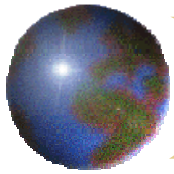
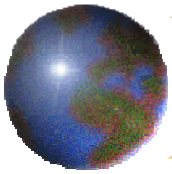


Numerical Methods for Partial Differential Algebraic Systems of Equations

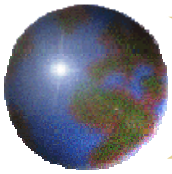
C.T. Miller

University of North Carolina

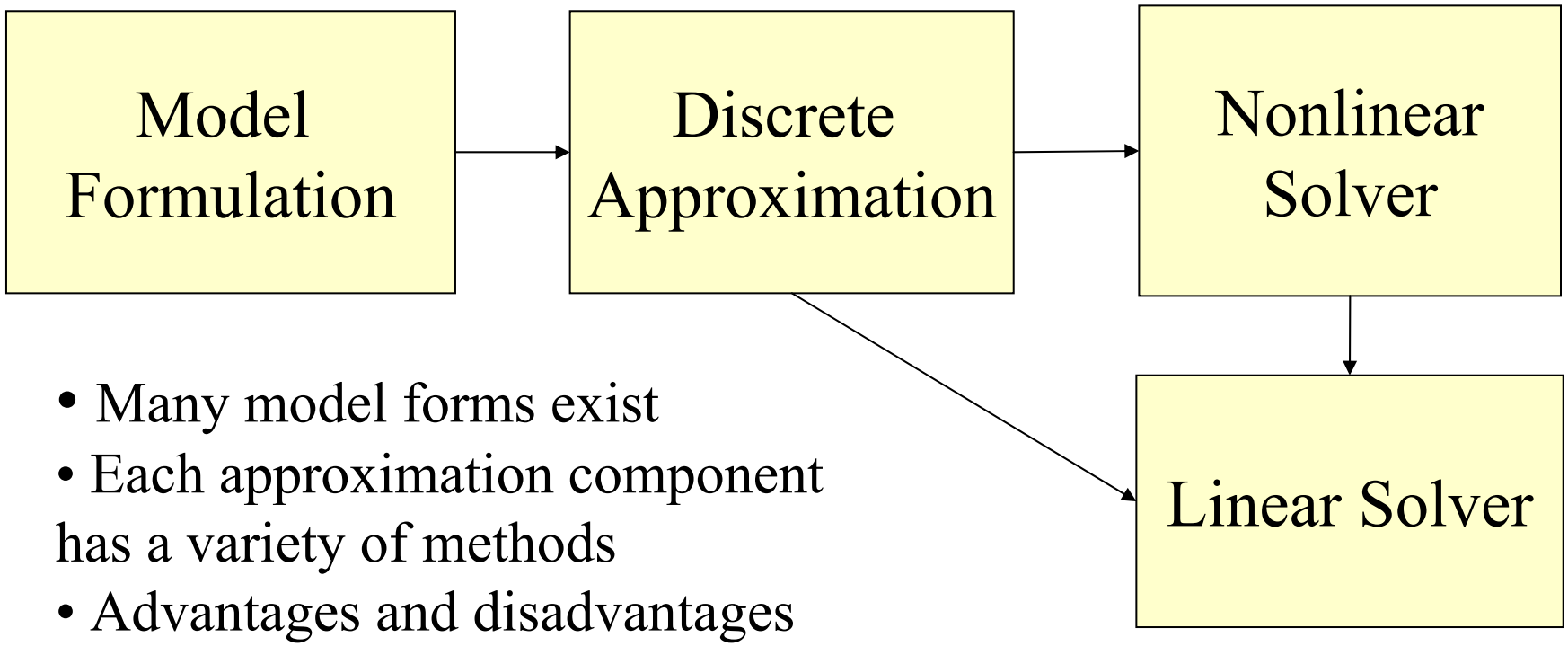


Scope

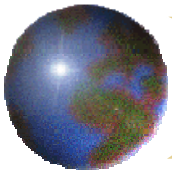
- Linear solvers
- Nonlinear solvers
- Algorithms
- Examples from mathematical geosciences



Approximation of PDAE's



- Many model forms exist
- Each approximation component has a variety of methods
- Advantages and disadvantages for choices made for each component
- Algorithmic considerations are important as well



Elliptic Equation Example

Consider an example elliptic PDE

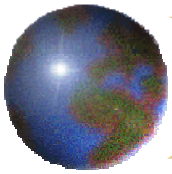
$$\nabla \cdot (K \nabla u) = -f$$

Applying some appropriate discretization operator gives

$$L_d(u) = f_i$$

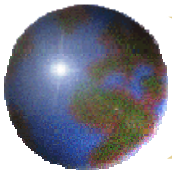
Mapping the local discrete approximation to a global linear algebraic form yields

$$[\mathbf{A}]\{\mathbf{x}\} = \{\mathbf{b}\}$$



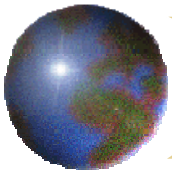
Elliptic Equation Example

- Assume a second-order FDM approximation was used as the discrete operator
- Also assume that the domain is regularly shaped and discretized
- Solve the algebraic system using Gaussian elimination
- Consider the computational requirements to compute the approximation for a 100 x 100 grid in 2D and a 100 x 100 x 100 grid in 3D



Elliptic Equation Example

- Chief computational issues involve memory, CPU time, and more completely computational efficiency
- 2D computational requirements are 800 MB and 11 min on a 1 Gflop machine
- 3D computational requirements are 8 TB and 21 years!
- Lessons:
  1. Computational performance can be an issue even for relatively simple problems
  2. Scaling of methods and algorithms should be considered when choosing methods
  3. Need to consider and exploit special features possible for model



Elliptic Equation Example

Memory for full structure

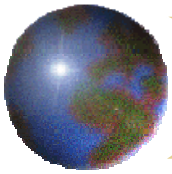
$$M_f = 8n^2$$

Memory for banded structure

$$M_b = 8n(2n_h + 1)$$

Savings in $\mathbb{R}^2$ and $\mathbb{R}^2$

$$\frac{M_b}{M_f} = \frac{2n_h + 1}{n} \approx 2 \times 10^{-2}$$



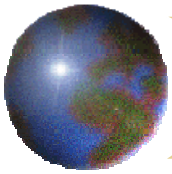
Work Scaling

- Work depends upon number and cost of operations
- Some useful equalities for assessing work are

$$\sum_{j=1}^{n} c = cn$$

$$\sum_{j=1}^{n} j = \frac{n(n+1)}{2}$$

$$\sum_{j=1}^{n} j^2 = \frac{n(3n^2+3n+1)}{6}$$



Gaussian Elimination Work

Full Gaussian elimination work

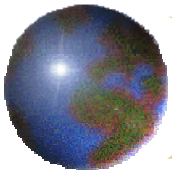
$$w \sim \mathcal{O}\left(2n^3/3\right)$$

Banded Gaussian elimination work

$$w \sim \mathcal{O}\left(2n_h^2 n\right)$$

Optimal scaling is

$$w \sim \mathcal{O}\left(n\right)$$



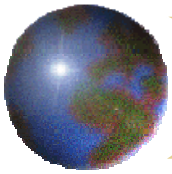
Comparison of Computational Work

Work ratio

$$\frac{w_b}{w_f} = \frac{3n_h^2}{n^2}$$

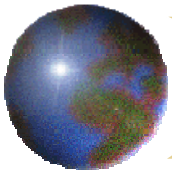
In $\mathbb{R}^2$ or $\mathbb{R}^3$

$$\frac{w_b}{w_f} \approx 3 \times 10^{-4}$$



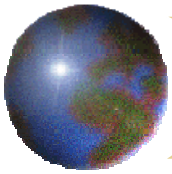
Elliptic Example Implications

- In 2D storage reduced from 800 MB to 16 MB and CPU time reduced from 11 min to 0.2 sec---clearly acceptable
- In 3D storge reduced from 8 TB to 160 GB and CPU reduced from 21 years to 2.31 days---work might be acceptable but storage is not based on current standards



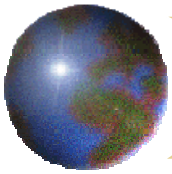
Ponderables

- What are the implications for the scaling of 1D problems using a similar discrete operator?
- What would be the implications of needing to perform pivoting to reduce numerical round-off error?
- What guidance applies for the mapping of the local discrete operator to the global system?
- What simple observation would allow us to reduce the storage and work estimates for the banded case by an additional factor of 2?



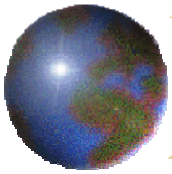
Algebraic Solver Observations

- Even for a simple elliptic model, a compact discrete operator, and an intermediate discretization level---direct methods of solution are untenable
- Storage considerations are even more severe than work limitations
- The direct methods considered are relatively inefficient candidates for parallel processing, which is the chief strategy for increasing computational performance



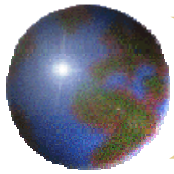
Sparse Storage

- Our example problem, and many others that occur routinely in computational science, are not only banded and symmetric but also very sparse
- We took partial advantage of this for banded systems, but still had to live with fill in
- Iterative methods endeavor to approximate the solution of linear systems taking advantage of the sparse nature
- Special storage schemes are needed to do so



Sparse Storage Schemes

- Many schemes exist
- Store only non-zero entries
- Must be able to reconstruct initial matrix and perform common matrix-vector operations
- Some examples include primary storage, linked list, and specific structure based approaches



Primary Storage

Primary storage transforms

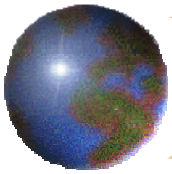
$$[\mathbf{A}] \longmapsto [\mathbf{A}_s] \text{ and } [\mathbf{P}]$$

$[\mathbf{A}_s]$ is an $n \times n_{nz}$ matrix of reals

$[\mathbf{P}]$ is an $n \times n_{nz}$ integer pointer matrix

Total storage becomes

$$M_{ps} = 12 n_{nz} n$$



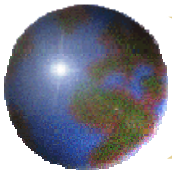
Primary Storage Implications

For elliptic example in $\mathbb{R}^2$

$$\frac{M_{ps}}{M_b} = \frac{3}{80}$$

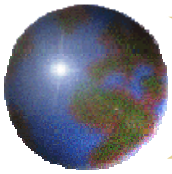
For elliptic example in $\mathbb{R}^3$

$$\frac{M_{ps}}{M_b} \approx 4 \times 10^{-4}$$



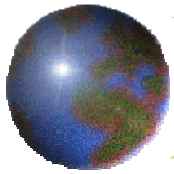
Ponderables

- Show the primary storage scheme meets our requirements for a valid approach
- What would be the implication of using primary storage for the elliptic example in 1D?
- What approaches might further reduce the storage required?



Iterative Solution Approaches

- Seek approaches that in general can operate on linear systems stored in a sparse format
- Two main classes exist: (1) stationary iterative methods, and (2) nonstationary iterative methods
- A primary contrast between direct and iterative methods is the approximate nature of the solution sought in iterative approaches



Stationary Iterative Method Example

Jacobi iteration formulation

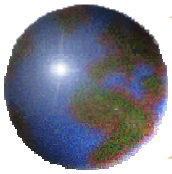
$$[\mathbf{A}]\{\mathbf{x}\} = \{\mathbf{b}\}$$

$$([\mathbf{L}] + [\mathbf{D}] + [\mathbf{U}])\{\mathbf{x}\} = \{\mathbf{b}\}$$

$$\{\mathbf{x}\}^{m+1} = -[\mathbf{D}]^{-1}([\mathbf{L}] + [\mathbf{U}])\{\mathbf{x}\}^{m} + [\mathbf{D}]^{-1}\{\mathbf{b}\}$$

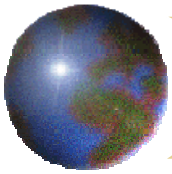
or computationally

$$x_i^{m+1} = (-\sum_{\substack{j=1,n \\ j \neq i}} a_{ij} x_j^m + b_i)/d_{ii}$$



Stationary Iterative Methods

- Theorem: Conditions for the convergence and rate of convergence of this and related methods can be proven
- Similar related methods such as Gauss-Seidel and successive over-relaxation can converge much faster and have a similar computational expense per iteration, which is on the order of one sparse matrix-vector multiply
- These methods have special significance and use as preconditioners for non-stationary iterative methods and as the basis of multigrid methods



Conjugate Gradient Method

Consider a symmetric positive definite matrix

$$[\mathbf{A}] = [\mathbf{A}]^{\mathsf{T}}$$

$$\{\mathbf{x}\}^{\mathsf{T}}[\mathbf{A}]\{\mathbf{x}\} > 0, \ \forall \{\mathbf{x}\} \neq \{\mathbf{0}\}$$

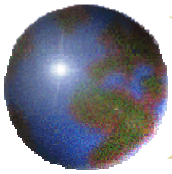
The conjugate gradient method minimizes

$$\phi(\mathbf{x}) = \frac{1}{2}\{\mathbf{x}\}^{\mathsf{T}}[\mathbf{A}]\mathbf{x}\} - \{\mathbf{x}\}^{\mathsf{T}}\{\mathbf{b}\}$$

over $\{\mathbf{x}_0\} + \mathcal{K}_m$ for

$\mathcal{K}_m = \text{span}(\mathbf{r}_0, \mathbf{A}\mathbf{r}_0, \ldots, \mathbf{A}^{m-1}\mathbf{r}_0)$ and

$\{\mathbf{r}_0\} = \{\mathbf{b}\} - [\mathbf{A}]\{\mathbf{x}_0\}$



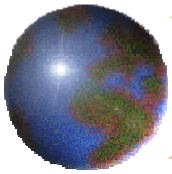
Conjugate Gradient Method

Computational effort for CG method includes: $[\mathbf{A}]\{\mathbf{x}_m\}$, two $\{\mathbf{x}\}^{\mathsf{T}}\{\mathbf{y}\}$, and three $\alpha\{\mathbf{x}\} + \{\mathbf{y}\}$

For full matrix system work per iteration is

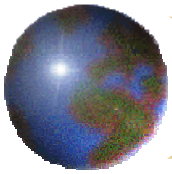
$$w_{cgf} \approx \mathcal{O}\left(2n^2\right)$$

For elliptic example problem in $\mathbb{R}^2$ the work required per iteration is about twice the work required per iteration of Jacobi iteration.



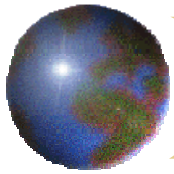
Conjugate Gradient Method

- Theorem: Convergence can be proven to occur in at most n iterations for SPD systems
- Sufficiently accurate solution can usually be obtained in many fewer iterations depending upon the distribution of the eigenvalues of the matrix
- Preconditioning can greatly increase the rate of convergence
- PCG methods can be shown to converge optimally at a rate of n log(n)



Non-Symmetric Systems

- GMRES is a related krylov subspace method for non-symmetric systems for which convergence can also be proven
- The expense of GMRES typically leads to simplifications of the general algorithm in the way of restarts
- Alternative krlov-subspace methods, such as BiCGstab, have proven useful in practice, even though they are not amenable to proofs of convergence
- Suggested references: Kelley (SIAM, 1995) and Barrett et al. (Templates, SIAM, 1994)



Nonlinear Models

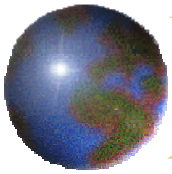
Example parabolic transport problem

$$\frac{\partial C}{\partial t} = \nabla \cdot (\mathbf{D} \cdot \nabla C) - \mathbf{v} \cdot \nabla C - kC^2$$

which can be written as the sum of three operators

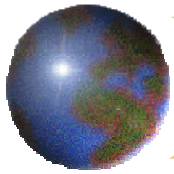
$$\frac{\partial C}{\partial t} = L_D(C) + L_A(C) + L_R(C)$$

The nonlinearity in the $L_R(c)$ operator gives rise to a corresponding nonlinearity in discrete representations of the model.



Nonlinear Models

- Nonlinear models are very common
- Nonlinear algebraic problems results from discrete representations
- Solution requires an iterative approach leading to increased complexity and expense
- Convergence issues are more difficult for nonlinear problems than for linear problems
- A few methods are commonly used



Picard Iteration

Apply an implicit temporal discretization and a low-order spatial discretization to the model problem

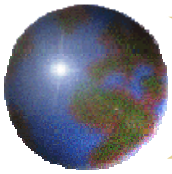
$$\frac{\partial C}{\partial t} = L_D(C) + L_R(C)$$

giving

$$[\mathbf{A}(C)]^m \{\mathbf{C}\}^{m+1} = \{\mathbf{b}\}$$

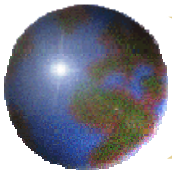
for m an iteration index. Solve iteratively until

$$\|\mathbf{C}^{m+1} - \mathbf{C}^m\| \leq \epsilon_n$$



Picard Iteration

- Nonlinear iteration proceeds until convergence at each time step
- Theorem: Rate of convergence is linear
- For many problems of interest in hydrology, Picard iteration has proven to be robust
- Method is relatively cheap computationally per iteration and easy to implement
- Also known as fixed-point iteration, successive substitution, or nonlinear Richardson iteration



Newton Iteration

Consider a nonlinear equation of the form

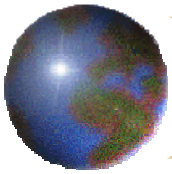
$$f(x) = 0$$

Writing as a Taylor series

$$f(x + \Delta x) = f(x) + \Delta x \frac{\mathrm{d}f}{\mathrm{d}x} + \ldots$$

leading to iterates of the form

$$x^{m+1} = x^m - \frac{f(x)}{\frac{\mathrm{d}f}{\mathrm{d}x}}$$



Newton Iteration

For a system of equations this approach generalizes to

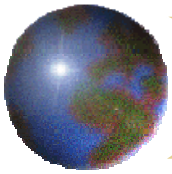
$$\{\mathbf{f}\} = \{\mathbf{0}\}$$

$$[\mathbf{J}]\{\boldsymbol{\Delta x}\} = -\{\mathbf{f}\}$$

$$[\mathbf{J}] = \left[\frac{\partial f_i}{\partial x_j}\right]$$

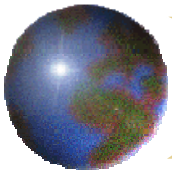
$$\{\mathbf{x}\}^{m+1} = \{\mathbf{x}\}^m + \{\boldsymbol{\Delta x}\}^m$$

until $\|\mathbf{f}\| \leq \epsilon_n$



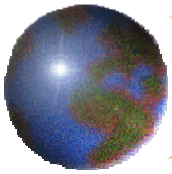
Newton Iteration

- Theorem: Close to the solution, Newton iteration converges quadratically
- [J] may be expensive to compute or not accessible
- The ball of convergence of Newton's method may be small
- Each nonlinear iteration requires the solution of a linear system of equations, which may be accomplished directly or more commonly iteratively---resulting in nested iteration



Newton Iteration

- If [J] cannot be compute analytically, it can be formed using a finite difference approximation
- If [J] is costly to compute, it can be reused over multiple iterations, which is known as the chord method
- Inexact Newton methods result when the iterative linear solution tolerance is functionally dependent upon the magnitude of f

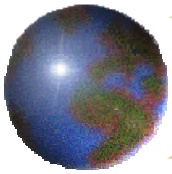


Newton Iteration/Line Search

If a single solution to a smooth convex discrete problem exists, Newton's method can still fail to converge, and frequently does so!

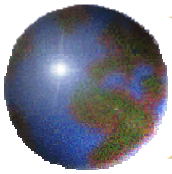
For such a problem, it can be proven that $\{\boldsymbol{\Delta x}\}$ points in decent direction for $\{\mathbf{f}\}$

However, $\|\mathbf{f}(\mathbf{x}+\boldsymbol{\Delta x})\| < \|\mathbf{f}(\mathbf{x})\|$ cannot be proven and frequently isn't



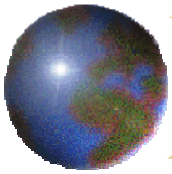
Newton Iteration/Line Search

- Accept Newton direction but not the step size
- If step size doesn't produce a sufficient decrease in ||f|| reduce the magnitude of the step by ½ (Armijo's rule) or a local quadratic/cubic model
- Continue until a sufficient decrease is found
- Close to the solution, full Newton steps and thus quadratic convergence is expected



Algorithms

- MOL approaches---formal decoupling of spatial and temporal components
- Operator splitting methods---approximate the overall operator as a sum of operators acting on components of the original problem
- Adaptive methods in time, space (h, p, r, h-p) and space-time



Split-Operator Approaches

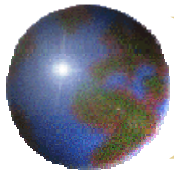
Recall example

$$\frac{\partial C}{\partial t} = \nabla \cdot (\mathbf{D} \cdot \nabla C) - \mathbf{v} \cdot \nabla C - kC^2$$

which can be written as the sum of three operators

$$\frac{\partial C}{\partial t} = L_D(C) + L_A(C) + L_R(C)$$

which can be split into component operator solves over a time step.



Sequential Split-Operator Approach

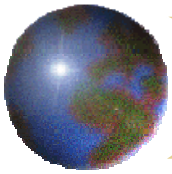
$$\frac{\partial C}{\partial t} = L_A(C), t \in [t, t + \Delta t]$$

$$\frac{\partial C}{\partial t} = L_D(C), t \in [t, t + \Delta t]$$

and

$$\frac{\partial C}{\partial t} = L_R(C), t \in [t, t + \Delta t]$$

leads to $\epsilon_s = \mathcal{O}(\Delta t)$

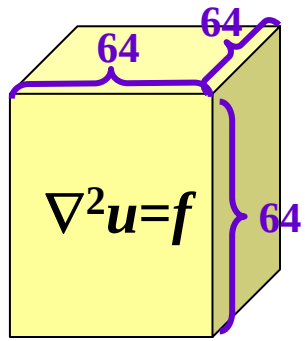


Split-Operator Approaches

- Variety of algorithms exist with tradeoffs of complexity and accuracy
- Splitting error can range from $O(\Delta t)$ to zero
- Allow combining methods well suited to individual components---hyperbolic and parabolic parts, linear and nonlinear parts, etc
- Can lead to reductions in overall size of solve for any component and advantages for parallel algorithms

Computation and Algorithms

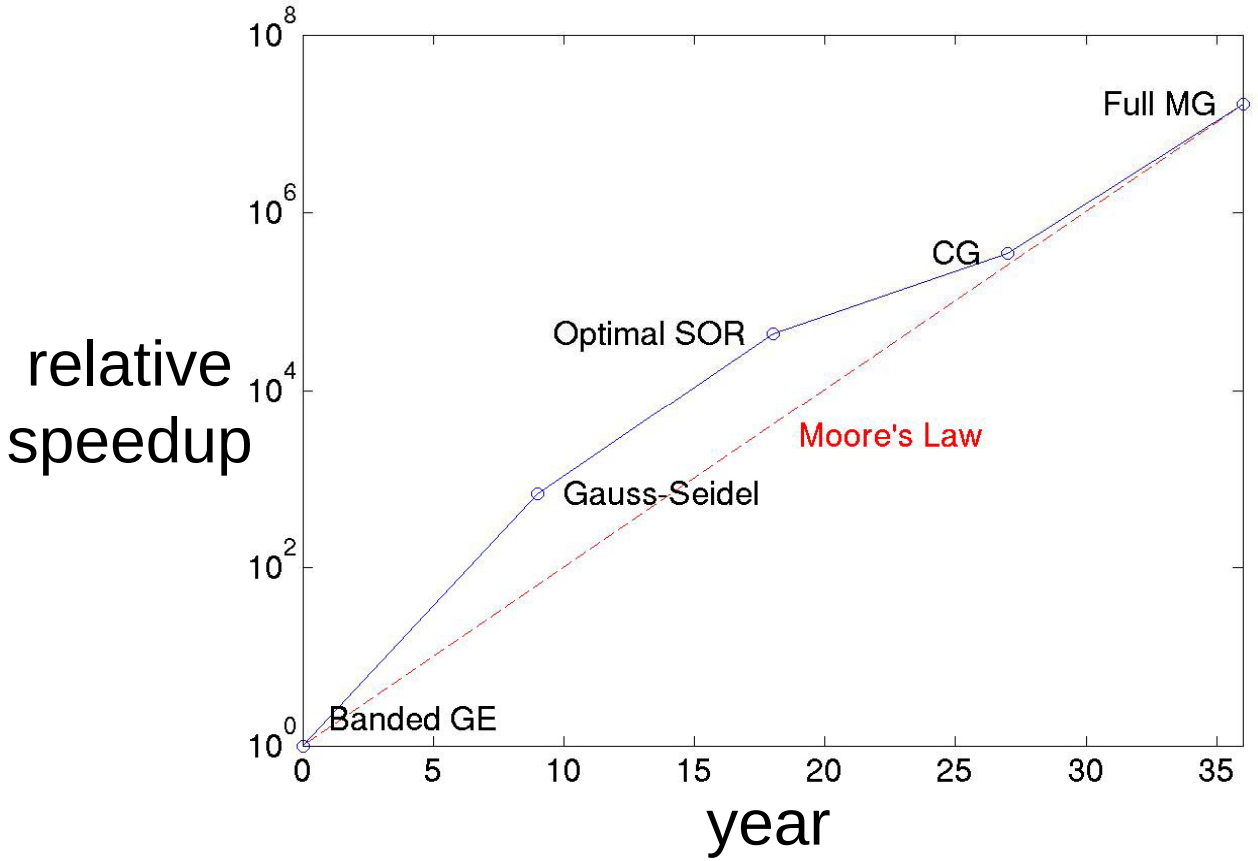
Year	*Method*	*Reference*	*Storage*	*Flops*
1947	GE (banded)	Von Neumann & Goldstine	n^5	n^7
1950	Optimal SOR	Young	n^3	$n^4 \log n$
1971	CG	Reid	n^3	$n^{3.5} \log n$
1984	Full MG	Brandt	n^3	n^3



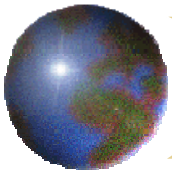
- Advances in algorithmic efficiency rival advances in hardware architecture
- Consider Poisson's equation on a cube of size $N=n^3$
- If n=64, this implies an overall reduction in flops of ~16 million

D. E. Keyes, Columbia University

Computation and Algorithms



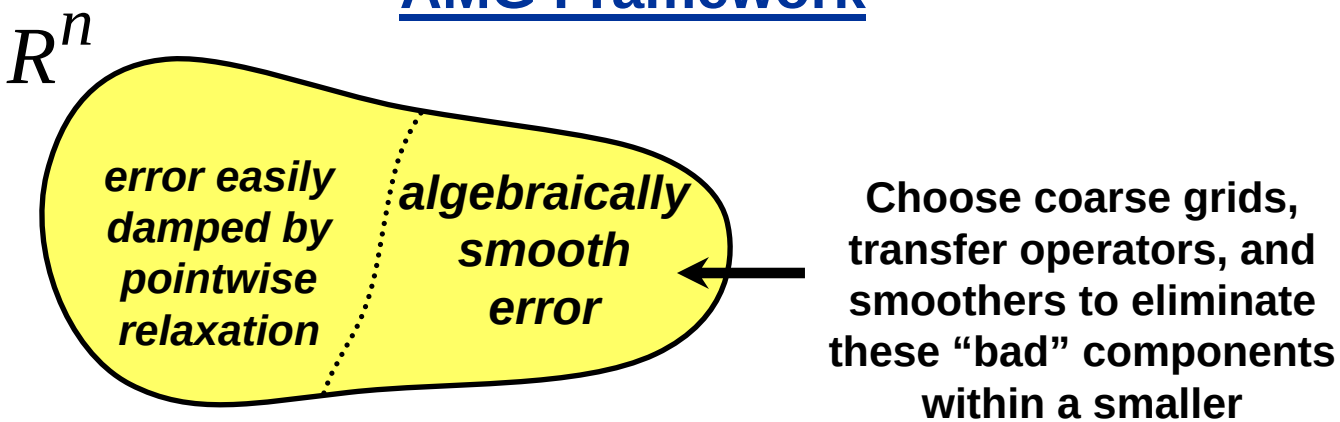
D. E. Keyes, Columbia University



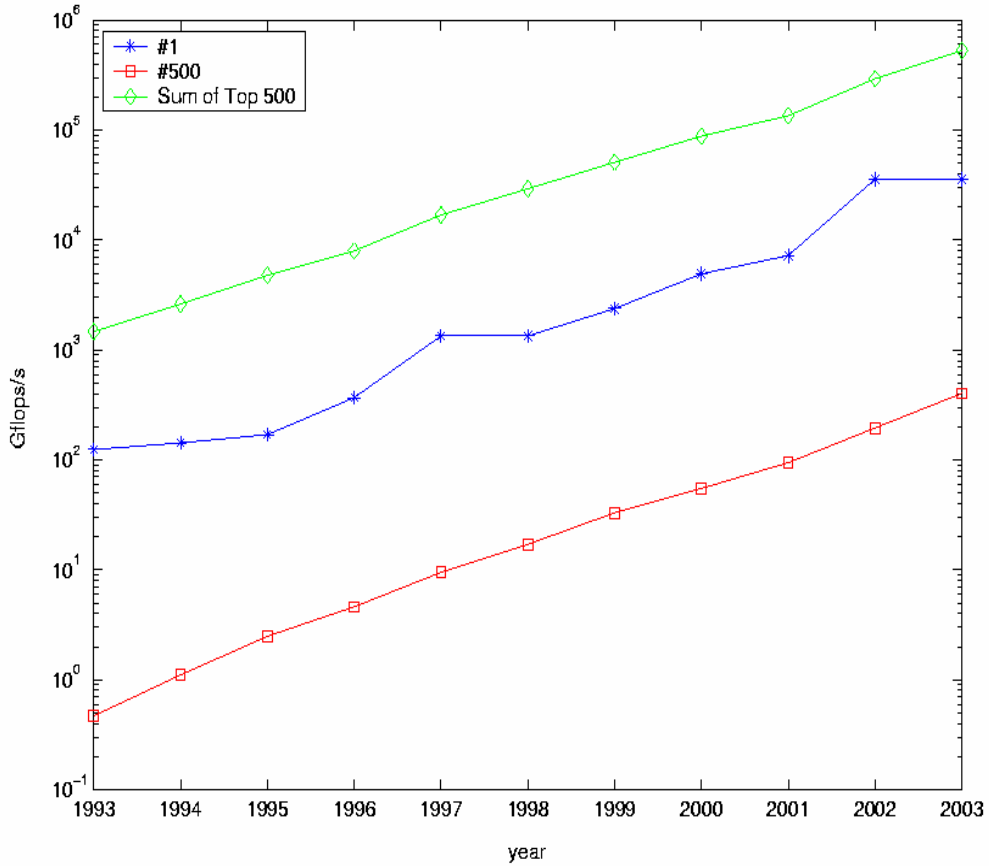
Where to go past O(N) ?

- Hence, for instance, algebraic multigrid (AMG), obtaining *O(N)* in *indefinite*, *anisotropic*, or *inhomogeneous* problems
- Since *O(N)* is already optimal, there is nowhere further "upward" to go in efficiency, but one must extend optimality "outward", to more general problems

AMG Framework



Computational Performance



TOP500 SUPERCOMPUTER SITES (http://www.top500.org/)

- Current peak performer is the DOE's BlueGene/L at LLNL, which has 131,072 processors and peaks at 367,000 GFLOPs
- Number 10 on the current list is Japan's Earth Simulator with has 5,200 processors and peaks at 40,960 GFLOPs, which was built in 2002
- Number 500 on current list is a 1028 Xeon 2.8 GHz processor IBM xSeries cluster, which peaks at 5,756.8 GFLOPs

Richards' Equation Formulation

$$\epsilon^{\alpha}\frac{\partial \rho^{\alpha}}{\partial t} + \rho^{\alpha}\frac{\partial \epsilon^{\alpha}}{\partial t} = -\nabla \cdot (\epsilon^{\alpha}\rho^{\alpha}\mathbf{v}^{\alpha}) + \mathcal{I}^{\alpha} + \mathcal{S}^{\alpha}$$

$$\sum_{\alpha} \epsilon^{\alpha} = 1$$

$$\rho^{\alpha} = \rho^{\alpha}(p^{\alpha})$$

Richards' Equation Formulation

$$\mathbf{q}^{\alpha} = \epsilon^{\alpha}\mathbf{v}^{\alpha} = -\frac{kk^{r\alpha}}{\mu^{\alpha}}\nabla\left(p^{\alpha} + \rho^{\alpha}gz\right)$$

$$\epsilon^{\alpha}\frac{\partial\rho^{\alpha}}{\partial t} + \rho^{\alpha}\frac{\partial\epsilon^{\alpha}}{\partial t} =$$

$$\nabla\cdot\left[\frac{\rho^{\alpha}kk^{r\alpha}}{\mu^{\alpha}}\nabla\left(p^{\alpha} + \rho^{\alpha}gz\right)\right] + \mathcal{I}^{\alpha} + \mathcal{S}^{\alpha}$$

Richards' Equation Formulation

$$S^{\alpha} = f\left(p^{\beta}(t)\right), \text{ for } \beta = 1, ..., n_f$$

$$k^{r\alpha} = f\left(S^{\beta}(t)\right), \text{ for } \beta = 1, ..., n_f$$

$$S_e = \frac{\epsilon^a - \epsilon^r}{\epsilon^s - \epsilon^r} = (1 + |\alpha_v \psi|^{n_v})^{-m_v} \text{ for } \psi < 0$$

$$= 1 \text{ for } \psi \geq 0$$

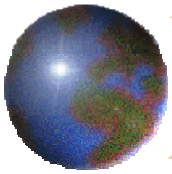
$$k^{rw}(S_e) = S_e^{1/2}\left[1 - \left(1 - S_e^{1/m_v}\right)^{m_v}\right]^2$$

Richards' Equation Formulation

$$S_s S_a(\psi)\frac{\partial \psi}{\partial t} + \frac{\partial \theta}{\partial t} = \frac{\partial}{\partial z}\left[K_z(\psi)\left(\frac{\partial \psi}{\partial z} + 1\right)\right]$$

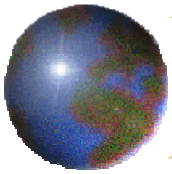
$$[c(\psi) + S_s S_a(\psi)]\frac{\partial \psi}{\partial t} = \frac{\partial}{\partial z}\left[K_z(\psi)\left(\frac{\partial \psi}{\partial z} + 1\right)\right]$$

$$\begin{aligned} \psi(z, t = 0) &= \psi_0(z) \\ \psi(z = 0, t > 0) &= \psi_1 \\ \psi(z = Z, t > 0) &= \psi_2 \end{aligned}$$



Standard Solution Approach

- Mixed-form of RE
- Arithmetic mean relative permeabilities
- Analytical evaluation of closure relations
- Low-order finite differences or finite element methods in space
- Backward Euler approximation in time
- Modified Picard iteration for nonlinear systems
- Thomas algorithm for linear equation solution



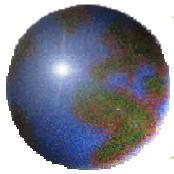
Algorithm Advancements

- Spline closure relations
- Variable transformation approaches
- Mass conservative formulation
- DAE/MOL time integration
- Spatially adaptive methods
- Nonlinear solvers
- Linear solvers

DAE/MOL Solution to RE

$$A(\psi_i)\frac{\mathrm{d}\psi_i}{\mathrm{d}t} = O_{sdi}(\psi)$$

$$O_{sdi}(\psi) = \Delta z^{-1}\left(\frac{K_{i+1/2}(\psi_{i+1}-\psi_i)}{\Delta z} - \frac{K_{i-1/2}(\psi_i-\psi_{i-1})}{\Delta z} + K_{i+1/2} - K_{i-1/2}\right)$$



DAE/MOL Solution to RE

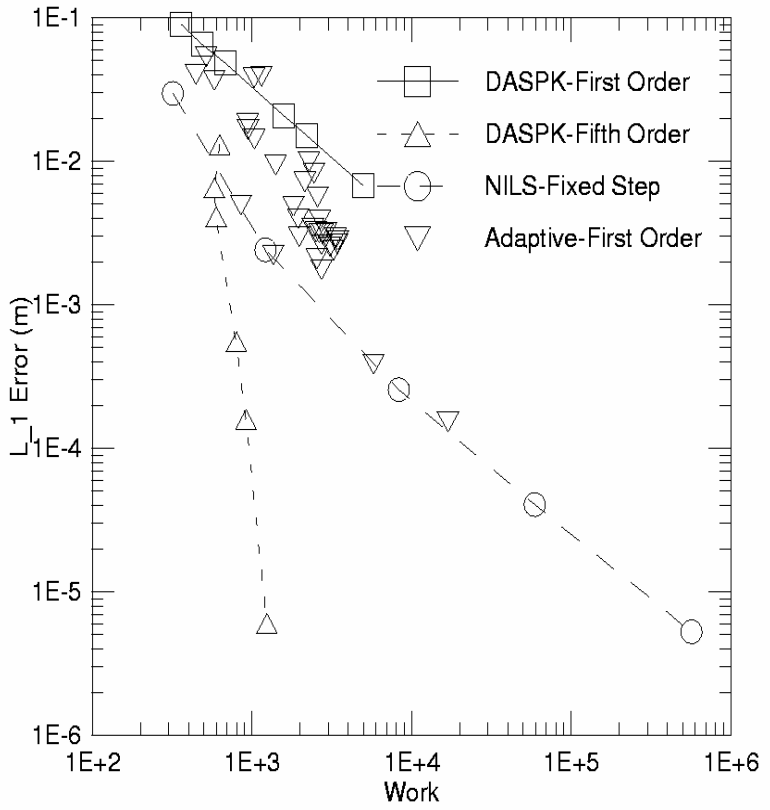
$$F(t_n, y^n, h_n^{-1} \sum_{}^{k} \alpha_j y^{n-j}) = 0$$

$$F(t, y, \alpha y + \beta) = 0$$

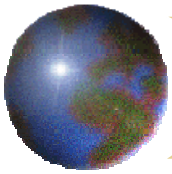
$$\left(\alpha \frac{\partial F}{\partial y'} + \frac{\partial F}{\partial y} \right) s_m = -F(t, y_m, \alpha y_m + \beta)$$

$$\tau_i \leq \epsilon_i = \epsilon_r |y_i| + \epsilon_a, \text{ for } i = 1, ..., n_n$$

DAE/MOL RE



- Temporal truncation error comparison
- Mixed-form Newton iteration, line search
- Heuristic adaptive time stepping
- DASPK first and fifth order integration
- Reference: Tocci et al. (1997), AWR



SAMOL Algorithm

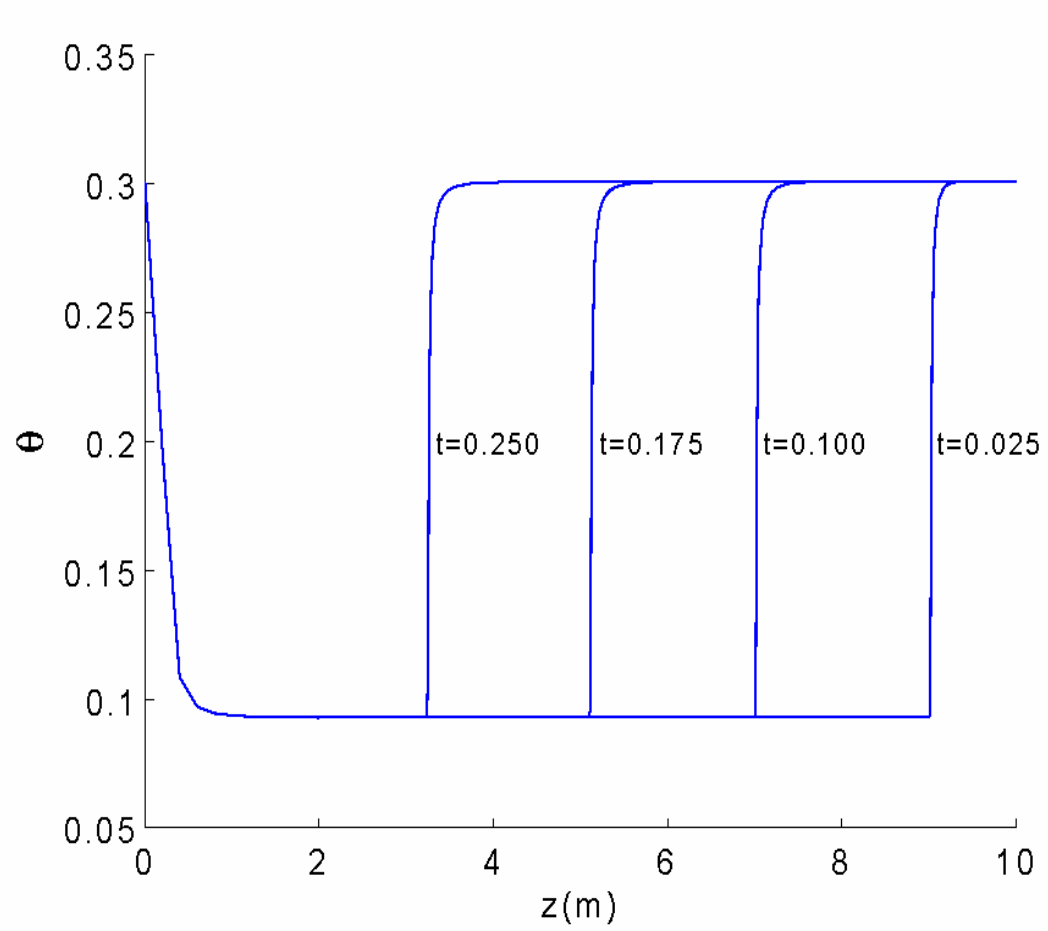
Algorithm 1 SAMOL-RE

Require: $\boldsymbol{\psi}_f^{n=0}$ and $\boldsymbol{\theta}_f^{n=0}$ to be consistent and adequately resolved in space on $\mathcal{G}^{n=0}$, and the operators $\mathcal{P}, \mathcal{I},$ and $\mathcal{D}$ to be robust and sufficiently accurate

1: **for** $n = 0$ to $n = n_f$ **do**
2: $\quad \boldsymbol{\psi}_c^n(\mathcal{G}_c) \leftarrow \mathcal{P}[\boldsymbol{\psi}_f^n(\mathcal{G}^n)]$
3: $\quad \boldsymbol{\theta}_c^n(\mathcal{G}_c) \leftarrow \mathcal{P}[\boldsymbol{\theta}_f^n(\mathcal{G}^n)]$
4: $\quad \boldsymbol{\psi}_c^{n+1}(\mathcal{G}_c) \leftarrow \mathcal{I}[\boldsymbol{\psi}_c^n(\mathcal{G}_c)]$
5: $\quad \mathcal{G}^{n+1} \leftarrow \mathcal{D}[\boldsymbol{\theta}_c^n, \boldsymbol{\theta}_c^{n+1}]$
6: $\quad \boldsymbol{\psi}_f^n(\mathcal{G}^{n+1}) \leftarrow \mathcal{P}[\boldsymbol{\psi}_f^n(\mathcal{G}^n)]$
7: $\quad \boldsymbol{\theta}_f^n(\mathcal{G}^{n+1}) \leftarrow \mathcal{P}[\boldsymbol{\theta}_f^n(\mathcal{G}^n)]$
8: $\quad \boldsymbol{\psi}_f^{n+1}(\mathcal{G}^{n+1}) \leftarrow \mathcal{I}[\boldsymbol{\psi}_f^n(\mathcal{G}^{n+1})]$
9: **end for**

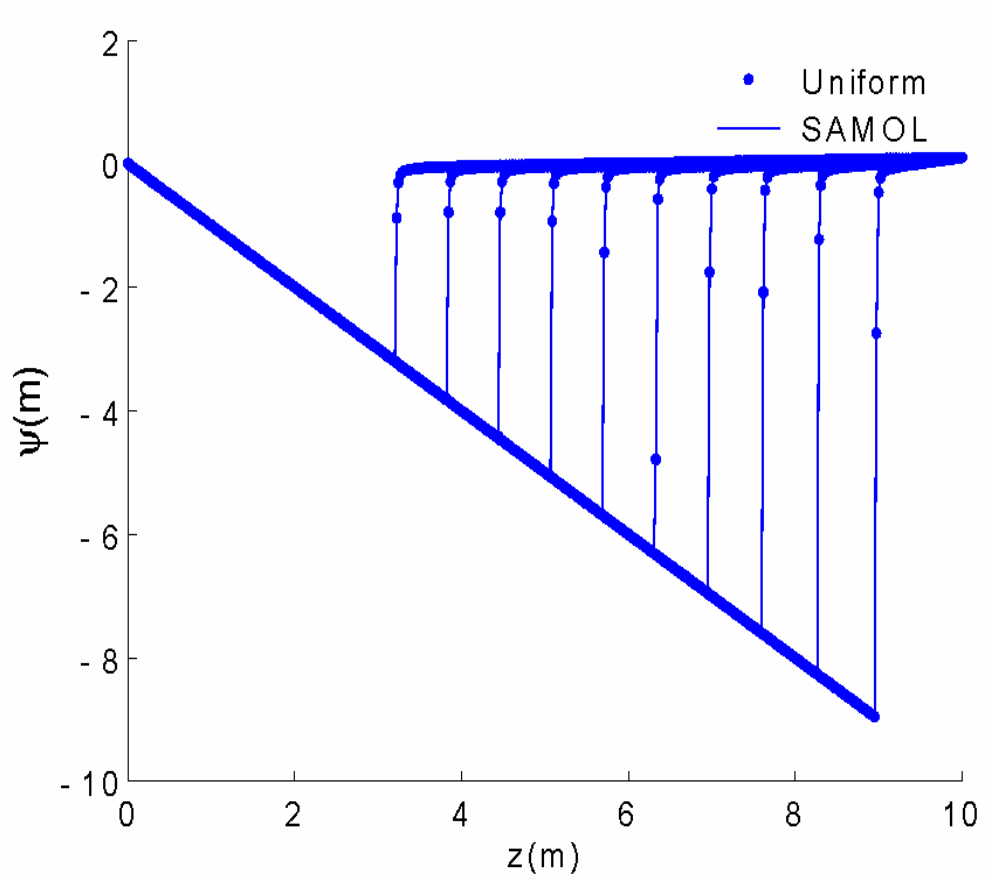
Ensure: $\boldsymbol{\psi}_f^{n_f}$ and $\boldsymbol{\theta}_f^{n_f}$ to be consistent and adequately resolved in space and time within the bounds provided by the spatial error indicator and temporal error criteria

Infiltration Test Problem

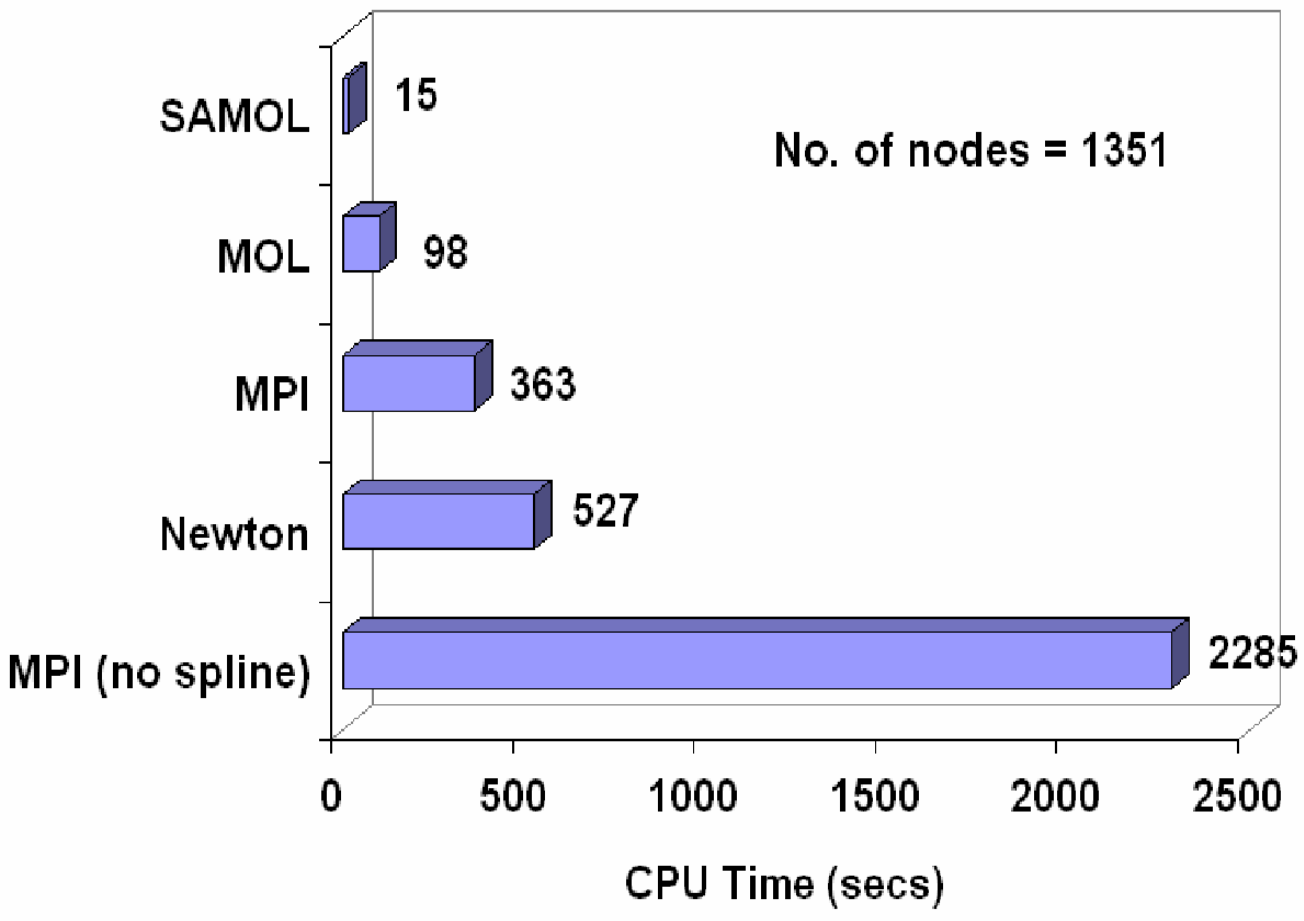


- VG-Mualem psk relations
- Dune sand medium
- Drained to equilibrium
- First-kind boundary conditions
- Simulation time in days

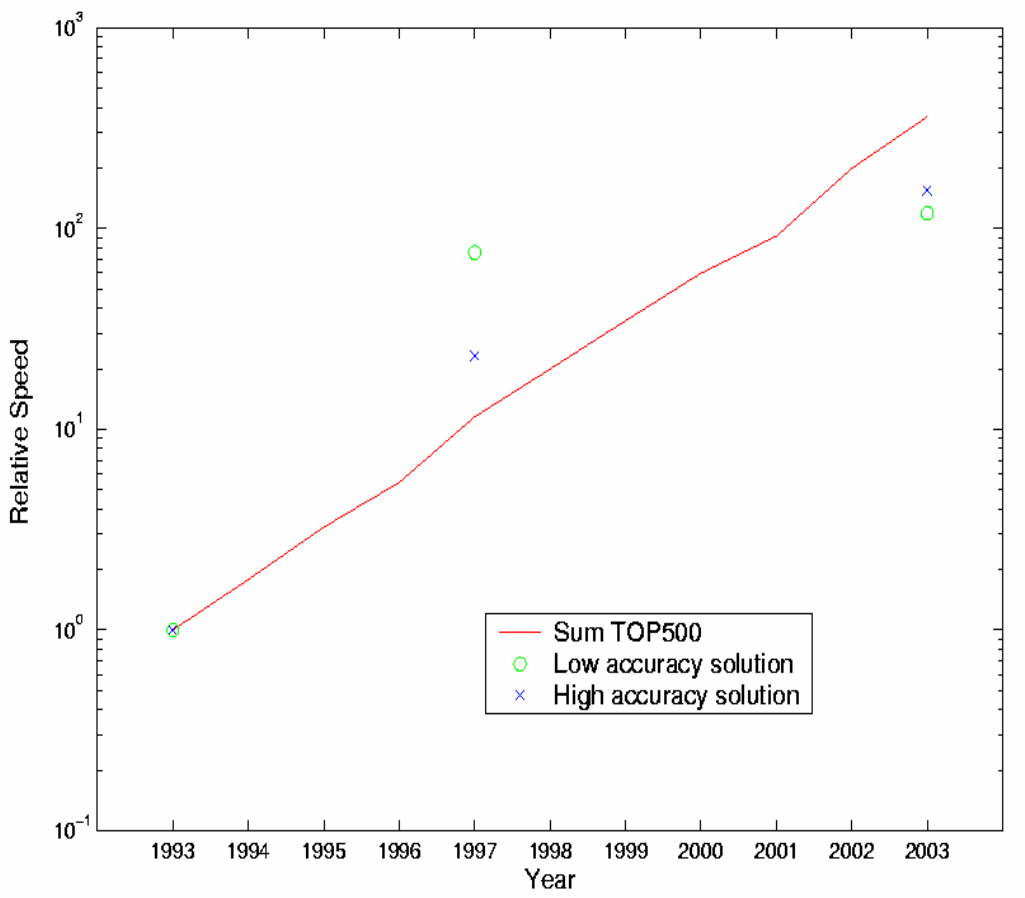
SAMOL Simulation Profile



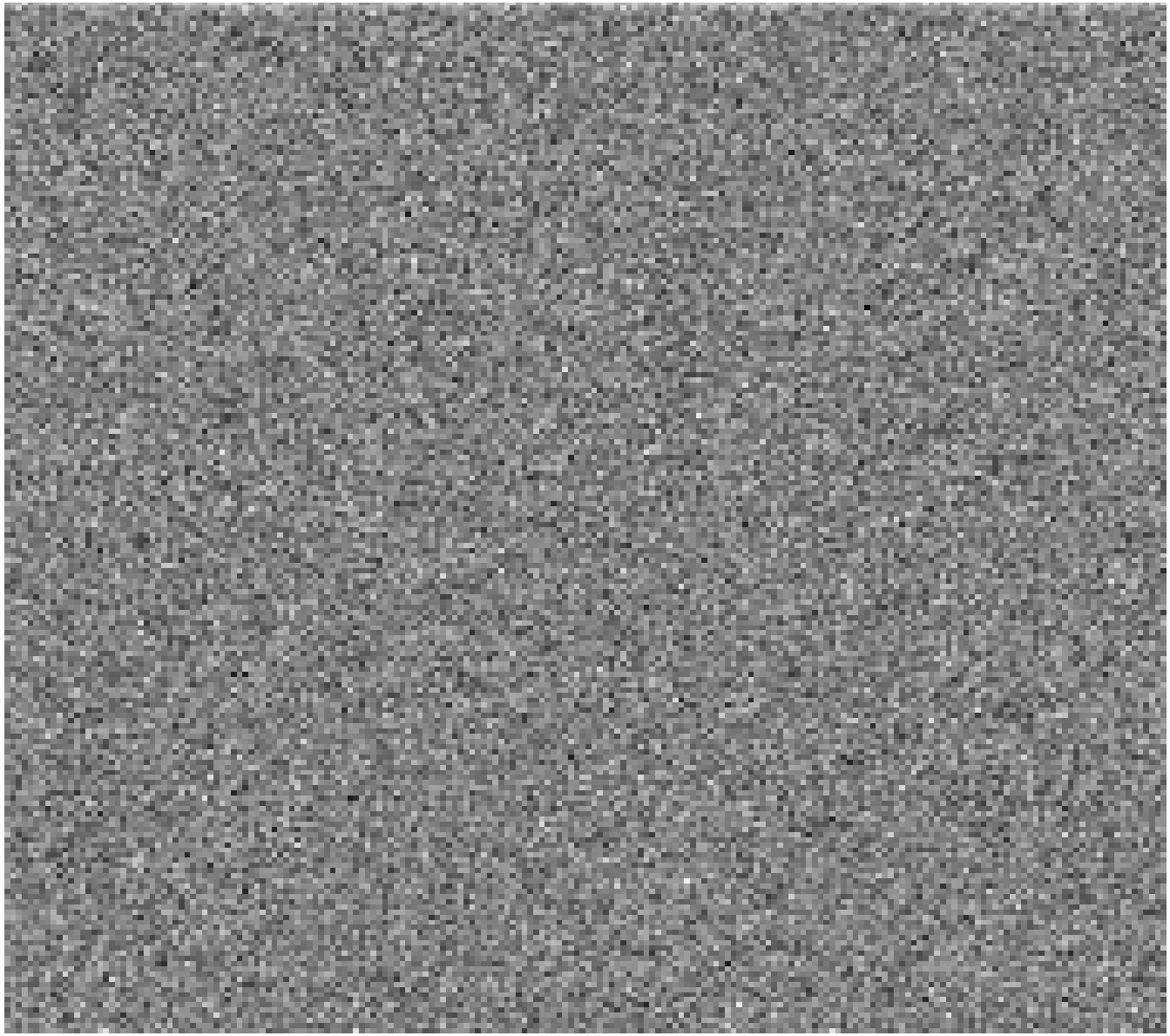
Comparison of RE Results

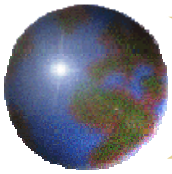


Computational and Algorithm Performance



Dissolution Fingering Example





Conservation Equations and Constraints

Species Balance Equation:

$$\frac{\partial}{\partial t}(\epsilon^{\alpha}\rho^{\alpha}\omega^{\iota\alpha}) = -\nabla\cdot(\mathbf{j}^{\iota\alpha} + \epsilon^{\alpha}\rho^{\alpha}\omega^{\iota\alpha}\mathbf{v}^{\alpha}) + \mathcal{I}^{\iota\alpha} + \mathcal{R}^{\iota\alpha} + \mathcal{S}^{\iota\alpha}$$

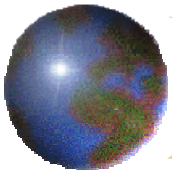
Species-Summed Flow Equation:

$$\frac{\partial}{\partial t}(\epsilon^{\alpha}\rho^{\alpha}) = -\nabla\cdot(\epsilon^{\alpha}\rho^{\alpha}\mathbf{v}^{\alpha}) + \mathcal{I}^{\alpha} + \mathcal{S}^{\alpha}$$

$$\sum_{\alpha}\epsilon^{\alpha} = 1,\ \sum_{\alpha}\mathcal{I}^{\iota\alpha} = 0,$$

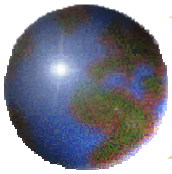
$$\sum_{\iota}\omega^{\iota\alpha} = 1,\ \sum_{\iota}\mathbf{j}^{\iota\alpha} = 0,\ \sum_{\iota}\mathcal{R}^{\iota\alpha} = 0$$

$$\sum_{\iota}\mathcal{I}^{\iota\alpha} = \mathcal{I}^{\alpha}\ \sum_{\iota}\mathcal{S}^{\iota\alpha} = \mathcal{S}^{\alpha}$$



Simulation of Dissolution Fingering

- Two-phase flow and species transport
- Complexity in flow field must be resolved
- Separation of time scales
- Adaptive methods in space useful



Current Research Foci

- Locally conservative methods
- Higher-order methods in space and time
- Integral equation methods
- Multiscale methods
- Problem solving environments