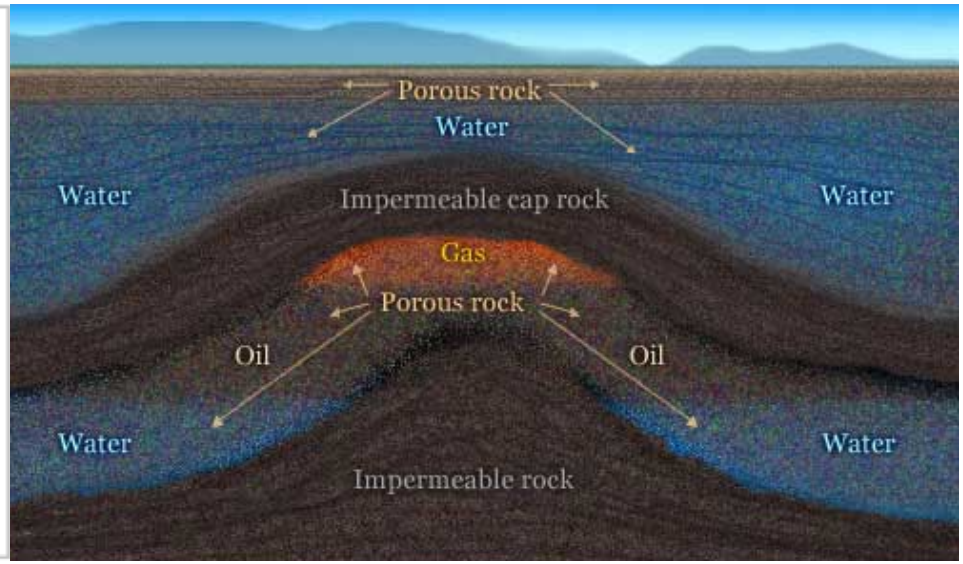


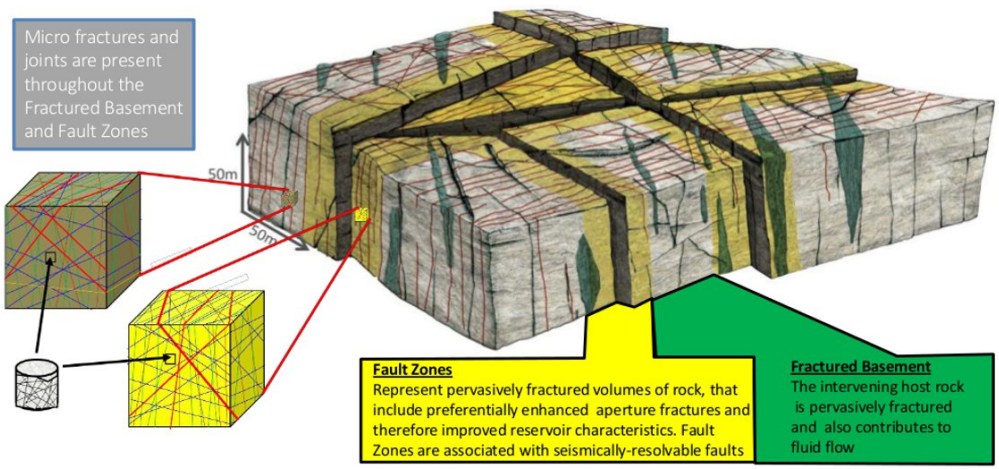


Approximate $f(x) = x^{1/2}$. Left: Solution and error; Right: Adaptive algorithm.

Why Gridding Is Challenging: Fractures

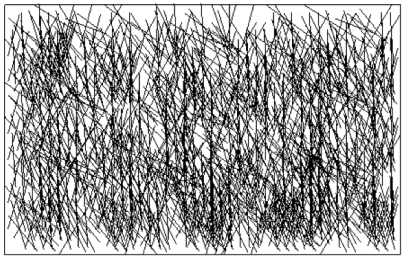


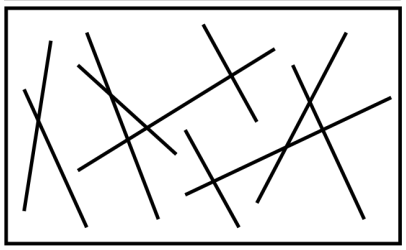


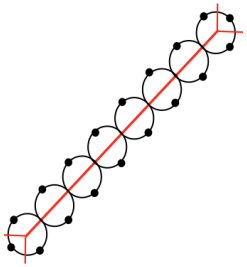


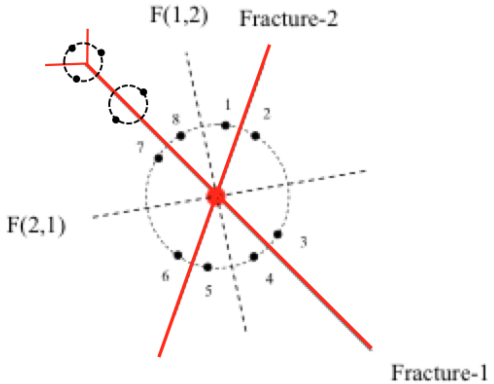
http://www.geoexpro.com/articles/2017/01/hiding-in-the-basement

Capturing Natural and Hydraulic Fractures



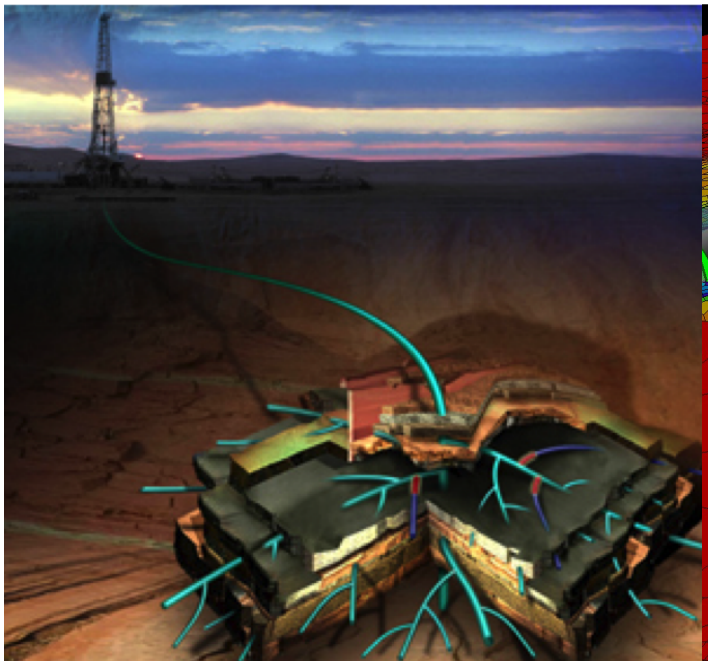


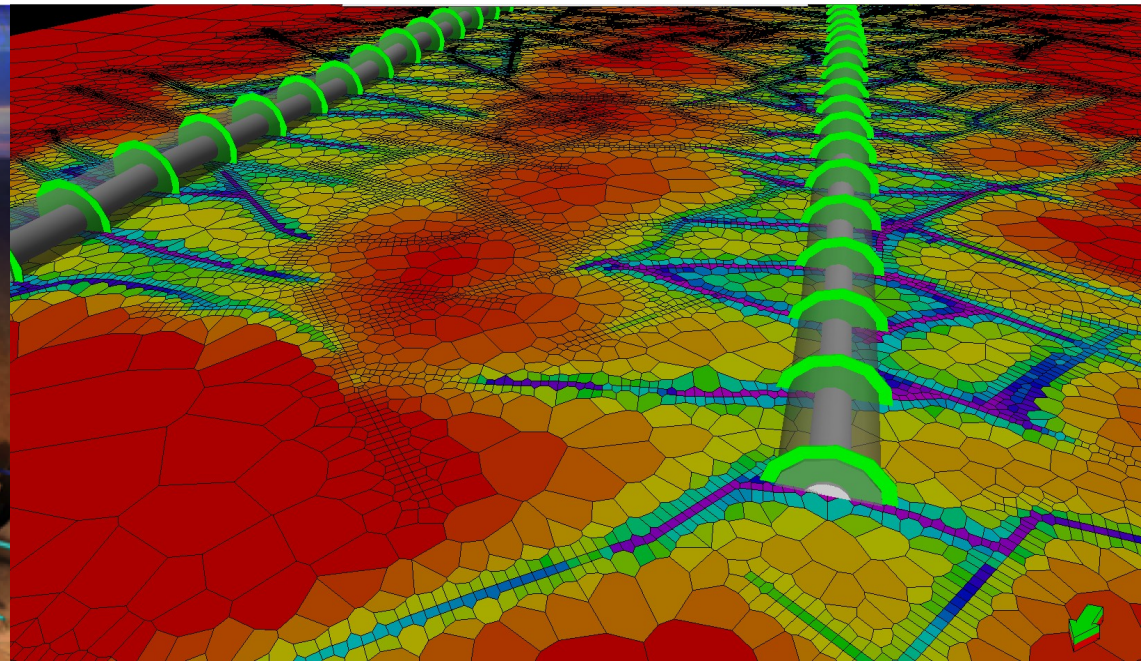




1. Dual continuum model: matrix-fractures, simple
   - DPDP [Warren, Root, 1963; Blaskovich, et al. 1983]
   - Well developed, connected, without localized anisotropy
   - Regard fractures as part of the pore volume
   - No flow occurs between matrix blocks
2. Equivalent porous media model: generalization of DCMs
   - Representative elementary volume
   - Multiple INteraction Continua: [Wu, Pruess, 1988]
3. Discrete fracture model (DFM): large-scale / isolated fractures
   - Representing fracture aperture / shape / direction explicitly
   - How multi-phase fluid flows inside the fracture network?
   - Flow-geomachanics coupling [Karimi-Fard, et al. 2004]
   - Unstructured grid / high computational cost $\Longrightarrow$ EDFM
4. Mixed discrete-continuum model:
   - Based on discrete fracture network analysis
   - Modeling reservoirs with multiscale fractures
   - Transportation between discrete and continuum parts

Why Gridding Is Challenging: Wells



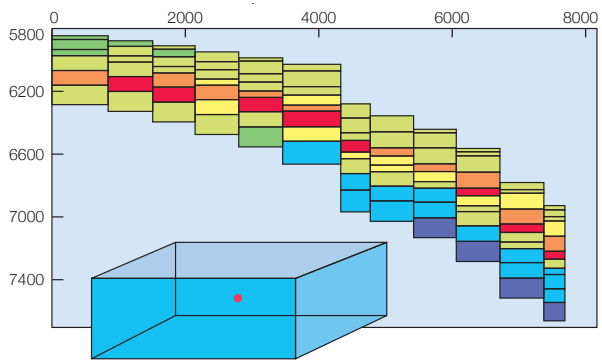


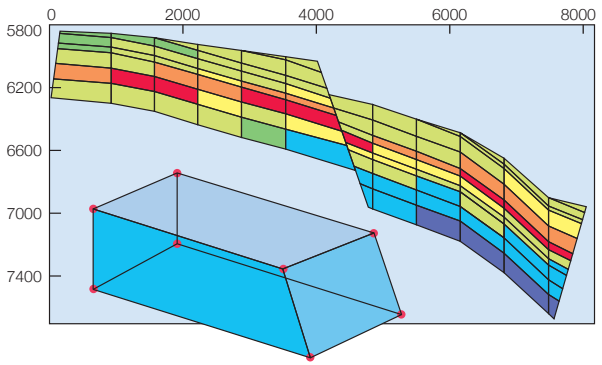
Different length scales:

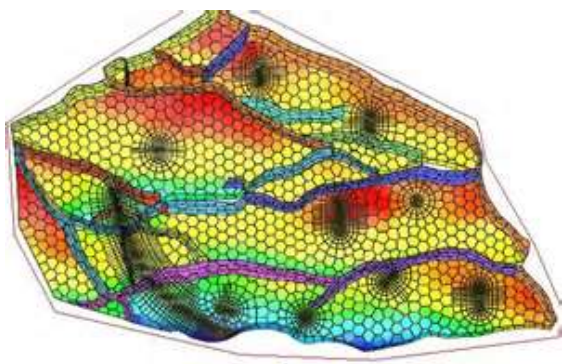
- Vertical grid size $\sim$ 10cm–1m
- Horizontal grid size $\sim$ 10m–100m
- Fractures $\sim$ 1cm–10m
- Well radius $\sim$ 1cm
- Well length $\sim$ 100m

Pictures from Saudi Aramco and Schlumberger

Grid Partitions for Reservoir Simulation







1. Cartesian block-centered grids
   - CNOOC: SOCF (2009.12—2011.6)
   - Easy for implementation
   - Multiple-domain, local refinement
   - Difficult to simulate fault/dip
2. Corner-point grids
   - PetroChina: HiSim (2011.1—2015.12)
   - A type of hexahedral grid
   - Logically still structured
   - Difficult to compute flux accurately
3. Unstructured grids (PEBI and beyond)
   - PennSim (2013.1—2016.12) $\Longrightarrow$ ExSim
   - Voronoi, 2.5D
   - Better description of faults and wells
   - Incompatible with structured seismic data
   - Challenges in discretizations and solvers

[Goldthorpe, et al. 1985; Heirich, 1987; Heinemann, 1989; SPE8, 1993; Palagi, 1994]

§3. Discretization Methods

23 Fully Implicit Discretization

24 Simplified Oil-Water Two-Phase Model

25 IMPES/IMPEC Discretization

26 Galerkin Method

27 Finite Element Method

28 Finite Volume Method

29 Cell-Centered and Vertex-Centered FVM

30 Control-Volume Finite Element Method

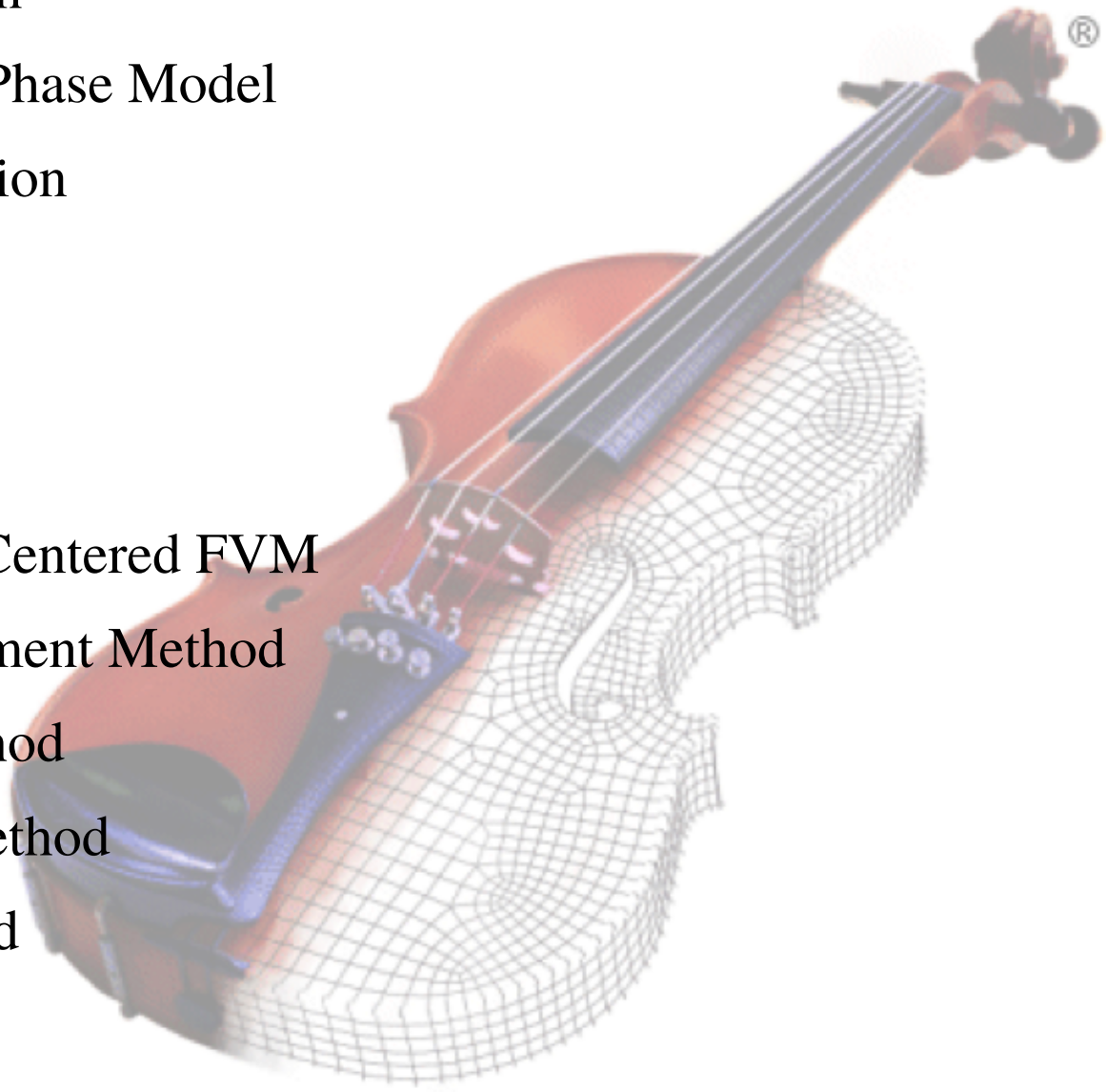
31 Mixed Finite Element Method

32 Discontinuous Galerkin Method

33 Interior Penalty DG Method

34 Weak Galerkin Method

35 Weak Galerkin Finite Element Method



Fully Implicit Discretization

Set of equations and unknowns

- FIM or SS discretization [Douglas, Peaceman, Rachford, 1959]
- Primary equations: n_c mass conservation laws + volume balance:

$$V^{\text{fluid}}(P, N_1, \ldots, N_{n_c}) = V^{\text{pore}}(P)$$

- Secondary equations: phase equilibrium, density, relative permeability, ...
- Primary unknowns: $\vec{X} := (P, N_1, \ldots, N_{n_c})^T$ $\longleftarrow$ More variables!
- Secondary unknowns: $\vec{Y} := (x_{11}, \ldots, x_{n_c n_p}, S_1, \ldots, S_{n_p})^T$

Discrete linear equations (no reaction term)

- Update the primary unknowns (Backward Euler + FVM + Newton)

$$\Psi_i := \frac{N_i^{n+1} - N_i^n}{\Delta t} + \sum_s F_{i,s}^{n+1} - \sum_w Q_{i,k,w}^{n+1} = 0 \qquad i = 1 : n_c$$

$$\Psi_0 := V^{\text{pore}} - V^{\text{fluid}} = 0$$

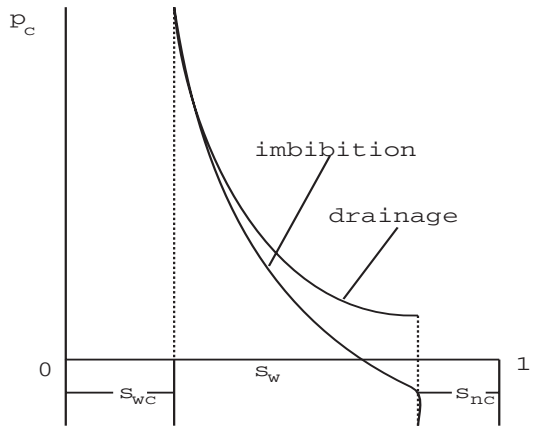
- Jacobian matrix $J := \dfrac{d\vec{\Psi}}{d\vec{X}} = \dfrac{\partial \vec{\Psi}}{\partial \vec{X}} + \dfrac{\partial \vec{\Psi}}{\partial \vec{Y}} \dfrac{\partial \vec{Y}}{\partial \vec{X}}$ $\longleftarrow$ More expensive!

Simplified Oil-Water Two-Phase Model

In order to introduce IMPES/IMPEC, we give a simplified model

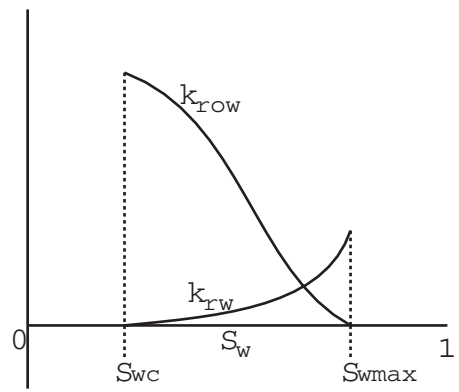
1. Mass conservation:

$$\frac{\partial}{\partial t}\left(\phi\rho_w S_w\right) = -\nabla\cdot\left(\rho_w \mathbf{u}_w\right) + Q_W$$
$$\frac{\partial}{\partial t}\left(\phi\rho_o S_o\right) = -\nabla\cdot\left(\rho_o \mathbf{u}_o\right) + Q_O$$



2. Darcy's law and constitutive equations:

$$\mathbf{u}_\alpha = -\frac{k k_{r\alpha}}{\mu_\alpha}(\nabla P_\alpha - \rho_\alpha \mathfrak{g}\nabla z), \quad \alpha = o, w$$
$$S_o + S_w = 1$$
$$P_o - P_w = 0$$



3. Well constraints + B.C. + I.C.

IMPES/IMPEC Discretization

Implicit pressure / explicit saturation (concentration)

- Separate computation of pressure from that of saturation/concentration
  [Sheldon-Zondek-Cardwell 1959; Stone-Garder 1961; Collins et al. 1992]
- Two-phase classical IMPES: Define the total velocity $\mathbf{u} = \mathbf{u}_o + \mathbf{u}_w$ and then

$$\nabla \cdot \mathbf{u} = \frac{Q_w}{\rho_w} + \frac{Q_o}{\rho_o}$$

$$\mathbf{u} = -k\left[\left(\frac{k_{rw}}{\mu_w} + \frac{k_{ro}}{\mu_o}\right)\nabla P - \left(\frac{k_{rw}}{\mu_w}\rho_w + \frac{k_{ro}}{\mu_o}\rho_o\right)\mathfrak{g}\nabla z\right]$$

- Obtaining an equation for pressure: $-\nabla \cdot (\mathbf{K}\nabla P) = Q$
- Updating saturation/concentration with explicit time-marching

Pros & Cons and Variants

- The linear system to solve is SPD: "easy"

☞ Not as stable as FIM $\Longrightarrow$ requires small time stepsize Δt

- Other improvements: Smaller Δt for saturation update; used in Newton iterations; adaptive scheme; ...

Galerkin Method

Model problem and its weak form

$$\begin{cases} -\nabla\cdot\nabla p &= f, \quad \Omega \\ p &= 0, \quad \partial\Omega \end{cases} \quad \Longrightarrow \quad p \in V: \ a[p,q] = (f,q), \ \forall\, q \in V$$

where $V := H_0^1(\Omega)$ and

$$a[p,q] := -\int_\Omega (\nabla\cdot\nabla p) q \, dx = \int_\Omega \nabla p \cdot \nabla q \, dx - \int_{\partial\Omega} (\nabla p \cdot \mathbf{n})\, q \, dS = (\nabla p, \nabla q)$$

Galerkin approximation: Choose a finite dimensional space $V_N \subset V$, such that

$$p_N \in V_N: \quad a[p_N, q_N] = (f, q_N), \quad \forall\, q_N \in V_N$$

Galerkin orthogonality: $\quad a[p - p_N, q_N] = 0, \quad \forall\, q_N \in V_N$

Theorem (Quasi-optimality property)

If the bilinear form $a[\cdot,\cdot]$ is continuous and coercive, then we have

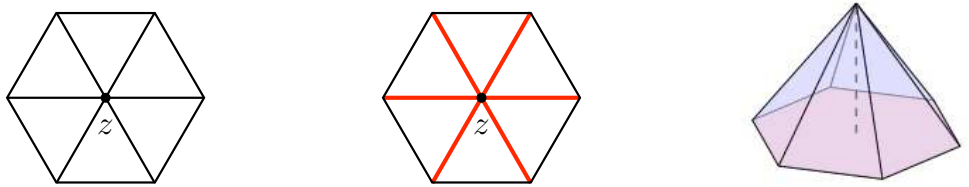
$$\|p - p_N\|_1 \lesssim \|p - q_N\|_1, \quad \forall\, q_N \in V_N$$

Finite Element Method

Piecewise linear finite element

Let $V_N = \text{span}\{\psi_1, \ldots, \psi_N\}$ be the space of continuous p.w. linear polynomials on a quasi-uniform triangulation and $p_h = \sum_{j=1}^{N} P_j \psi_j$. Then we have

$$\sum_{j=1}^{N} a[\psi_j, \psi_i] P_j = (f, \psi_i), \quad i = 1, \ldots, N$$



(a) Local patch ω_z (b) Skeleton γ_z (c) Basis function ψ_z

Error estimate: $\|p - p_h\|_1 \lesssim \inf_{q \in V_N} \|p - q\|_1 \lesssim h\|p\|_2$

Forming a linear algebraic system

We then need to solve a linear equation for $\vec{P} = (P_1, \ldots, P_N)^T$

$$A\vec{P} = \vec{R}, \quad \text{with} \quad A_{ij} = a[\psi_j, \psi_i] \quad \text{and} \quad \vec{R} = \big((f, \psi_1), \ldots, (f, \psi_N)\big)^T$$

[Hrennikoff, 1941; Courant, 1943; Feng, 1965; Ciarlet, 1978; ...]

Finite Volume Method

Model problem and its primitive form

$$\begin{cases} -\nabla\cdot\nabla p &= f, \quad \Omega \\ p &= 0, \quad \partial\Omega \end{cases} \quad \Longrightarrow \quad \int_{\partial\omega} F(p)\, dS = \int_{\omega} f\, dx, \quad \forall\, \omega \subset \Omega$$

where $F(p) := -\nabla p \cdot \mathbf{n}$ is the flux across the volume boundary $\partial\omega$. It is also called the surface integral from.

Classical finite volume method

- Partition the domain Ω into control volumes $\omega_i \subset \Omega$
- Choose an approximation space of p in each control volume
- Approximate boundary flux $F(p)$ by an numerical flux $\tilde{F}(p_h)$

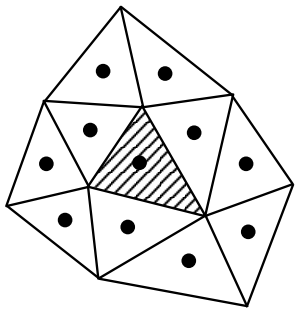
$$\frac{1}{|\omega_i|}\int_{\partial\omega_i} \tilde{F}(p_h)\, dS = \frac{1}{|\omega_i|}\int_{\omega_i} f\, dx, \quad \forall\, \omega_i \subset \Omega$$

On orthogonal grids: $$F_e(p) \approx \frac{p_h|_{\tau_{\text{in}}} - p_h|_{\tau_{\text{out}}}}{|c_{\text{in}} - c_{\text{out}}|} =: \tilde{F}_e(p_h)$$

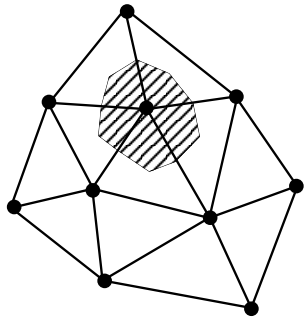
Centered difference method on irregular grids [Heinrich, 1987; LeVeque, 2002]

Cell-Centered and Vertex-Centered FVM

Mesh and dual mesh

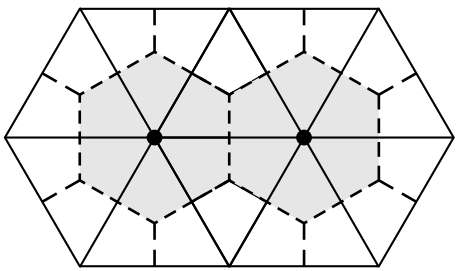


(a) Mesh and dual mesh of cell-centered FVM

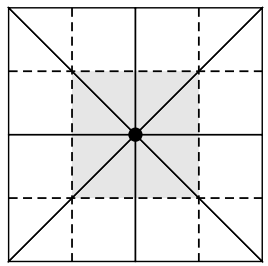


(b) Mesh and dual mesh of vertex-centered FVM

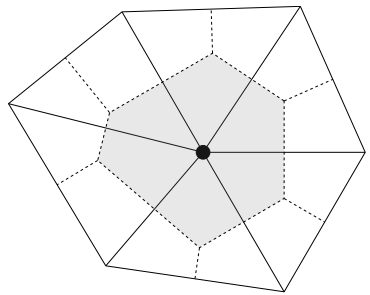
Choices of control volume



(a) Type A



(b) Type B



(c) Type C

- Type A: c_τ = barycenter of τ, commonly used for equilateral triangles
- Type B: c_τ = middle point of longest edge of τ, better for right triangles
- Type C: c_τ = circumcenter of τ, for Delaunay triangulations (dual mesh is a Voronoi diagram)

Box methods [Bank, Rose, 1987; Hackbusch, 1989]

Control-Volume Finite Element Method

Petrov-Galerkin formulation

Let $\mathcal{D}$ be the dual mesh of $\mathcal{T}$. Define a piecewise constant space

$$V_{\mathcal{D}} := \{v \in L^2(\Omega) \,:\, v|_{\omega_i} = \text{const},\ \forall\, \omega_i \in \mathcal{D}\}.$$

Now we choose $p \in V_N$ and $q \in V_{\mathcal{D}}$. In this case, we have the bilinear form

$$\bar{a}[p,q] := -\sum_{e \in \mathcal{E}(\mathcal{D})} \int_e (\nabla p \cdot \mathbf{n}_e)\,[q]\, dS$$

Control-volume finite element method

Find $p_h \in V_N$, such that $\bar{a}[p_h, q] = (f, q), \quad \forall\, q \in V_{\mathcal{D}}$

Choose the standard basis functions for $V_{\mathcal{D}}$: $\chi_i(x) = 1, x \in \omega_i$; $\chi_i(x) = 0, x \notin \omega_i$

$$\bar{A}\vec{P} = \vec{R}, \quad \text{with } \bar{A}_{ij} = \bar{a}[\psi_j, \chi_i] = -\int_{\partial\omega_i} \nabla\psi_j \cdot \mathbf{n}\, dS \text{ and } R_i = \int_{\omega_i} f\, dx$$

Special case: If $\partial\omega \cap \partial\tau$ contains the middle points of edges, then $\bar{A} = A$.

Error estimate: $\|p - p_h\|_1 \lesssim h(\|p\|_2 + \|f\|_0)$

Mixed Finite Element Method

Model problem and its mixed form

$$\left\{\begin{array}{rcll} -\nabla\cdot\nabla p &=& f, & \Omega \\ p &=& 0, & \partial\Omega \end{array}\right. \Longrightarrow \left\{\begin{array}{rcll} \mathbf{u}-\nabla p &=& 0, & \Omega \\ \nabla\cdot\mathbf{u} &=& f, & \Omega \\ p &=& 0, & \partial\Omega \end{array}\right.$$

Weak form of the mixed problem: Find $(\mathbf{u}, q) \in H(\mathrm{div},\Omega)\times L^2(\Omega)$ such that

$$\left\{\begin{array}{rcll} \int_\Omega \mathbf{u}\cdot\mathbf{v}\,dx + \int_\Omega p\,\nabla\cdot\mathbf{v}\,dx &=& 0, & \forall\,\mathbf{v}\in H(\mathrm{div},\Omega) \\ \int_\Omega \nabla\cdot\mathbf{u}\,q\,dx &=& \int_\Omega f q\,dx, & \forall\, q\in L^2(\Omega) \end{array}\right.$$

Mixed finite element and algebraic system

- More variables and more difficult to analyze / to solve
- Need to solve a saddle-point type algebraic system

$$\begin{pmatrix} A & B^T \\ B & \end{pmatrix}\begin{pmatrix} \vec{U} \\ \vec{P} \end{pmatrix} = \begin{pmatrix} \vec{0} \\ \vec{R} \end{pmatrix}$$

- Hybridization and static condensation $\Longrightarrow$ Schur complement $\Longrightarrow$ reduce system size

[Brezzi 1973; Crouzeix, Raviart, 1973; Falk, Osborn, 1980; ...]

Discontinuous Galerkin Method

Complications of continuous Galerkin methods

- Mesh generation: How to handle meshes with hanging nodes
- Approximation functions: P^k used on triangles and Q^k used on quadrilaterals; complicate to construct C^1 conforming elements

Broken (discontinuous) Sobolev spaces

- $\mathcal{T}_h$ is a shape-regular quasi-uniform (conforming or not) triangulation of Ω
- $H^k(\mathcal{T}_h) := \{v \in L^2(\Omega) \,:\, v|_\tau \in H^k(\tau),\ \forall\, \tau \in \mathcal{T}_h\}$

DG formulations:

Assume $p \in H^2(\Omega) \cap H_0^1(\Omega)$ and test the model equation with discontinuous q

$$\sum_{\tau\in\mathcal{T}_h} \int_\tau \nabla p \cdot \nabla q \, dx - \sum_{\tau\in\mathcal{T}_h} \int_{\partial\tau} (\nabla p \cdot \mathbf{n}_e) q \, dS = \int_\Omega f q \, dx, \quad \forall\, q \in H_0^1(\mathcal{T}_h)$$

$$\Longrightarrow \sum_{\tau\in\mathcal{T}_h} \int_\tau \nabla p \cdot \nabla q \, dx - \sum_{e\in\mathcal{E}_h} \int_e \mathbf{n}_e \cdot \{\nabla p\}[q] \, dS = \int_\Omega f q \, dx, \quad \forall\, q \in H_0^1(\mathcal{T}_h)$$

[Reed, Hill, 1973; Lesaint, Raviart, 1974; ...]

Interior Penalty DG Method

Problems with the previous weak form

- No guarantee on the well-posedness $\Longrightarrow$ enforce continuity in a weaker sense
- No symmetry in the weak formulation $\Longrightarrow$ symmetrization

Introduce interior penalty

- $J_h^\sigma(p,q) := \sum_{e\in\mathcal{E}_h} \int_e \sigma[p][q]\,dS, \quad p,q \in H^1(\mathcal{T}_h); \qquad \sigma = \alpha h_e^{-1}$
- DG norm $|\!|\!| v |\!|\!|_{\mathrm{DG}} := \left(|v|_{H^1(\mathcal{T}_h)} + J_h^\sigma(v,v) \right)^{1/2}$

Examples of IPDG methods

- $a_h^i[p,q] := \sum_{\tau\in\mathcal{T}_h} \int_\tau \nabla p\cdot\nabla q\,dx - \sum_{e\in\mathcal{E}_h} \int_e \mathbf{n}_e\cdot\{\nabla p\}[q]\,dS + J_h^\sigma(p,q)$
- $a_h^s[p,q] := \sum_{\tau\in\mathcal{T}_h} \int_\tau \nabla p\cdot\nabla q\,dx - \sum_{e\in\mathcal{E}_h} \int_e (\mathbf{n}_e\cdot\{\nabla p\}[q] + \mathbf{n}_e\cdot\{\nabla q\}[p])\,dS + J_h^\sigma(p,q)$
- $a_h^n[p,q] := \sum_{\tau\in\mathcal{T}_h} \int_\tau \nabla p\cdot\nabla q\,dx - \sum_{e\in\mathcal{E}_h} \int_e (\mathbf{n}_e\cdot\{\nabla p\}[q] - \mathbf{n}_e\cdot\{\nabla q\}[p])\,dS + J_h^\sigma(p,q)$
- IIPG, SIPG, NIPG [Dawson, Sun, Wheeler, 2004; Wheeler, 1978; Rivière, Wheeler, Girault, 1999]

Error estimate: $|\!|\!| p - p_h |\!|\!|_{\mathrm{DG}} \lesssim h|p|_2$

Weak Galerkin Method

Problems with the DG methods

- Complicate finite element formulations
- More unknowns than the continuous Galerkin methods

Weak Galerkin approximation

- Use discontinuous approximation functions:

$$W_h := \left\{ v = \{v_0, v_b\} \,:\, v_0|_\tau \in P_j(\tau),\ v_b|_e \in P_l(e),\ v_b|_{\partial\Omega} = 0 \right\}$$

- Keep the simple weak form as the continuous Galerkin methods:

$$\text{Find } p_h \in W_h \text{ s.t. } (\nabla_w p_h, \nabla_w q_h) + \sigma(p_h, q_h) = (f, v), \quad \forall\, q_h \in W_h$$

- Define weak gradient $\nabla_w v \in [P_r(\tau)]^d$:

$$(\nabla_w v, \mathbf{w})_\tau := -(v_0, \nabla \cdot \mathbf{w})_\tau + \int_{\partial\tau} v_b \mathbf{w} \cdot \mathbf{n}\, dS, \quad \forall\, \mathbf{w} \in [P_r(\tau)]^d$$

- For example, $\left(P_j(\tau), P_l(e), [P_r(\tau)]^d\right) = \left(P_1(\tau), P_0(e), [P_0(\tau)]^d\right)$

[Wang, Ye, 2013]

Weak Galerkin Finite Element Method

WG finite element formulation

Define a bilinear form

$$a_w[p_h, q_h] := (\nabla_w p_h, \nabla_w q_h) + \sum_\tau \frac{\alpha}{h_\tau}(p_{h,0} - p_{h,b}, q_{h,0} - q_{h,b})_{\partial\tau}$$

Find $p_h = \{p_{h,0}, p_{h,b}\} \in W_h$ such that

$$a_w[p_h, q_h] = (f, q_h), \quad \forall\, q_h \in W_h$$

Theorem (Error Estimate)

The WG solution $p_h \in \left(P_1(\tau), P_0(e), [P_0(\tau)]^d\right)$ satisfies that

$$h|\!|\!|Q_h p - p_h|\!|\!| + \|Q_h p - p_h\|_0 \lesssim h^2 \|p\|_2$$

Implementation of WG finite element method: Static condensation

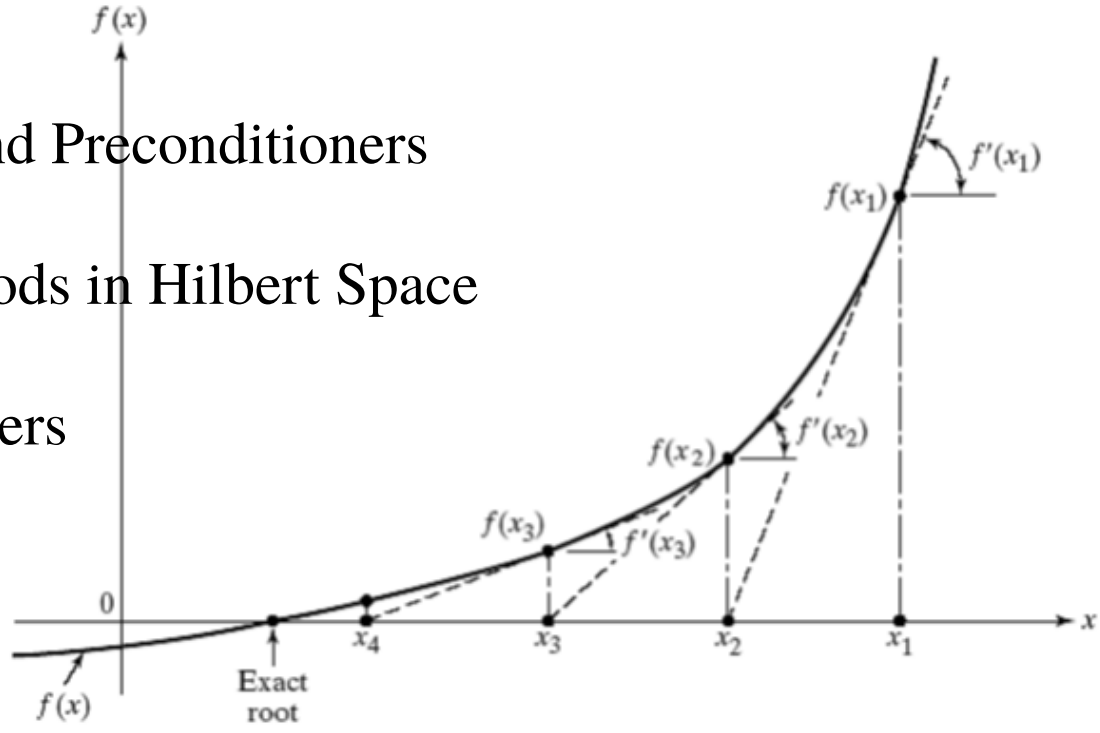
1. Local problem: Fix $p_{h,b}$ and solve for $p_{h,0}$ such that

$$a_w[p_h, q_h] = (f, q_h), \quad \forall\, q_h = \{q_{h,0}, 0\} \in W_h$$

2. Global problem: Solve $p_{h,b}$ such that

$$a_w[p_h, q_h] = (f, q_h), \quad \forall\, q_h = \{0, q_{h,b}\} \in W_h$$

§4. Iterative Solvers



Linear Algebraic Solvers

A fundamental problem in scientific computing:

> Given a sparse matrix $A \in \mathbb{R}^{N\times N}$ and $f \in \mathbb{R}^N$, solve $Au = f$!

In many applications, it takes most of the simulation time!

- General purpose direct solvers: Gaussian Elimination, ...
  - Robust, exact, multiple right-hand sides, ...
  - Black Box $\Longrightarrow$ Many packages available: PARDISO, MUMPS, SPOOLES, SuiteSparse (CHOLMOD/UMFPACK), SuperLU, WSMP, H2Lib, ...
  - Memory: Require explicit matrices, need more RAM for decomposition
  - Computation: General $\mathcal{O}(N^3)$, banded $\mathcal{O}(N^2)$, nested dissection $\mathcal{O}(N^{1.5})$ [George 1973; Duff-Erisman-Reid 1986; Demmel 1997]
- $\mathcal{H}$-matrix, data-sparsity, low-rank approximation: [Hackbush 1999; Chandrasekharan-Gu-Lyons 2005; Xia-Chandrasekharan-Gu-Li 2009, 2010; Ho-Greengard 2012; Schmitz-Ying 2012; ...]
- Specialized methods: FFT, ...

Iterative Solution Methods

Pros:

- Optimal cost is possibly: $\mathcal{O}(N|\log N|^{\sigma})$ operations
- Adjustable accuracy with good initial guess in practice
- Matrix-free operations can be used
- Singular or nearly singular problems

Cons:

- Problem-dependence: require different methods for different problems
- Robustness: (arguably) biggest disadvantage in practice
- Optimality: optimal algorithm or fastest algorithm?
- Implementation: difficult if not impossible to make efficient software

Goals:

> convergence, robustness, optimality, efficiency, scalability, reliability

Krylov Subspace Methods and Preconditioners

1. Conjugate gradient method for $Au = f$

$$\frac{\|u - u^m\|_A}{\|u - u^0\|_A} \leq 2 \left(\frac{\sqrt{\kappa(A)} - 1}{\sqrt{\kappa(A)} + 1} \right)^m \quad (m \geq 1), \quad \kappa(A) := \frac{\lambda_{\max}(A)}{\lambda_{\min}(A)}$$

2. Convergence estimate using effective condition number

$$\frac{\|u - u^m\|_A}{\|u - u^0\|_A} \leq 2C \left(\frac{\sqrt{b/a} - 1}{\sqrt{b/a} + 1} \right)^{m - m_0} \quad (m \geq m_0)$$

- decomposition: $\sigma(A) = \sigma_{\text{bad}}(A) \cup \sigma_{\text{eff}}(A)$ with m_0 entries in $\sigma_{\text{bad}}(A)$
- effective condition number $\kappa_{\text{eff}}(A) := b/a = \max \sigma_{\text{eff}} / \min \sigma_{\text{eff}}$
- constant $C := \max_{\lambda \in \sigma_{\text{eff}}(A)} \prod_{\mu \in \sigma_{\text{bad}}(A)} \left| 1 - \frac{\lambda}{\mu} \right|$
- $C < 1$ if σ_{bad} are isolated large eigenvalues; $C \leq \left| \kappa(A) - 1 \right|^{m_0}$ in general

☞ Preconditioners:

- Incomplete factorizations: ILUk, ILUt, ILUtp, ...
- Domain decomposition methods: RAS, FETI, BDDC, ...
- Multilevel preconditioners: AMG, GAMG, GMG, ...

Preconditioned Krylov Methods in Hilbert Space

1. What about more general problems with $A : X \to X' \supset X$?
   - Need to construct a SPD $B : X' \to X$ to make KSM's to work
   - If A is SPD, then $\langle \cdot, \cdot \rangle_A := \langle A\cdot, \cdot \rangle$ defines an inner-product and
     $$\langle BAx, y \rangle_A = \langle ABAx, y \rangle = \langle Ay, BAx \rangle = \langle Ax, BAy \rangle = \langle x, BAy \rangle_A$$
   - BA is a SPD in terms of $\langle A\cdot, \cdot \rangle$ or $\langle B^{-1}\cdot, \cdot \rangle$
   - Convergence estimate of CG holds true with $\kappa(BA)$
2. Find a natural (canonical) preconditioner for continuous problem
   - Bilinear form $a : X \times X \to \mathbb{R}$ is symmetric and bounded, and it satisfies the inf-sup condition $\inf_{x \in X} \sup_{y \in X} \frac{a(x,y)}{\|x\|_X \|y\|_X} \geq \gamma > 0$
   - For $f \in X'$, let $B : X' \to X$ be a Riesz operator $(Bf, y)_X = \langle f, y \rangle$
   - Then $BA : X \to X$ is symmetric in $(\cdot, \cdot)_X$ and $\kappa(BA) \leq C_a/\gamma$!
     - $\|BA\| \leq \sup_{x \in X} \frac{(BAx, x)_X}{\|x\|_X^2} = \sup_{x \in X} \frac{|a(x,x)|}{\|x\|_X^2} \leq C_a$
     - $\|(BA)^{-1}\|^{-1} = \inf_{x \in X} \frac{\|BAx\|_X}{\|x\|_X} = \inf_{x \in X} \sup_{y \in X} \frac{a(x,y)}{\|x\|_X \|y\|_X} \geq \gamma$

[Mardal, Winther, 2011]

Construction of Preconditioners

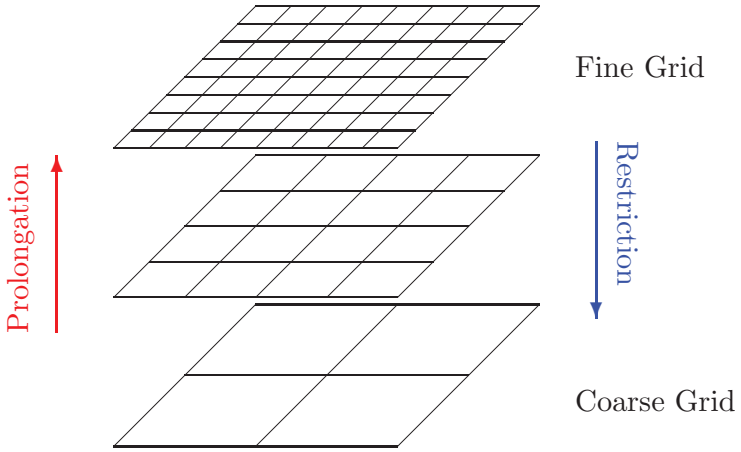
1. What does a "natural" preconditioner look like?
   - $A : H_0^1(\Omega) \to H^{-1}(\Omega)$, $\langle Au, v\rangle := a[u,v] = \int_\Omega (\alpha(x)\nabla u)\cdot\nabla v\, dx$
   - Kernel $\alpha(x) \in \mathbb{R}^{d\times d}$ satisfies $\gamma|\xi|^2 \le \xi^T\alpha(x)\,\xi \le C_a|\xi|^2$
   - Define $B = (-\Delta)^{-1} : H^{-1}(\Omega) \to H_0^1(\Omega) \implies \kappa(BA) \le C_a/\gamma < \infty$
   - Stokes problem: $a\big((u,p),(v,q)\big) := \langle\nabla u, \nabla v\rangle + \langle p, \nabla\cdot v\rangle + \langle q, \nabla\cdot u\rangle$
   - $A : \big[H_0^1(\Omega)\big]^d \times L_0^2(\Omega) \to \big[H^{-1}(\Omega)\big]^d \times L_0^2(\Omega)$
     $\implies B = \mathrm{diag}\big[(-\Delta)^{-1}, \ldots, (-\Delta)^{-1},\ I\big] \implies$ Block Trig Precond, ...
2. Solve a discrete problem $\implies$ Employ a stable discretization
   - Stable discretization, i.e., $\inf_{x\in X_h}\sup_{y\in X_h}\frac{a(x,y)}{\|x\|_X\|y\|_X} \ge \gamma_1 > 0$
   - Condition number can be bounded $\kappa(B_hA_h) \le C_a/\gamma_1$
3. Construct a cheap spectral-equivalent preconditioner
   - Discretization, grid generation/adaptation, parallelization, ...
   - Example: need components like $(-\Delta)^{-1}$ when solving Stokes, Darcy, ...

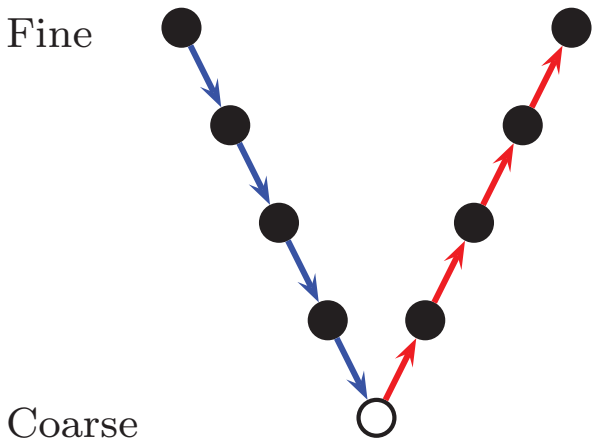
Multilevel Iterative Methods

Examples of multilevel algorithms

- Quick Sort, FFT, FMM, GMG, AMG, H-Matrix, H^2-Matrix, ...

Multigrid V-cycle





Key gradients for multilevel iterative methods

- Construct multilevel hierarchy in an efficient way
- Find effective (and cheap) smoothers for each level
- Find good coarser level solvers (nested iterations)

Multigrid Method

Performance comparison: GMG vs AMG

Solution Method	FMG	GMG-PCG	CA-PCG	UA-PCG
Number of Iterations	—	5	6	12
Wall Time (sec)	0.143	0.251	1.57 (0.87)	1.50 (0.26)

Table: Solving 2D Poisson's equation using multigrid methods (Five-point stencil, FASP 1.8.3, DOF = 1M, TOL = 10^{-6}, Macbook Pro 13', gcc-4.9.3, -O2)

Methods based on PDE and/or discretization information

- Using connectivity information from coefficient matrix (AMG)
- Using an extended matrix (Jacobi = BPX, GS = MG V-cycle)
- Using an auxiliary grid or discretization
- Using coarsening based on the finest grid
- Block preconditioners for coupled PDEs

Must plan ahead of time: meshing, linearization, discretization, ...

Method of Subspace Corrections

Divide and conquer

- Space decomposition: $V = \sum_{i=1}^{n} V_i$
- Subspace correction: $e_i \approx A_i^{-1} P_i (f - Au)$

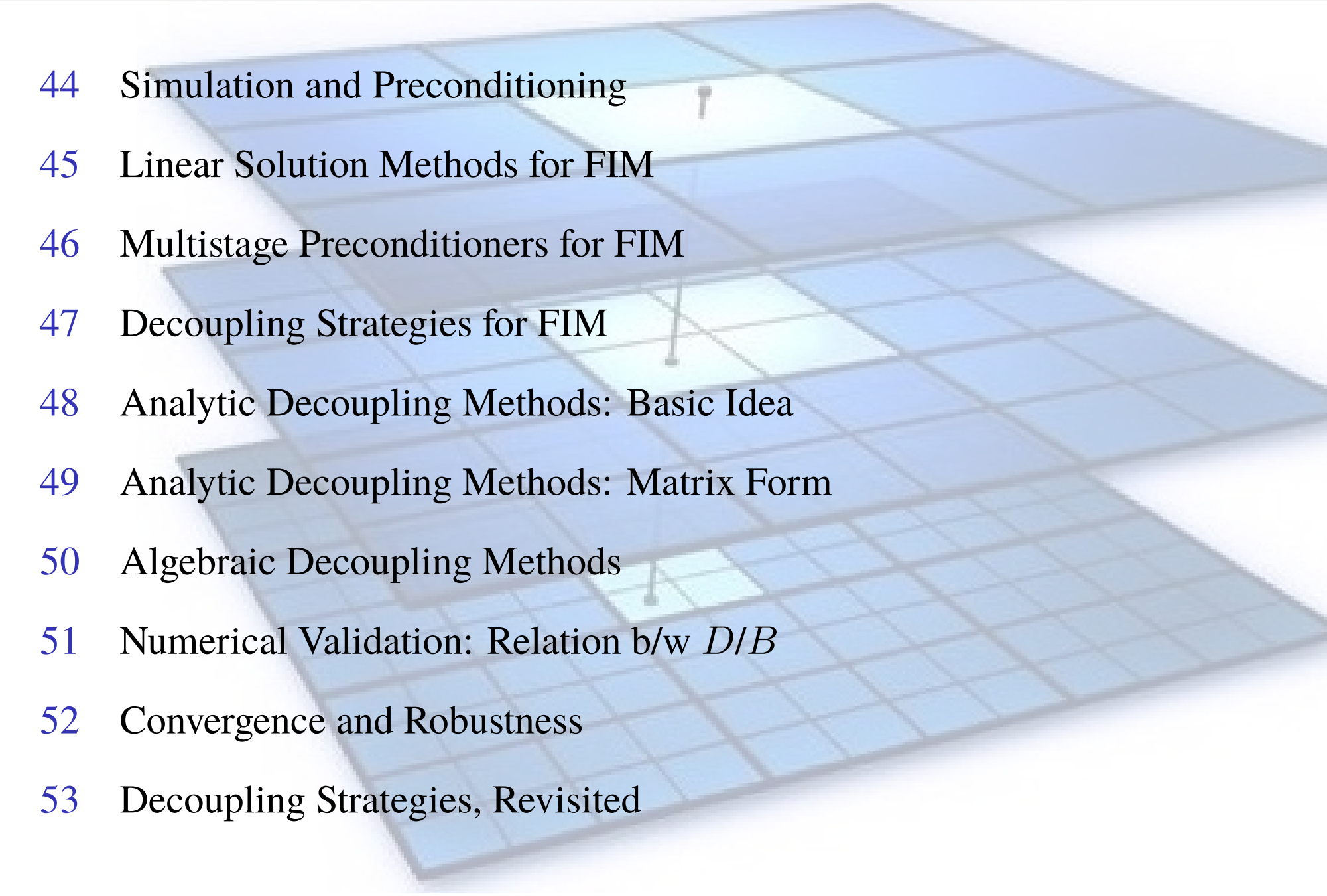
$$u \leftarrow u + \sum_{i=1}^{n} e_i \qquad \text{(Parallel subspace corrections, Jacobi)}$$

$$u \leftarrow u + e_i, \ i = 1 : n \qquad \text{(Successive subspace corrections, GS)}$$

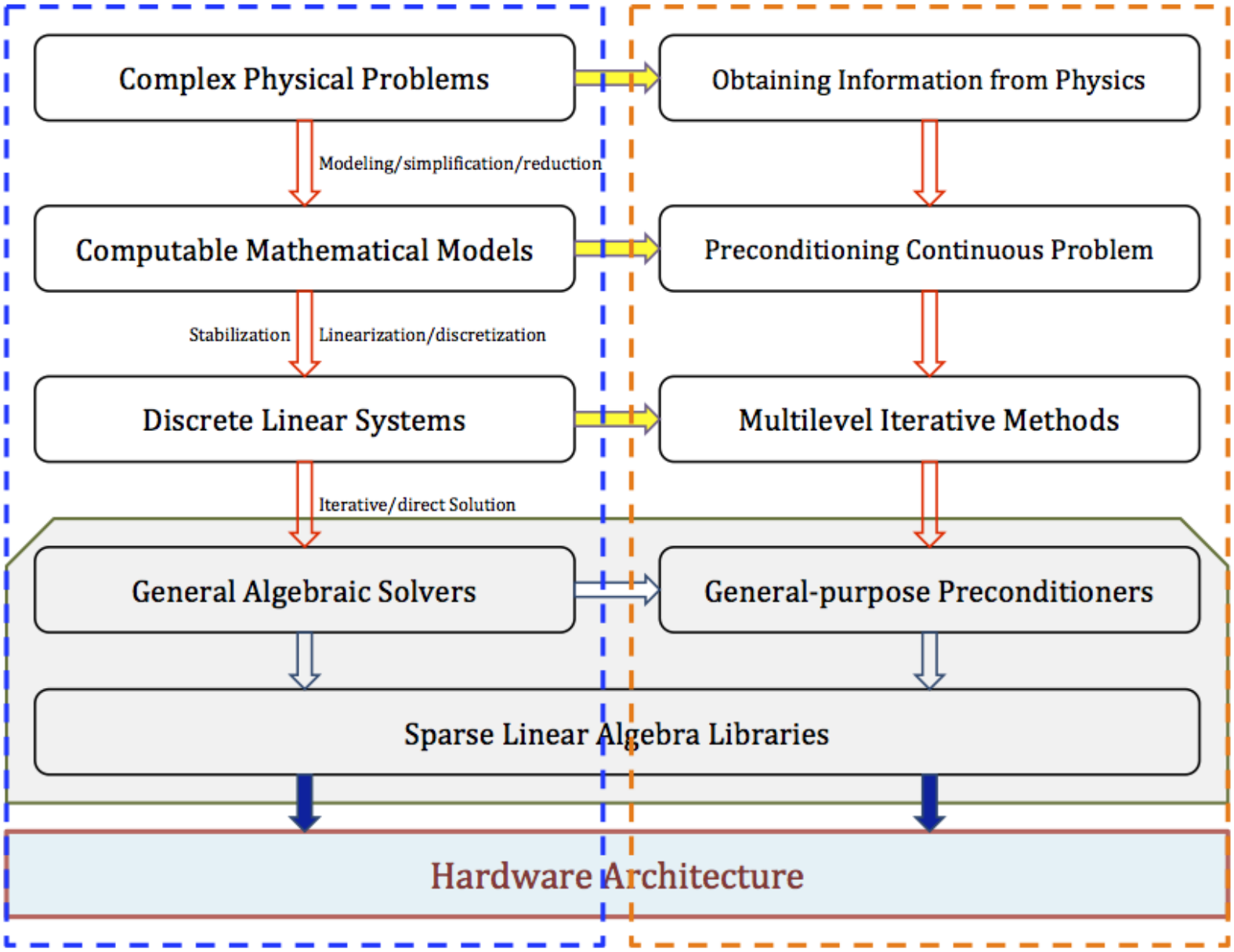
Some examples and generalizations

- BPX preconditioner [Bramble-Pasciak-Xu 1990]
- SIAM Review [Xu 1992]
- Fictitious domain method [Nepomnyaschikh 1992]
- Auxiliary space method [Xu 1996]
- Nonlinear equations [Tai-Xu 2002]
- $H(\mathrm{div})$, $H(\mathrm{curl})$ solvers [Hiptmair-Xu 2007]

§5. Decoupling and Preconditioning



Simulation and Preconditioning



Linear Solution Methods for FIM

Solve the Jacobian system:

$$J\,u = r$$

1. Decoupling Step:

$$\tilde{J}\,u = \tilde{r}$$

where

$$\tilde{J} := DJ \quad \text{and} \quad \tilde{r} := Dr$$

2. Solution Step: Solve the preprocessed linear equation by a Krylov space method (e.g. GMRES or BiCGstab) with a multi-stage preconditioner

Difficulties in solving the Jacobian system:

- Fully-coupled, large, non-symmetric, ill-conditioned
- Usually takes more than 80% of the computing time

Now we consider linear algebraic solvers for the FIM discretization!

Multistage Preconditioners for FIM

Define subspaces:

$$V = V_P + V_N$$

A two-stage preconditioner: Given u_0, $Bu_0 := u_2$, where

$$u_1 = u_0 + \Pi_P \tilde{J}_{PP}^{-1} \Pi_P^* (\tilde{r} - \tilde{J} u_0)$$
$$u_2 = u_1 + \Pi_N \tilde{J}_{NN}^{-1} \Pi_N^* (\tilde{r} - \tilde{J} u_1)$$

- Form subspaces according to physical properties
- Choose appropriate solvers for each subspace
- Example: CPR-type preconditioners [Wallis 1983]

- A decoupling stage is necessary before the solution stage
- Decouple different unknowns (P and N) effectively
- Obtain a reasonable pressure equation $\tilde{J}_{PP}$
- How to choose the decoupling (D) and preconditioning (B)?

[Hu, Xu, Zhang, 2013]

Decoupling Strategies for FIM

Formal (semi-discrete) Jacobian matrix

$$J = \frac{1}{\Delta t}\begin{bmatrix} V_P^{\text{pore}} - V_P^{\text{fluid}} & -V_1^{\text{fluid}} & \cdots & -V_{n_c}^{\text{fluid}} \\ 0 & 1 & & \\ \vdots & & \ddots & \\ 0 & & & 1 \end{bmatrix}$$

$$+ \begin{bmatrix} 0 & 0 & \cdots & 0 \\ -\nabla\cdot(T_1\nabla\circ) - \nabla\cdot(\vec{\beta}_{1P}\circ) & -\nabla\cdot(\vec{\beta}_{11}\circ) & \cdots & -\nabla\cdot(\vec{\beta}_{1n_c}\circ) \\ \vdots & \vdots & \ddots & \vdots \\ -\nabla\cdot(T_{n_c}\nabla\circ) - \nabla\cdot(\vec{\beta}_{n_cP}\circ) & -\nabla\cdot(\vec{\beta}_{n_c1}\circ) & \cdots & -\nabla\cdot(\vec{\beta}_{n_cn_c}\circ) \end{bmatrix}$$

Decoupling methods [Lacroix, Vassilevski, Wheeler, 2001; ...]

$$\tilde{J} = DJ = \begin{bmatrix} \tilde{J}_{PP} & \tilde{J}_{PN} \\ \tilde{J}_{NP} & \tilde{J}_{NN} \end{bmatrix} \quad \text{as a pre-processor}$$

- Cheap to apply and give an easy-to-solve pressure equation
- Make $\tilde{J}_{PN}$ (sometimes $\tilde{J}_{NP}$ as well) not dominant

☞ Limiting behavior: $I - B\tilde{J}$ reduces to 0 as $\Delta t \to 0$, which is invalid for J

Analytic Decoupling Methods: Basic Idea

Decoupling at the PDE level:

$$\tilde{J}_{ANL} = \frac{1}{\Delta t}\begin{pmatrix} \alpha_P & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 1 \end{pmatrix}$$

$$+\begin{pmatrix} -\nabla\cdot(\tilde{T}\nabla\circ) + \vec{\beta}_1\cdot(\nabla\circ) - \sum_{i=1}^{n_c} V_{ti}\nabla\cdot(\vec{\beta}_{iP}\circ) & -\sum_i V_{ti}\nabla\cdot(\vec{\beta}_{i1}\circ) & \dots & -\sum_i V_{ti}\nabla\cdot(\vec{\beta}_{in_c}\circ) \\ -\nabla\cdot(T_1\nabla\circ) - \nabla\cdot(\vec{\beta}_{1P}\circ) & -\nabla\cdot(\vec{\beta}_{11}\circ) & \dots & -\nabla\cdot(\vec{\beta}_{1n_c}\circ) \\ \vdots & \vdots & \ddots & \vdots \\ -\nabla\cdot(T_{n_c}\nabla\circ) - \nabla\cdot(\vec{\beta}_{n_cP}\circ) & -\nabla\cdot(\vec{\beta}_{n_c1}\circ) & \dots & -\nabla\cdot(\vec{\beta}_{n_cn_c}\circ) \end{pmatrix}$$

where $\alpha_P, \vec{\beta}_1, \vec{\beta}_{ik}, \vec{\beta}_{iP}$ are knowns.

- We know the underlying equations we are solving
- A multigrid friendly system can be formed
- Becomes diagonally dominant as Δt goes to 0

Analytic Decoupling Methods: Matrix Form

Decoupling in matrix form:

Consider the decomposition $J = A + F$. Let

$$A := \frac{1}{\Delta t}\begin{bmatrix} A_{PP} & A_{PN} \\ A_{NP} & A_{NN} \end{bmatrix} \quad \text{and} \quad D^{\text{ANL}} := \begin{bmatrix} I & X \\ 0 & I \end{bmatrix}$$

such that

$$D^{\text{ANL}}A = \frac{1}{\Delta t}\begin{bmatrix} \tilde{A}_{PP} & 0 \\ A_{NP} & A_{NN} \end{bmatrix} \Longrightarrow \tilde{J} = D^{\text{ANL}}J = \frac{1}{\Delta t}\begin{bmatrix} V_P^{\text{pore}} - V_P^{\text{fluid}} & 0 \\ 0 & I \end{bmatrix} + \cdots$$

General comments and advantages

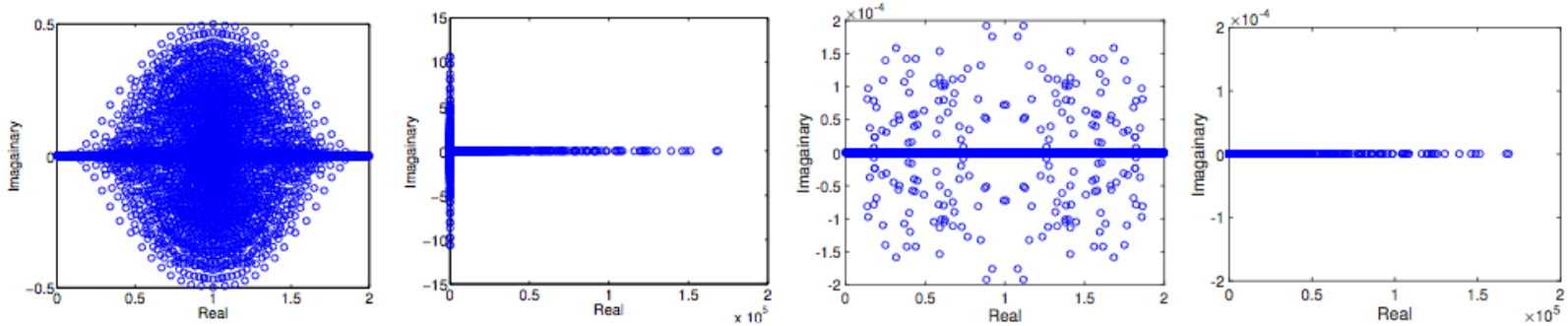
- ☞ Note that our A is in a very special form
  - Closely related to the IMPES discretization (eliminate N-terms)
  - Black oil model $\Longrightarrow$ True-IMPES decoupling method [Coats 1999]
  - Giving "good" pressure equations that work well with multigrid
  - We have $I - B\tilde{J} \to 0$ as $\Delta t \to 0$

Algebraic Decoupling Methods

- Alternate Block Factorization [Bank-Chan-Coughran-Smith 1989; Klie 1997]:

$$D^{\text{ABF}} := \begin{bmatrix} \text{diag}(J_{PP}) & \text{diag}(J_{PN}) \\ \text{diag}(J_{NP}) & \text{diag}(J_{NN}) \end{bmatrix}^{-1}$$

☞ Eigenvalues clustered around 1, but the pressure equations difficult to solve



(a) $\tilde{J}^{\text{ABF}}$ (b) $\tilde{J}^{\text{ANL}}$ (c) $\tilde{J}^{\text{ABF}}_{PP}$ (d) $\tilde{J}^{\text{ANL}}_{PP}$

- There are several algebraic decoupling methods (Householder, Quasi-IMPES, CPR, ...) that are equivalent to ABF up to a scaling
- We have $I - B\tilde{J} \to 0$ as $\Delta t \to 0$, which does not hold for J
- More stable and take less iterations if the pressure is approximated well

[Qiao, Wu, Xu, Zhang, 2017]

Numerical Validation: Relation b/w D/B

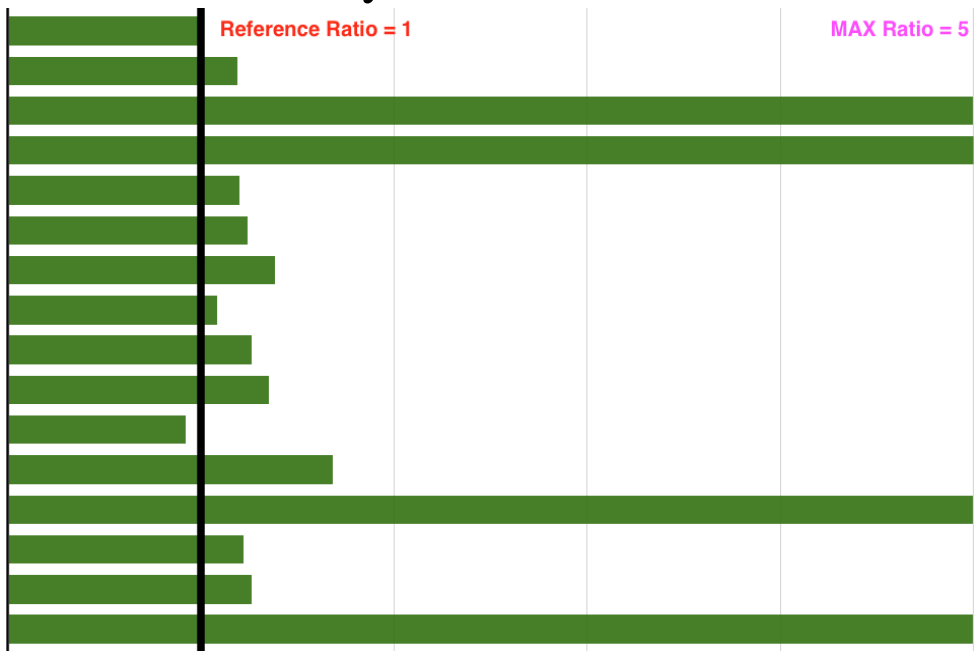
Pressure equations from ABF are difficult for AMG $\Longrightarrow$ Different solvers

Comparison of two preconditioners

- ☞ Method-I: Use one AMG V-cycle as a pressure solver
- ☞ Method-Ⅱ: Use AMG preconditioned GMRES as a pressure solver

No	Model	DAYS	GRID
1	Black-oil	900	9026
2	Black-oil	900	900026
3	Black-oil	15096	241474
4	Black-oil	15096	241471
5	Black-oil	15616	466913
6	Black-oil	10653	143786
7	Black-oil	9100	46825
8	Black-oil	11868	46574
9	Two-phase	5233	45156
10	Two-phase	4408	208842
11	Two-phase	21427	89339
12	Two-phase	2000	1094422
13	Two-phase	2000	10944220
14	Two-phase	19753	89048
15	Two-phase	708	51623
16	Two-phase	1825	104013

Wall time by Method-I / Method-Ⅱ



[Li, Wu, Zhang, et al., 2017; Li, Thesis, 2017]

Convergence and Robustness

No	Name	Properties				Ecl100		HiSim	
		Model	# Total Cells	# Active Cells	Peroid (day)	Newton	Time (min)	Newton	Time (min)
1	SPE10-2	Two-phase	1122000	1094422	2000	——	——	295	41.82
2	SPE9-9k	Black-oil	9000	9000	900	339	0.12	269	0.20
3	SPE1	CO2 flooding	300	300	3656	536	0.04	445	0.08
4	SPE2	Three-phase coning	150	150	900	209	0.01	538	0.14
5	SPE10-3	Black-oil	1122000	1094422	2000	——	——	1462	354.12
6	SPE6	Dual porosity	100	100	7300	306	0.01	322	0.02
7	DPSP	Dual porosity	60984	40294	360	545	2.64	116	0.81
8	SPE7	Horizontal wells	488	488	1500	120	0.01	75	0.02
9	Voliatle	Extended black-oil	2100	2100	0.694			67	0.03
10	Zaoyuan	Field test (black-oil)	417480	143786	10653	3302	105.49	5204	66.20
11	Jidong	Field test (black-oil)	335664	154598	10587	1091	139.69	161	4.41
12	Chengbei	Field test (black-oil)	1646500	585123	2191	1971	155.57	420	28.47
13	Daqing1	Field test (black-oil)	1453248	466913	15616	——	——	5227	338.00
14	Daqing2	Field test (black-oil)	847895	241474	15096	8562	92.46	3072	88.05
15	SPE10-10M	Two-phase (large-scale)	11220000	10944220	2000	——	——	592	962.12
16	SPE9-9M	Black-oil (large-scale)	9000000	9000000	900	——	——	2460	10932.81

Tested by the Research Institute of Petroleum Exploration and Development, PetroChina (2015): Dell E5-2690 v2 CPU@3.0GHz, 200GB DDR3, Windows 7/VS2010/Intel Fortran Compiler 2015, HiSim 2.0, ECL 2012

Decoupling Strategies, Revisited

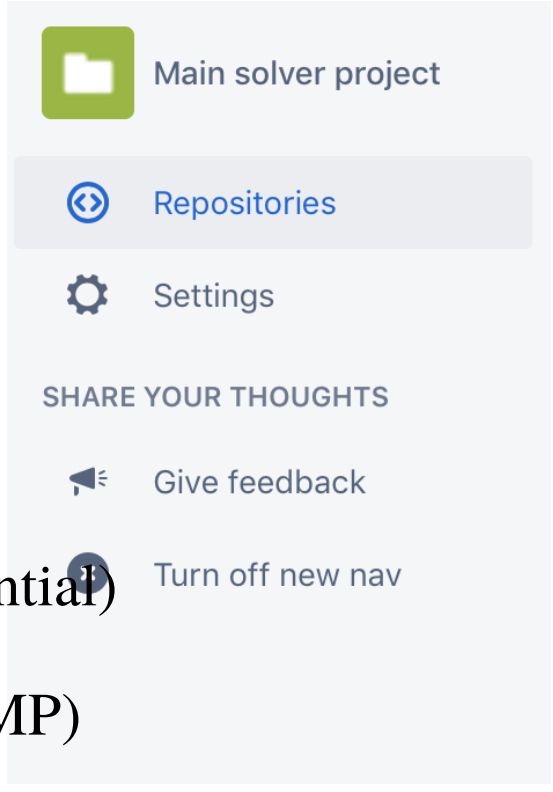
- Can we combine advantages of analytical and algebraic decouplings?
  - Cheap to compute; do no spoil outer iterations
  - Obtain an easy-to-solve pressure equation
  - Keep the asymptotic behavior $I - B\tilde{J} \to 0$ as $\Delta t \to 0$
- A semi-analytical decoupling method: [Qiao, Wu, Xu, Zhang, 2017]

$$D^{\text{SEM}} := \begin{bmatrix} D_{PP}^{\text{ANL}} & D_{PN}^{\text{ANL}} \\ D_{NP}^{\text{ABF}} & D_{NN}^{\text{ABF}} \end{bmatrix}$$

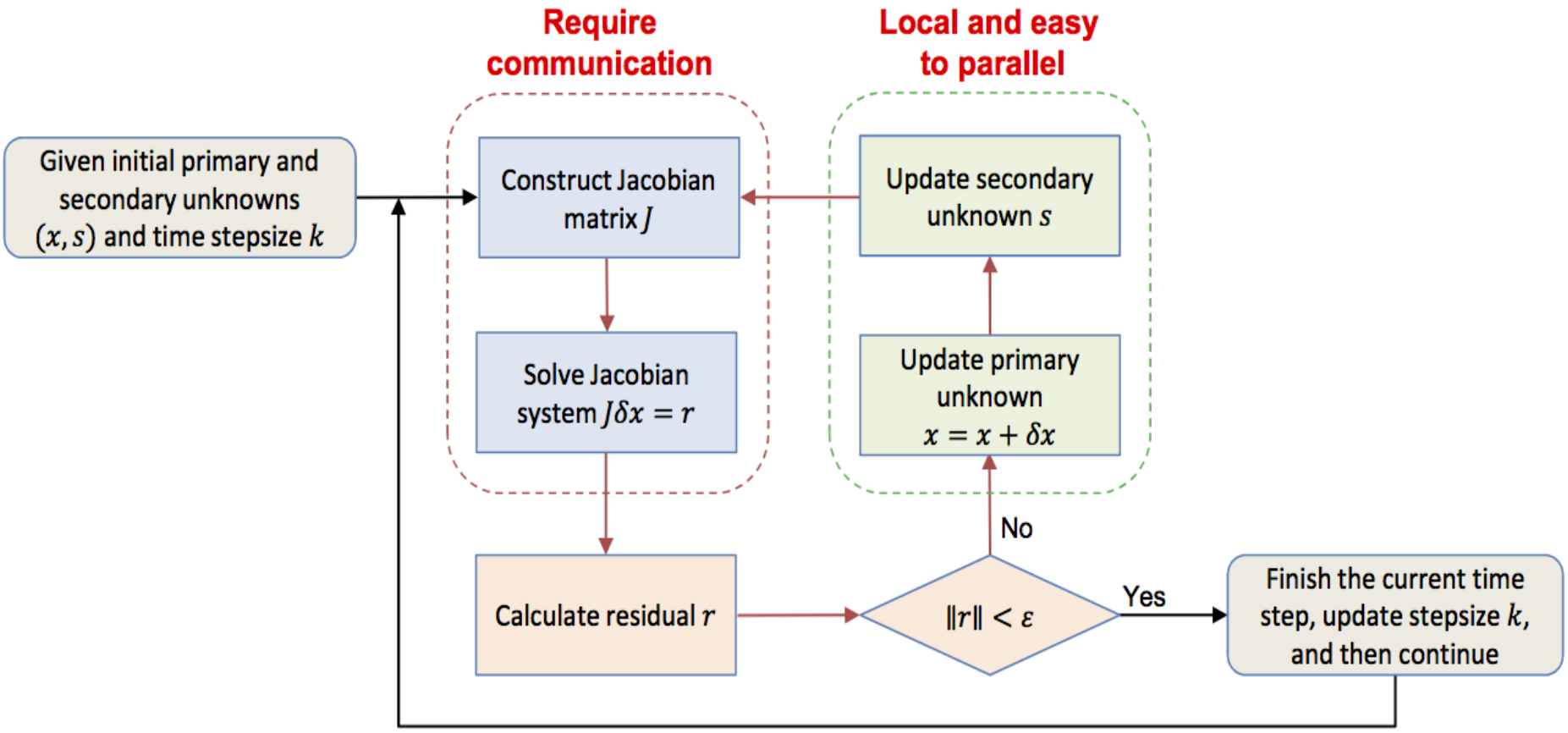
Test Problem	Matrix1			Matrix2		
Decoupling method	ABF	ANL	SEM	ABF	ANL	SEM
Number of iterations	3	3	3	7	7	7
Total AMG iterations	5	6	6	97	27	32
Average AMG iterations	1.67	2	2	13.86	3.86	4.57

Test Problem	Matrix3			Matrix4		
Decoupling method	ABF	ANL	SEM	ABF	ANL	SEM
Number of iterations	7	8	6	6	6	4
Total AMG iterations	111	32	25	73	25	18
Average AMG iterations	15.86	4	4.17	12.17	4.16	4.5

§6. Simulation Software



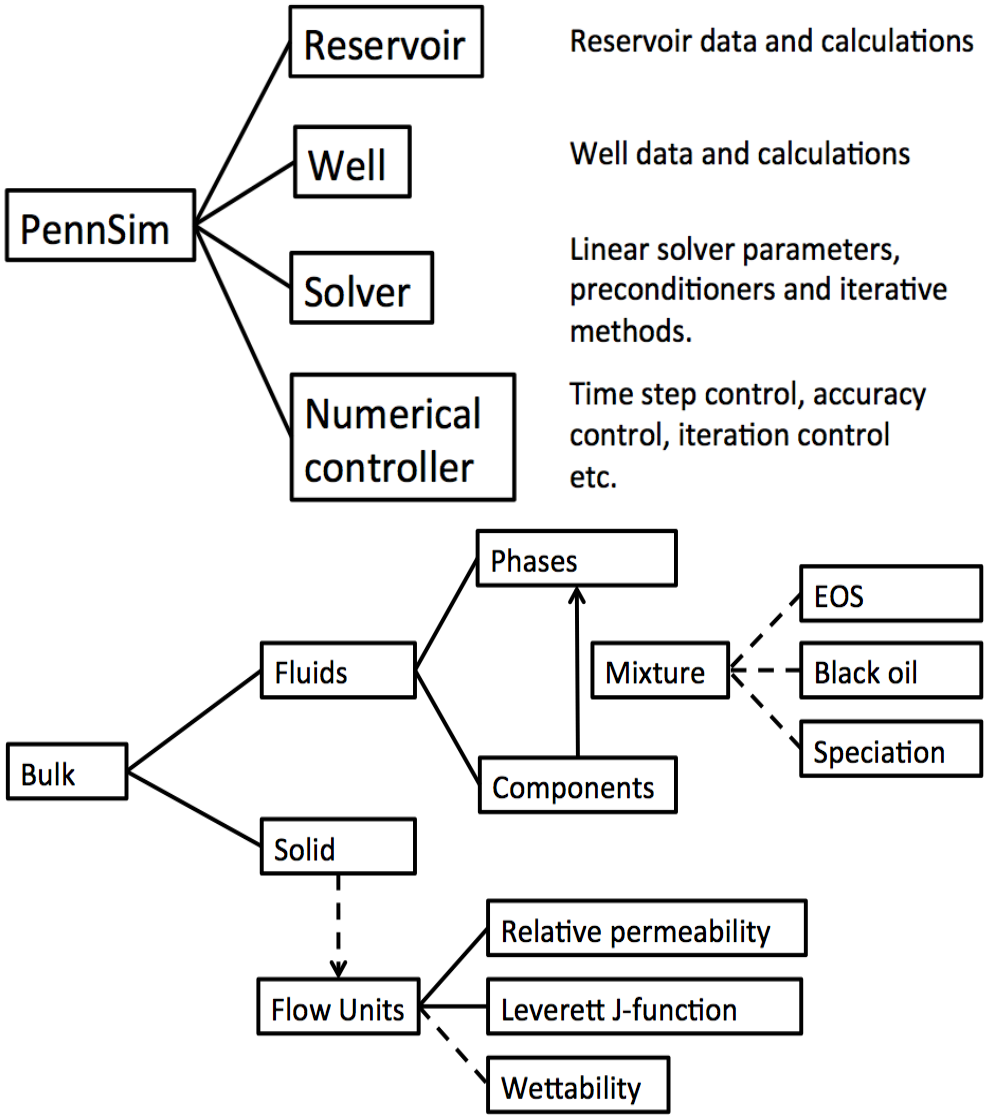
Solution Algorithm Flow Chart

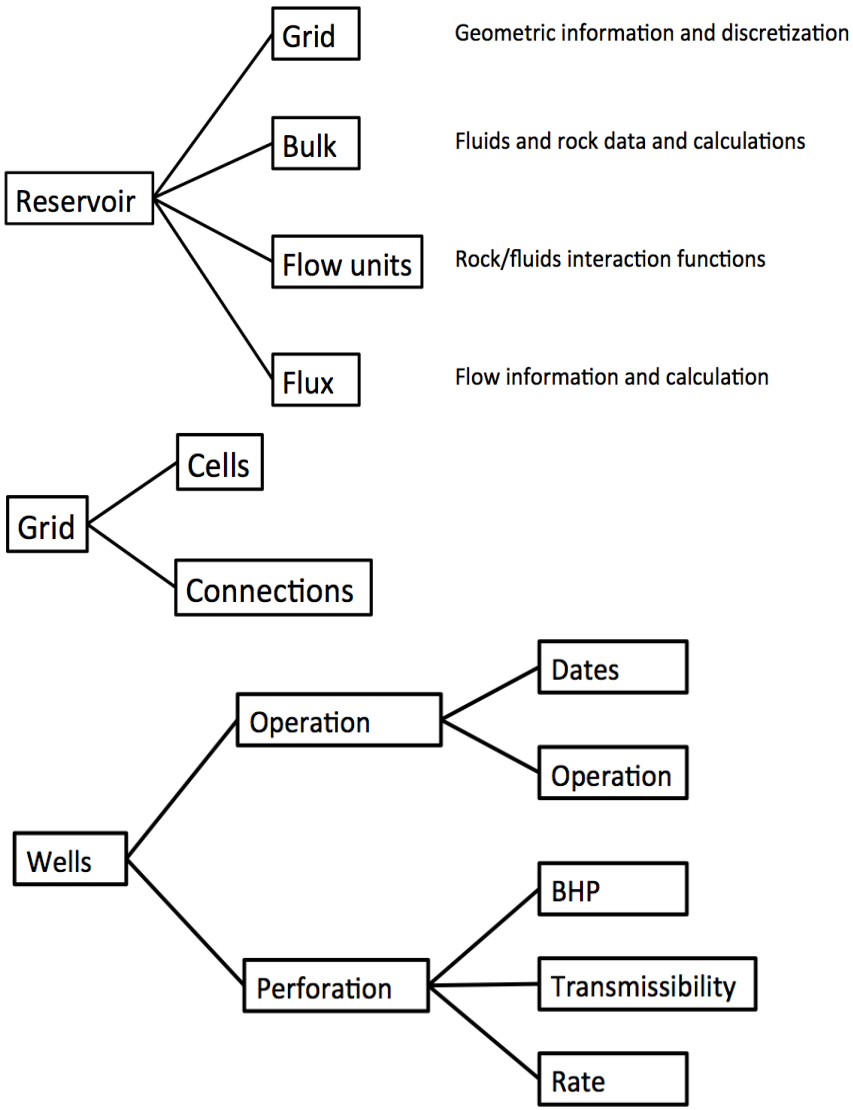


Need a scalable parallel linear algebraic solver to make it work!

[Guan, Qiao, Zhang, et al. 2015]

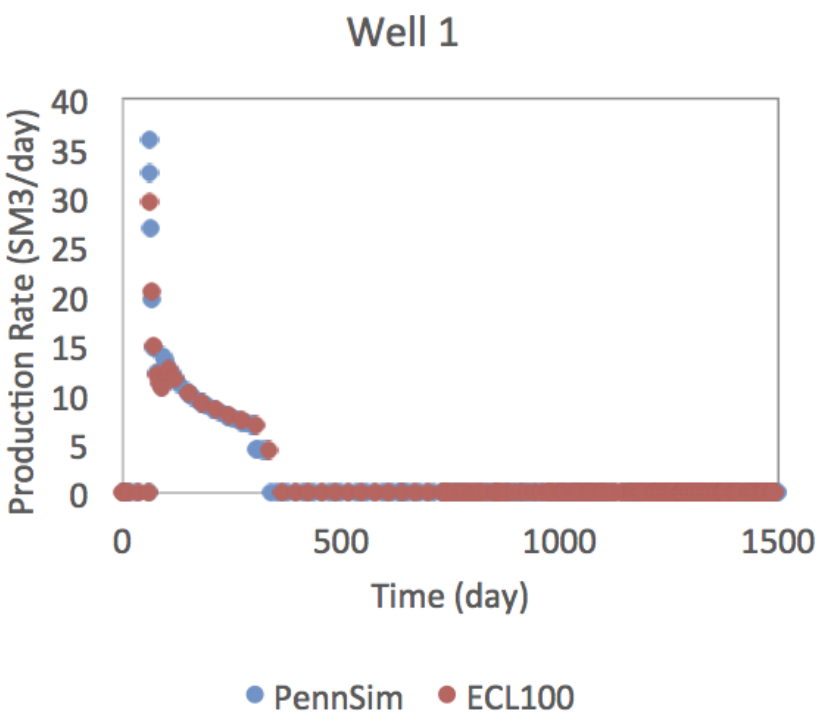
Introduction to PennSim

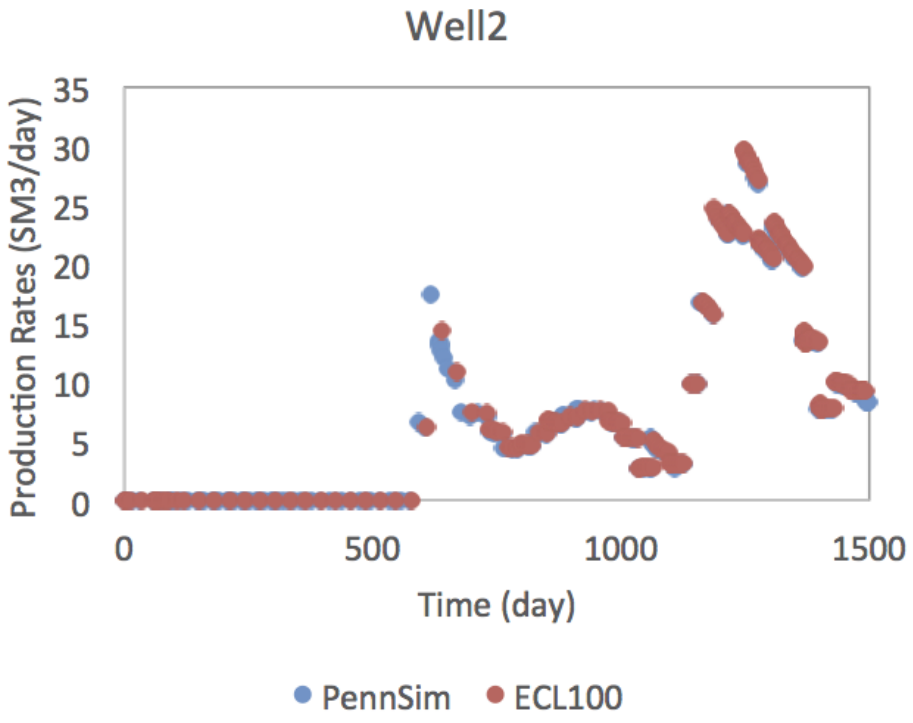




[Qiao, Thesis, 2016]

Comparison with ECL100





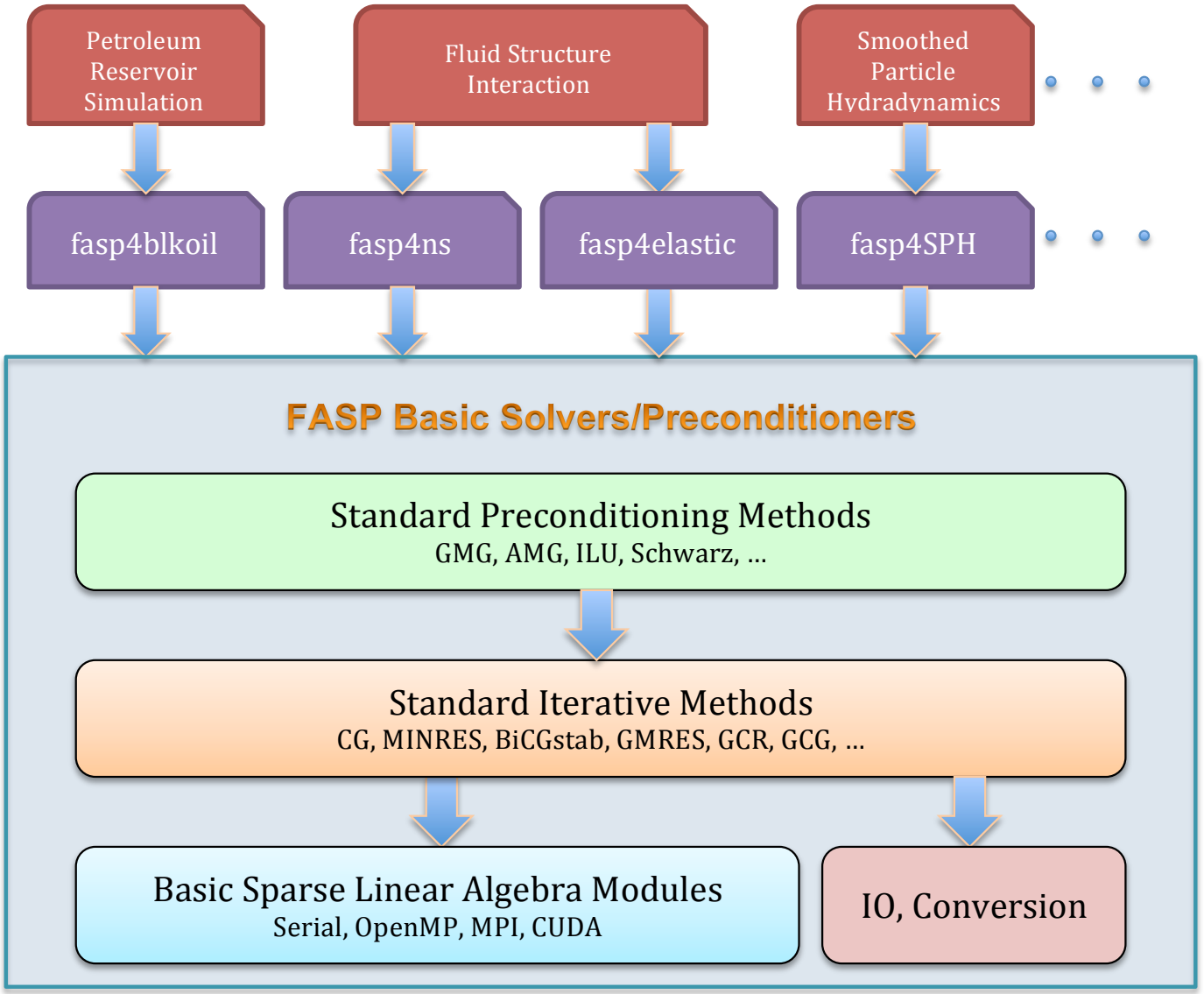
- Real data from an European field (60K corner-point grid)
- Qualitatively matches the results of commercial software
- Simulate five-year period (PennSim≈3hr, ECL100≈6hr)
- Cost only half of the CPU time

Multilevel Solver Software

1. How to handle a complicated PDE (system)?
   - Provide blockwise iterative methods and general preconditioners
   - Use mapping properties to construct a good preconditioner
   - Reduction: precond PDE systems $\Longrightarrow$ precond model problems
   - Use an auxiliary problem for preconditioning
2. How to choose a discretization?
   - Using a uniformly stable discretization is important
   - Using a solver-friendly discretization
3. How to handle discretizations on unstructured mesh?
   - To improve efficiency of the SETUP phase of multilevel methods
   - Use sparsity pattern or entries of coefficient matrices
   - Use an auxiliary structured (or semi-structured) grid

Need to plan ahead of time!

FASP Software Project



Supported by NSF DMS-0915153 and NSFC 91130011. http://fasp.sf.net

Preliminary Tests: AMG (Sequential)

Test Device: Intel Core i5 2.6GHz, 8GB RAM, gcc 4.9.2 -O2
Benchmark: FASP 1.7.0, hypre 2.10.0b, AGMG 3.2.0 (default parameters)

Problem	DOF	RS-V-CG	UA-NA-CG	hypre	AGMG
2D 5pt	1M	1.79	1.43	1.96	1.73
2D 5pt	4M	8.71	6.04	8.45	6.61
2D 9pt	1M	1.82	2.07	2.25	2.24
2D 9pt	4M	7.63	8.39	8.88	9.42
3D 7pt	$\frac{1}{4}$M	1.05	0.37	1.83	0.43
3D 7pt	2M	10.86	3.28	19.04	3.71
3D 27pt	$\frac{1}{4}$M	2.09	0.94	3.26	1.79
3D 27pt	2M	20.0	8.53	34.54	20.29

Table: Computing time (seconds) of the AMG-preconditioned conjugate gradient method. We solve the 2D/3D Poisson equation with one processing core. Stopping criteria: relative residual is less than 10^{-6}.

Preliminary Tests: AMG (OpenMP)

Test Device: Intel Xeon X5675 3.07GHz (6 cores), 24GB RAM, gcc 4.4.6

–O2	DOF	NT=1	NT=2	NT=4	NT=8	NT=12
2D 5pt	1M	2.12s	×1.43	×1.74	×1.90	×1.89
2D 5pt	4M	9.51s	×1.47	×1.80	×2.00	×2.00
3D 7pt	2M	9.86s	×1.50	×1.89	×2.15	×2.20
3D 7pt	16M	90.66s	×1.50	×1.90	×2.20	×2.29
–O0	DOF	NT=1	NT=2	NT=4	NT=8	NT=12
2D 5pt	1M	6.36s	×1.54	×2.15	×2.60	×2.62
2D 5pt	4M	27.14s	×1.56	×2.22	×2.70	×2.70
3D 7pt	2M	31.00s	×1.59	×2.25	×2.87	×3.07
3D 7pt	16M	274.3s	×1.64	×2.35	×3.01	×3.28

Table: Computing time (seconds) of the classical AMG method. We solve the 2D/3D Poisson equation with OpenMP. Stopping criteria: relative residual is less than 10^{-6}.

Preliminary Tests: GMG (CUDA)

CPU: AMD 2.8GHz 8-core (using a single core), gcc 4.4.6 –O2
GPU: NVIDIA GTX480 480 cores 1.5GB RAM ($485), nvcc 4.1 –O2

DOF	FFTW	FMG(1,2)	CUFFT	FMG(1,2)
1M	0.260	0.108	0.0110	0.0088
4M	2.020	0.452	0.0408	0.0257
16M	6.650	1.830	0.1364	0.0917

Table: Kernel time (seconds) in 2D case

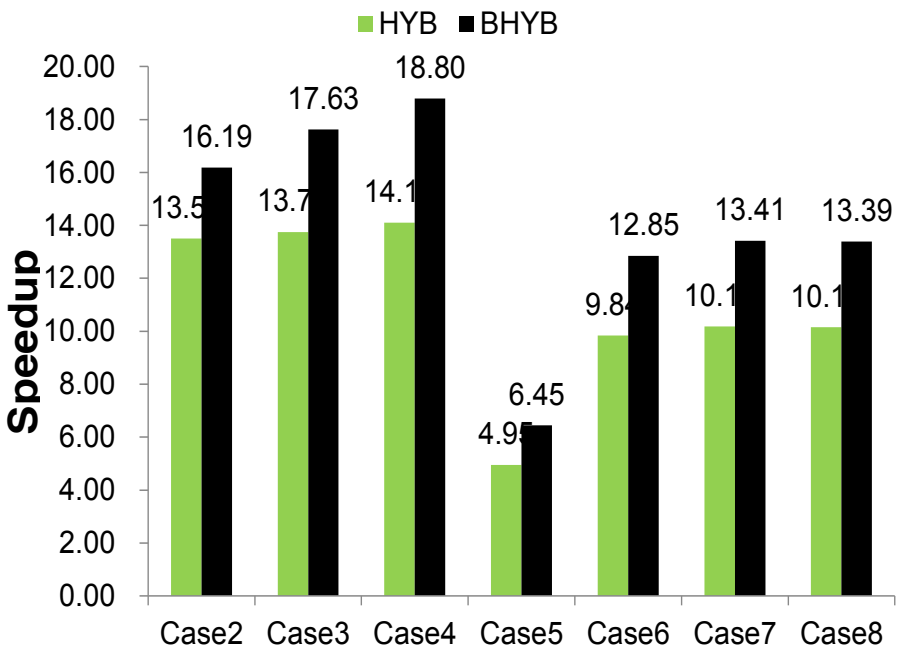
Some observations

- $15\times \sim 18\times$ speed-up compared with single-thread CPU version (2/3D)
- Speedup of GMG is not as good as FFT (almost $50\times$)
- GMG on GPU: 15GFlops, only 10% of peak performance
- Bottleneck: Visiting coarse level spaces
- Solution: BPX + Redundant Basis Formulation

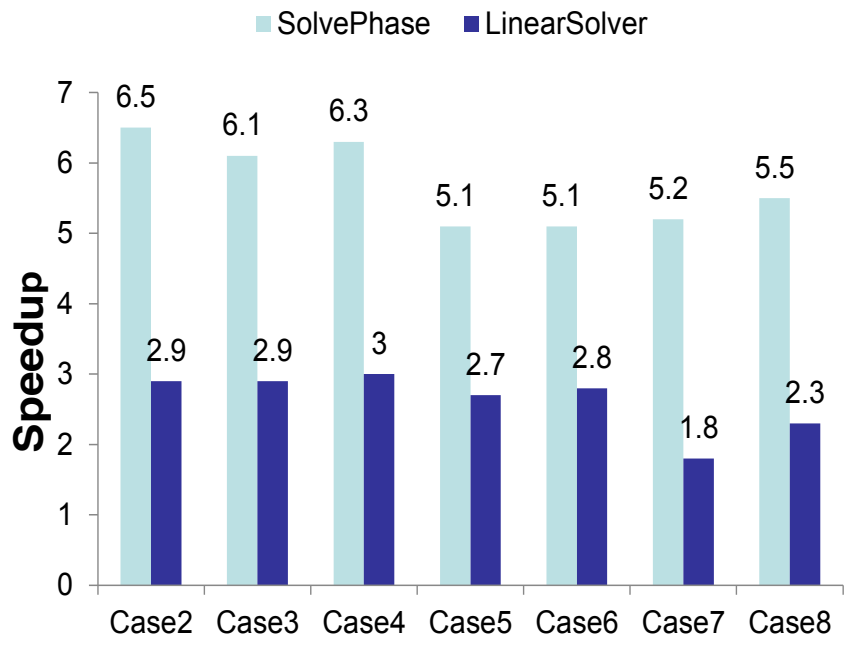
[Feng, Shu, Xu, Zhang, 2014]

Preliminary Tests: SpMV (CUDA)

- Based on heterogenous architecture, we developed a parallel solver for the black-oil model
- Numerical results show reasonable speedup for "easy-to-parallel" parts: Minimizing amount of time for code-tuning



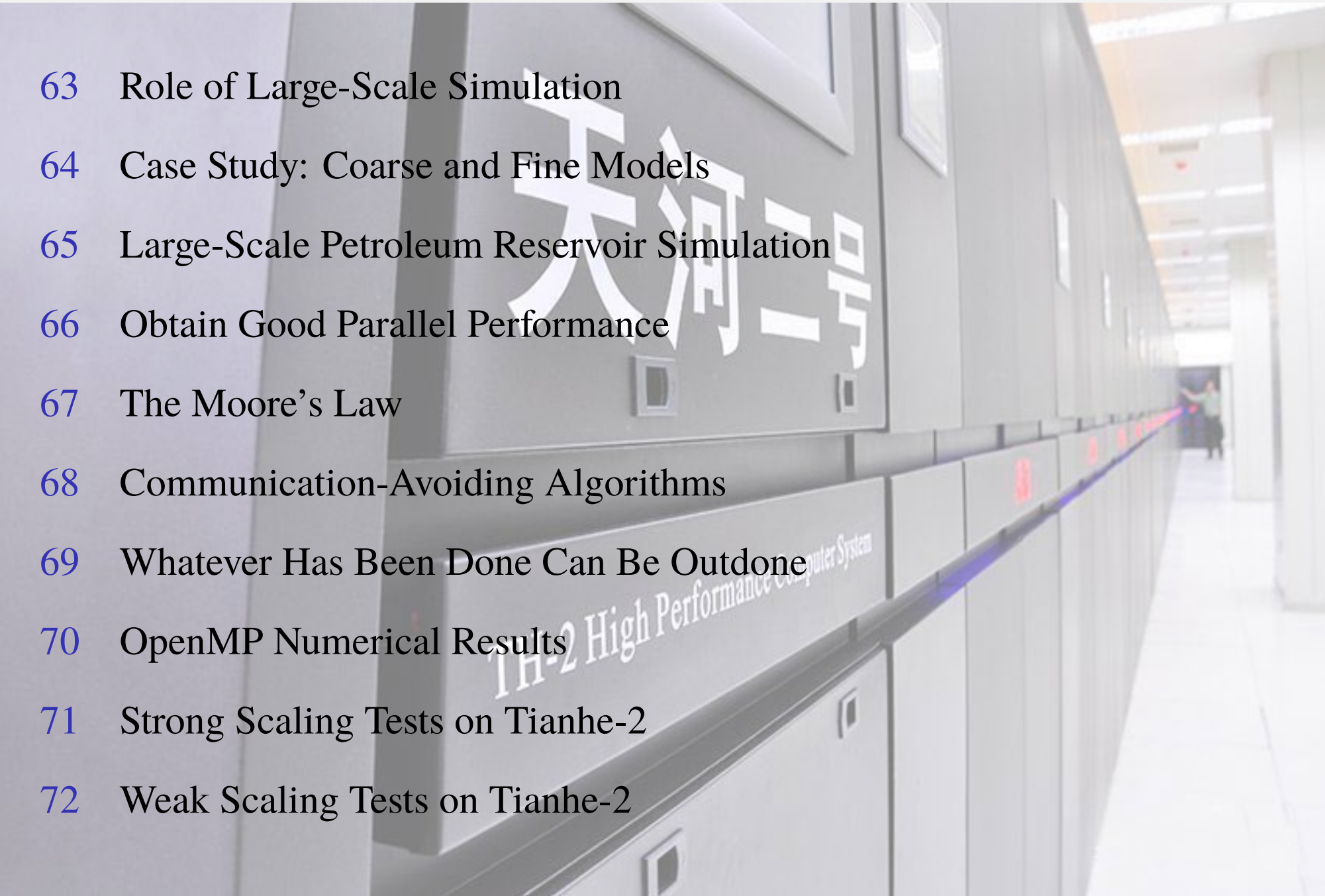
Speedup of SpMV



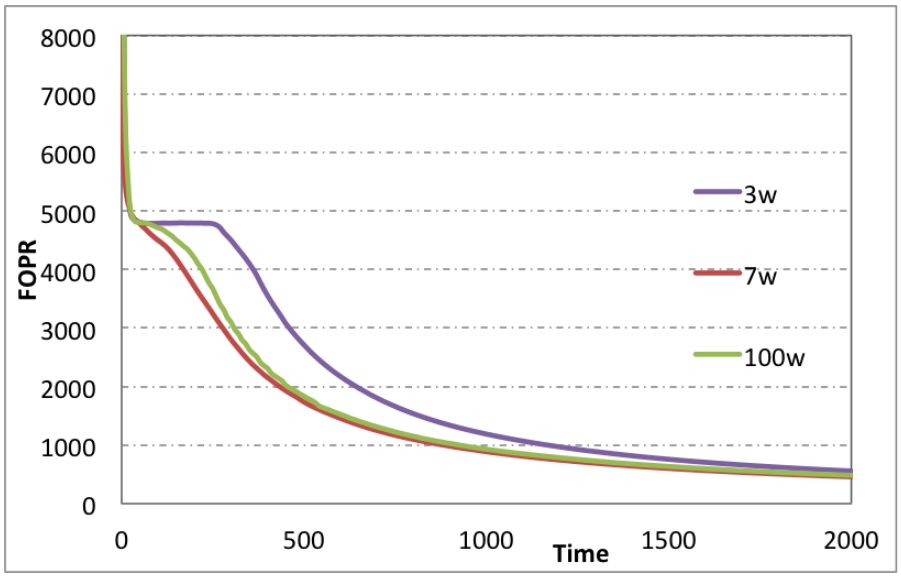
Speedup of linear solver

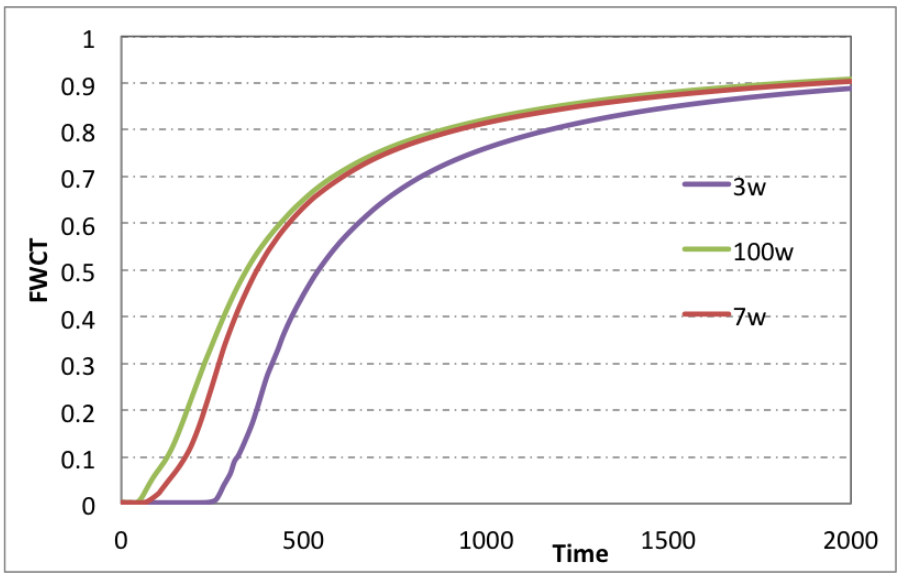
[Li, Wu, Xu, Zhang, 2015]

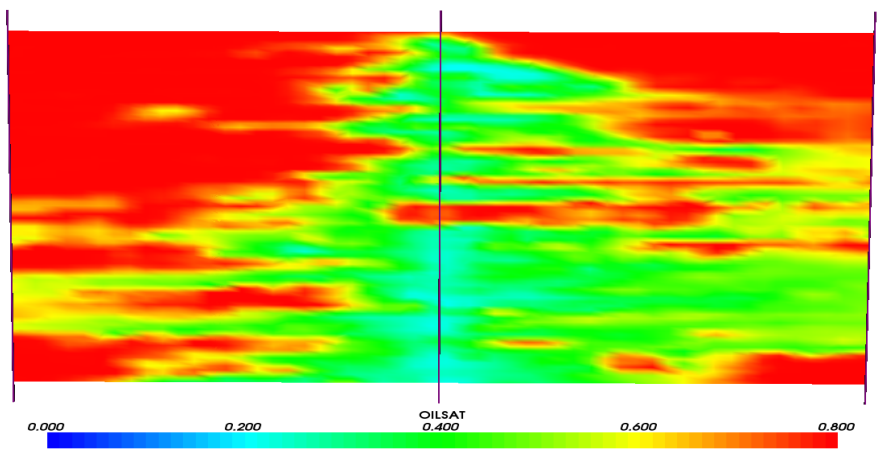
§7. Parallel Implementation

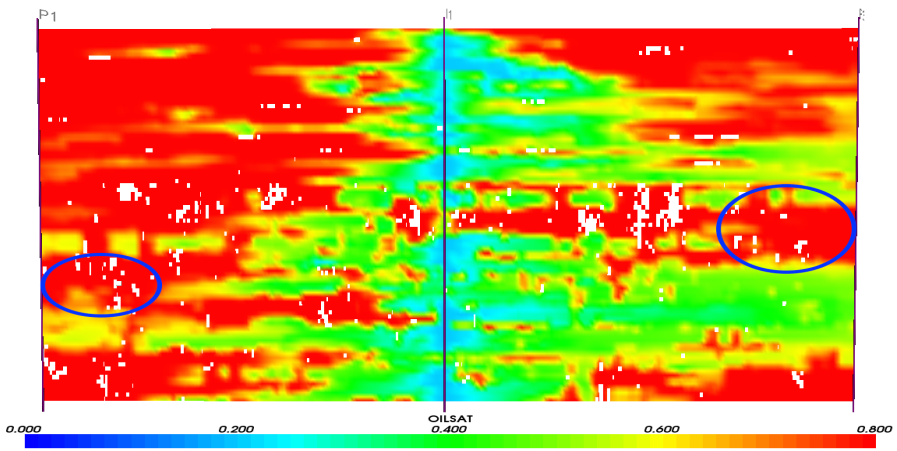


Role of Large-Scale Simulation





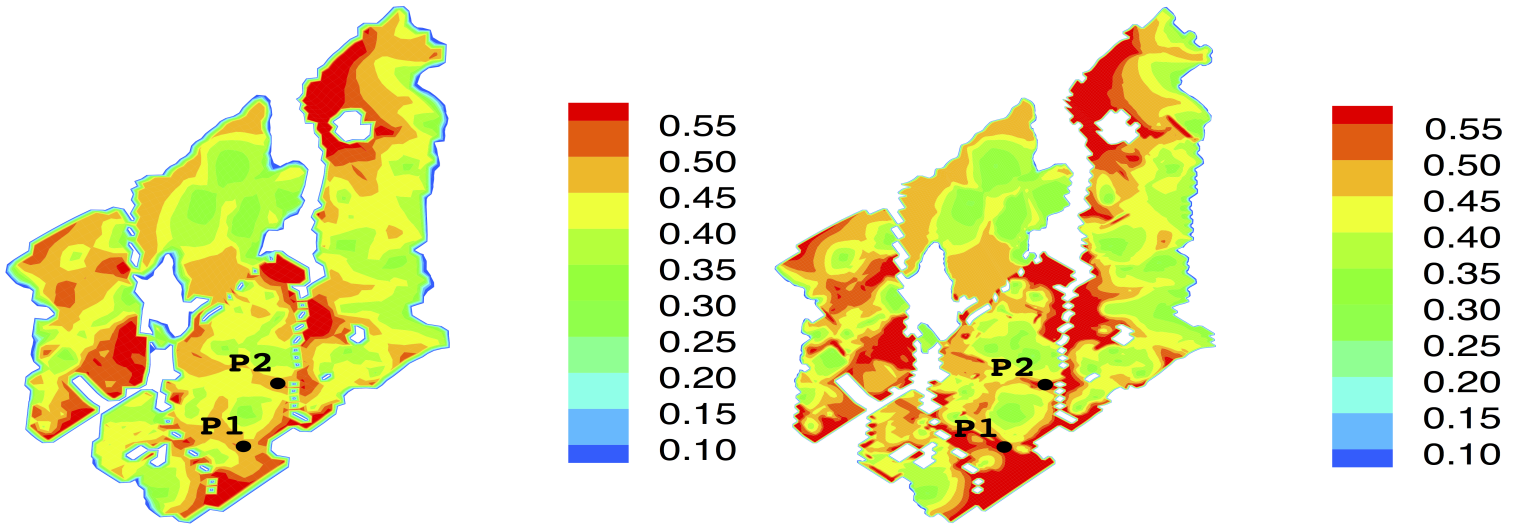


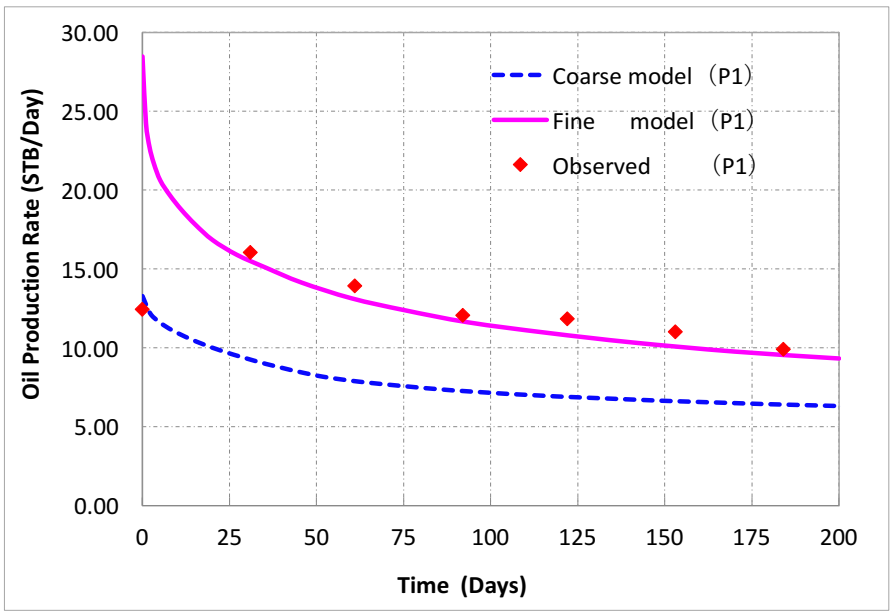


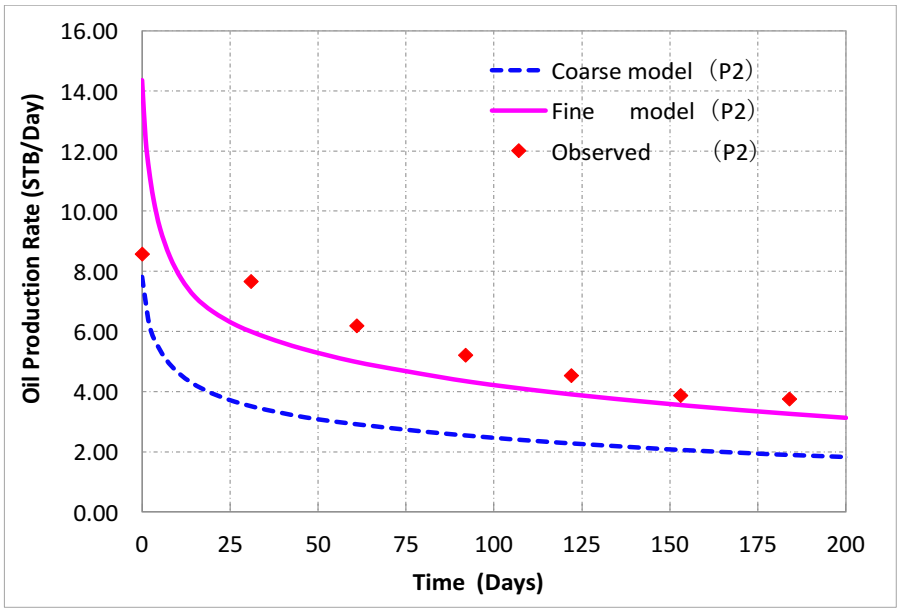
Effect of water injection. Left: 70K grid cells; Right: 1.1M grid cells.

[Wu, Xu, Zhang, et al. 2013]

Case Study: Coarse and Fine Models







[Li, Wu, Li, Zhang, et al. 2016]

Large-Scale Petroleum Reservoir Simulation

Challenges in petroleum reservoir simulation

1. Modeling and discretization
   - Unconventional reservoirs and their modeling
   - Multiscale, heterogeneous, and anisotropic
   - Large number of grid cells with a lot of inactive cells
   - Complicated production requirements and well models
2. Nonlinear and linear solvers
   - Nonlinear algebraic equations for flash calculation
   - Nonlinear coupling between pressure and non-pressure variables
   - Large ill-conditioned linear system to solve
   - Non-symmetric (sometimes indefinite) Jacobian systems for FIM
3. Uncertainty and reliability

Why do we need lager computers for reservoir simulation?

- Need to solve fine-scale problems (1M$\sim$1B grid cells)
- Need to simulate a long period of time (40$\sim$60 years)
- Have many problems to solve ($10^2 \sim 10^3$ repetitions)

Obtain Good Parallel Performance

Keys to Good Parallel Performance

extent of parallelism, granularity of partition, locality of computation

The Amdahl's Law

- The parallel speedup is limited by the time needed for sequential portions

$$\text{speedup}(p) = \frac{\text{time}(1)}{\text{time}(p)} = \frac{\text{time}(1)}{\text{Seq} + \text{Par}/p} \leq \frac{\text{time}(1)}{\text{Seq}}$$

- If 20% of the execution time is sequential, then the max speedup is 5!

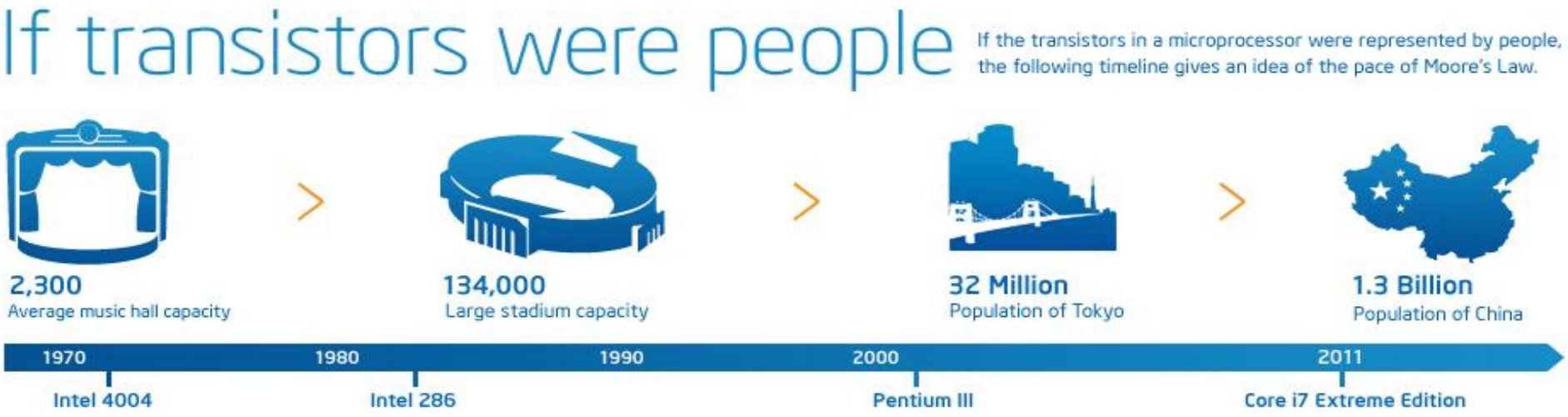
A HPC Paradox (G. Wittum)

- Assumption that algorithm complexity is $E_0 = \mathcal{O}(N^q),\ q \geq 1$
- We want to buy a computer $\alpha \gg 1$ times larger (faster) than the old one
- We wish to solve problems of size α times larger than the original
- The new computer then needs computing time proportional to

$$E_1 = \mathcal{O}(\alpha^q N^q) = \alpha^q E_0 = \alpha^{q-1} \alpha E_0 \geq \alpha E_0$$

- Weak scalability $\implies$ $E_1 = \alpha E_0$ $\implies$ $q = 1$ (optimal algorithm)

The Moore's Law



Challenges to Keep Up With the Moore's Law:

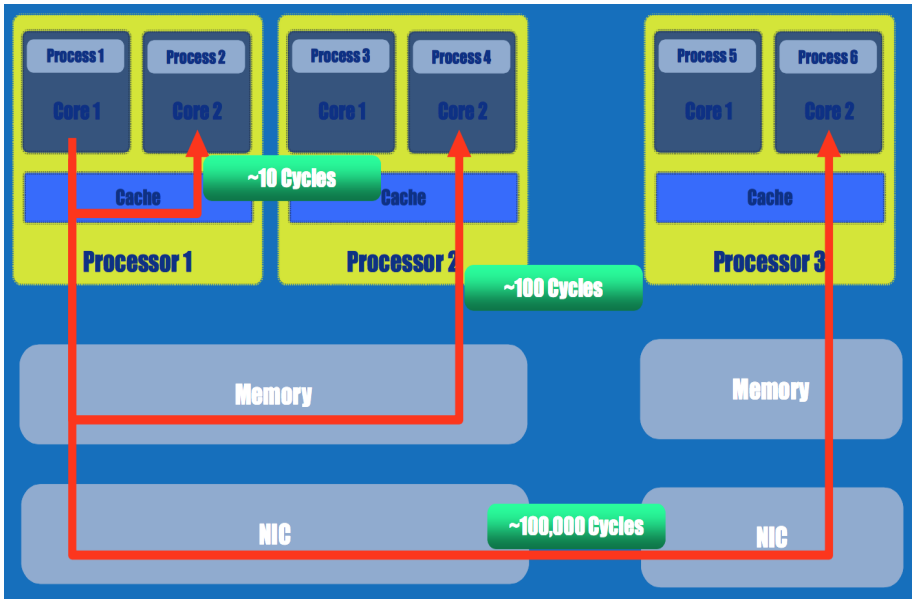
- Instruction-level parallelism (ILP) wall: availability of enough parallel instructions for a multi-core chip
- Power wall: the chip's overall temperature and power consumption

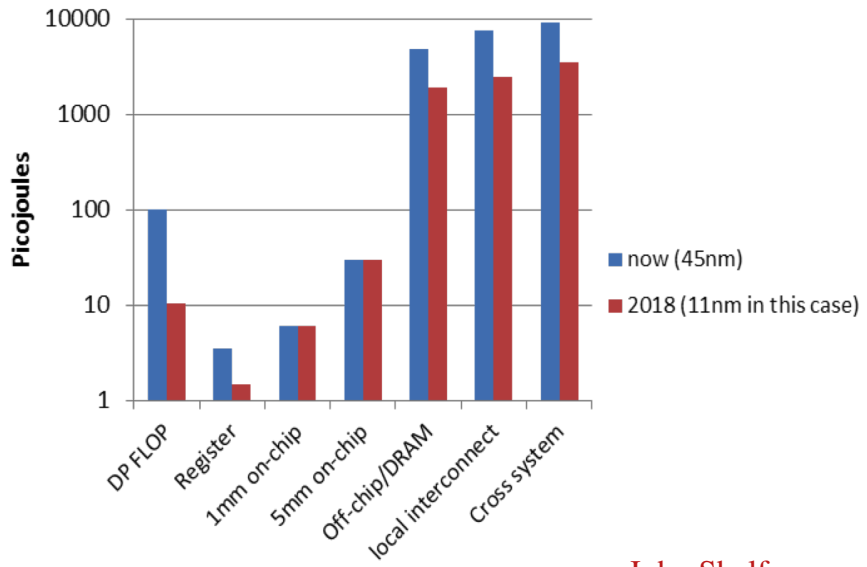
$$\text{Dynamic Power} = K \cdot (\text{Capacitive Load}) \cdot (\text{Voltage})^2 \cdot (\text{Frequency})$$

- Memory wall: bandwidth/latency of the channel b/w CPU and RAM

[Waldrop, 2016, Nature]

Communication-Avoiding Algorithms





John Shalf

Floating-point time	$\ll$	1 / Memory bandwidth	$\ll$	Memory latency
59%		23%		5%
Floating-point time	$\ll$	1 / Network bandwidth	$\ll$	Network latency
59%		26%		15%

- Avoid communication (data movement) to save time $\Longrightarrow$ Redesign algorithms
- Linear algebra, LAPACK/ScaLAPACK, ... J. Demmel and collaborators

Whatever Has Been Done Can Be Outdone

Obstacles to scalable simulation

- Model complexity
- Extreme concurrency
- Multiple levels of parallelism
- Complex memory hierarchies
- More costly data movement
- Hardware failures and soft error

- Optimization: Improve cooling, find hot spots, reduce power leakage
- Less transistors $\Longrightarrow$ lower frequency $\Longrightarrow$ more processing cores
- Scalable, power-aware, resilient parallel algorithms and software

OpenMP Numerical Results

N_T	1st			2nd			3nd			4nd		
	#It	Time	Speedup	#It	Time	Speedup	#It	Time	Speedup	#It	Time	Speedup
1	32	31.34	—	34	32.79	—	34	32.77	—	32	31.49	—
2	32	17.72	1.77	34	18.48	1.77	34	18.46	1.78	32	17.68	1.78
4	32	13.44	2.33	34	13.19	2.49	34	13.14	2.49	32	12.60	2.50
8	33	11.02	2.84	34	11.20	2.93	34	11.18	2.93	32	10.80	2.91

N_T	1st			2nd			3nd			4nd		
	#It	Time	Speedup	#It	Time	Speedup	#It	Time	Speedup	#It	Time	Speedup
1	45	39.01	—	45	38.90	—	43	37.36	—	42	36.56	—
2	45	21.95	1.78	45	21.90	1.78	43	21.00	1.78	42	20.67	1.77
4	45	15.42	2.53	45	15.44	2.52	44	15.19	2.46	42	14.56	2.51
8	45	13.12	2.97	45	13.09	2.97	44	12.86	2.90	42	12.35	2.96
12	45	13.19	2.96	45	13.18	2.95	43	12.66	2.95	42	11.93	3.07

N_T	1st			2nd			3nd			4nd		
	#It	Time	Speedup	#It	Time	Speedup	#It	Time	Speedup	#It	Time	Speedup
1	49	41.69	—	49	41.48	—	48	40.96	—	44	37.75	—
2	49	23.42	1.78	48	22.93	1.81	48	22.87	1.79	44	21.25	1.78
4	49	16.67	2.50	49	16.62	2.50	48	16.30	2.51	44	15.37	2.46
8	49	14.30	2.91	48	13.94	2.98	48	13.91	2.95	44	12.92	2.92
12	48	14.00	2.98	48	13.99	2.97	47	13.58	3.02	44	12.99	2.91

Multicore performance of FASP, CPR, and BTP preconditioners for SPE10 three-phase problems

[Feng, Shu, Xu, Zhang, 2014]

Strong Scaling Tests on Tianhe-2

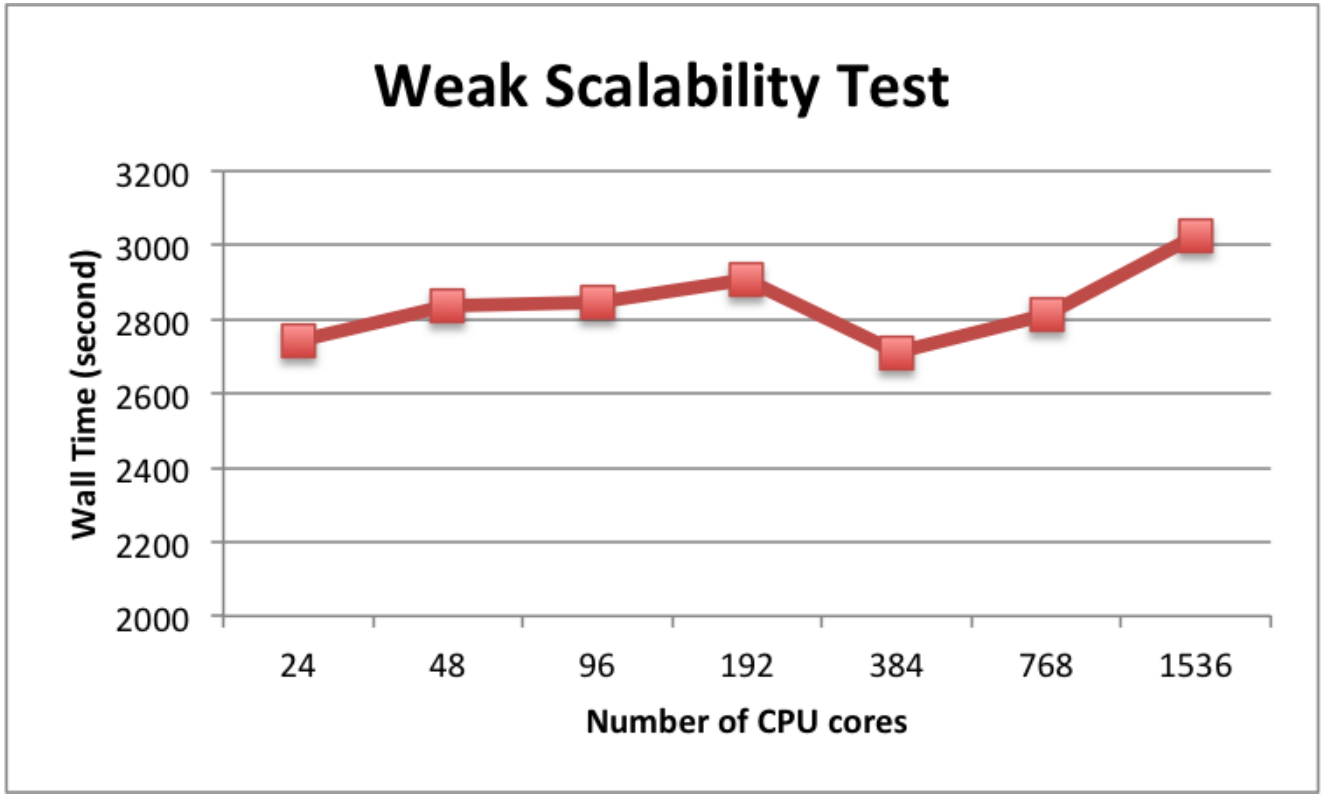
Problem	Size	# Nodes	# Processes	Efficiency	Total time (s)	Solver time(s)	# Newton	# Linear
SPE1-refine	75M cells	40	960		888	583.205	76	614
		60	1440	94%	631	425.531	76	631
		80	1920	89%	497	341.551	76	688
		100	2400	68%	520	367.979	87	958
SPE1-refine	150M cells	80	1920		1294	899.789	81	884
		100	2400	118%	878	609.921	82	818
		120	2880	108%	802	566.173	84	860
		160	3840	95%	678	472.916	92	907
		200	4800	84%	614	438.111	100	1095
SPE9-refine	90M cells	32	768		1953	1079.351	133	277
		128	3072	86%	567	307.249	148	525
		256	6144	64%	381	241.309	148	525

SPE1 and SPE9 benchmark problems refined and then tilted. Tested on the Tianhe-2 cluster, Guangzhou: 2nd among Top500 now, 3.12M cores (32K CPUs), 1.408PB RAM, Rmax 33.86PFlops, Rpeak 54.90PFlops, Power 17.8MW/hr.

[Guan, Qiao, Zhang, et al. 2015]

Weak Scaling Tests on Tianhe-2

# CPU Cores	24	48	96	192	384	768	1536
# Grid Block	3M	6M	12M	24M	48M	96M	192M
# Linear Iterations	723	724	726	727	723	725	715
Total CPU Time (s)	2741	2838	2846	2907	2711	2881	3026



[Guan, Qiao, Zhang, et al. 2015]

Ongoing Work

1. Design robust, optimal, and resilient solvers
   - Improve robustness [Qiao, 2016; Li, 2017]
     - Different reservoir models
     - Different mesh types
     - Different discretizations
     - Why the semi-analytic decoupling works?
   - Improve resilience [Cui, Xu, Zhang, 2017]
     - Less overhead when no error occurs
     - Convergence rate does not deteriorate when error occurs
2. Enhance scalability of MPI, OpenMP, and CUDA versions
   - Improve scalability of the CUDA version [Feng, 2014; Li, 2017]
     - SpMV
     - Block ILU
     - Algebraic multigrid
   - Improve scalability of the MPI version [Guan, et al. 2015]
     - Improve scalability of linear solvers
     - Hierarchical parallelization

Ongoing Work

3. Handle natural and hydraulic fractured reservoirs
   - (Embedded) discrete fracture model
     - Aperture distribution
     - Flow and geomechanic coupling, Biot model
   - Weak Galerkin method, virtual element method
     - Structured / unstructured polyhedral grids
     - Local mesh refinement
4. Improve model parameters (input data)
   - Remove noise from input data
     - Garbage in garbage out (lack of data / imprecise data)
     - Deep Learning
   - Quantify uncertainty
     - Global and local quantities (e.g. production rate v.s. well location)
     - History matching
   - Data-physics

References

Chen, L. and Zhang, C.-S. (2010).
A Coarsening Algorithm on Adaptive Grids by Newest Vertex Bisection and Its Applications.
Journal of Computational Mathematics, 28:1–23.

Cheng, J.-R., Huang, X.-H., Shu, S., Xu, J., Zhang, C.-S., Zhang, S., and Zhou, Z. (2013).
Application of an energy minimizing algebraic multigrid method for subsurface water simulation.
International Journal for Numerical Methods and Modeling, 10(2):374–388.

Cui, T., Xu, J., and Zhang, C.-S. (2017).
An Error-Resilient Redundant Subspace Correction Method.
Computing and Visualization in Science, 18(2):65–77.

Feng, C., Shu, S., Xu, J., and Zhang, C.-S. (2014a).
A Multi-Stage Preconditioner for the Black Oil Model and Its OpenMP Implementation.
In Domain Decomposition Methods in Science and Engineering XXI, pages 129–140.

References

Feng, C., Shu, S., Xu, J., and Zhang, C.-S. (2014b).
Numerical Study of Geometric Multigrid Methods on CPU–GPU Heterogeneous Computers.
Advances in Applied Mathematics and Mechanics, 6(1):1–23.

Guan, W., Qiao, C., Zhang, H., Zhang, C.-S., Zhi, M., Zhu, Z., Ye, W., Zhang, Y., Hu, X., Li, Z., Feng, C., Xu, Y., and Xu, J. (2015).
On Robust and Efficient Parallel Reservoir Simulation on Tianhe-2.
In SPE Reservoir Characterisation and Simulation Conference.

Hu, X., Liu, W., Qin, G., Xu, J., and Zhang, C.-S. (2011).
A Fast Auxiliary Subspace Preconditioner for Numerical Reservoir Simulators.
In SPE Reservoir Characterisation and Simulation Conference. SPE.

Hu, X., Wu, S., Wu, X.-H., Xu, J., Zhang, C.-S., Zhang, S., and Zikatanov, L. (2013a).
Combined Preconditioning with Applications in Reservoir Simulation.
Multiscale Modeling & Simulation, 11(2):507–521.

Hu, X., Xu, J., and Zhang, C.-S. (2013b).
Application of auxiliary space preconditioning in field-scale reservoir simulations.
Science in China Series A: Mathematics, 56(12):2737–2751.

References

Li, Z., Wu, S., Xu, J. J., and Zhang, C.-S. (2016).
Toward cost-effective reservoir simulation solvers on GPUs.
Advances in Applied Mathematics and Mechanics, 8(6):971–991.

Li, Z., Wu, S., Zhang, C.-S., Xu, J., Feng, C., and Hu, X. (2017).
Numerical Studies of A Class of Linear Solvers for Fine-Scale Petroleum Reservoir Simulation.
Computing and Visualization in Science, 18(2):93–102.

Qiao, C., Wu, S., Xu, J., and Zhang, C.-S. (2017).
Analytical Decoupling Techniques for Fully Implicit Reservoir Simulation.
Journal of Computational Physics, 336:664–681.

Wu, S., Xu, J., Feng, C., Zhang, C.-S., Li, Q., Shu, S., Wang, B., Li, X., and Li, H. (2014).
A Multilevel Preconditioner and Its Shared Memory Implementation for A New Generation Reservoir Simulator.
Petroleum Science, 11:540–549.


Thank You!