

**COMPARISON OF SOME PARALLEL KRYLOV
SOLVERS FOR LARGE SCALE
GROUNDWATER CONTAMINANT TRANSPORT
SIMULATIONS**

*G. Mahinthakumar
Center for Computational Sciences
Oak Ridge National Laboratory*

*F. Saied
Dept. of Computer Science
University of Illinois at Urbana — Champaign*

*A. J. Valocchi
Dept. of Civil Engineering
University of Illinois at Urbana — Champaign*

OUTLINE

● Introduction

- Krylov Methods for Non-Symmetric Systems
- Contaminant Transport Equation

● Implementation Details

- Numerical Formulation
- Parallelization
- Model Problems
- Parallel Architectures

● Results

- Scalability
- Parallel Efficiency
- Convergence behavior
- Comparison among parallel platforms
- Effect of Heterogeneity
- Effects of Courant, Peclet Numbers, increasing problem size etc.
- Floating Point Performance

● Conclusions

KRYLOV METHODS

● Krylov solvers are iterative solvers used particularly for large and sparse linear systems

- Krylov subspace methods for solving a linear system $Ax=b$ are iterative methods that pick the j -th iterate from the subspace $X_j \in x_0 + K_j(A, r_0)$ where x_0 is the initial guess, r_0 the corresponding residual vector and the Krylov subspace $K_j(A, r_0)$ is defined as $K_j(A, r_0) \equiv \text{span}(r_0, Ar_0, A^2r_0, \dots, A^{j-1}r_0)$.

● Advantages

- Requires much less storage than direct solvers [$O(N)$ vs $O(N^2)$]
- Much fewer operations than direct solvers especially for 3-D problems [e.g., $O(N^{1.3})$ vs $O(N^2)$ for sparse GE and $O(N^{1.3})$ vs $O(N^3)$ for full GE].
- Most operations can be categorized into saxpy's, dot products and sparse matrix vector products; easy to code.
- Does not require matrix, can benefit from sparsity or matrix free algorithms.

● Disadvantages

- Convergence is not guaranteed for all linear systems
- Lower floating point performance due to level 1 BLAS operations; particularly true for memory bandwidth limited machines. Performance can be 3 to 4 times lower than dense matrix operations.

SOME POPULAR KRYLOV SOLVERS FOR NON-SYMMETRIC SYSTEMS

Older Methods

- FOM/Arnoldi (Full Orthogonalization Method)
- ORTHOMIN (Orthogonal Minimization)
- CGS (Conjugate Gradient Squared)
- GCR (Conjugate Residual)
- CGNE and CGNR (Conjugate Gradient on Normal Equations)

Newer Methods

- BICGSTAB (Bi-Orthogonal Conjugate Gradient Stabilized)
- GMRES (General Minimal Residual Algorithm)
- QMR (Quasiminimal Residual Algorithm)
- TFQMR (Transpose Free QMR)
- QMRCGSTAB

GROUNDWATER TRANSPORT EQUATION

- Single species transport equation

$$R \frac{\partial c}{\partial t} = \nabla \cdot (\mathbf{D} \cdot \nabla c) - \nabla \cdot (c \mathbf{v}) - \lambda R c - \frac{q}{\theta} (c - c_0)$$

- Dispersion tensor

$$D_{ij} = \alpha_T |\mathbf{v}| \delta_{ij} + (\alpha_L - \alpha_T) \frac{v_i v_j}{|\mathbf{v}|} + D_m$$

- Retardation factor

$$R = 1 + \frac{\rho K_d}{\theta}$$

- Velocity field

$$\nabla (\nabla h) = q$$

$$\theta \mathbf{v} = - \nabla h$$

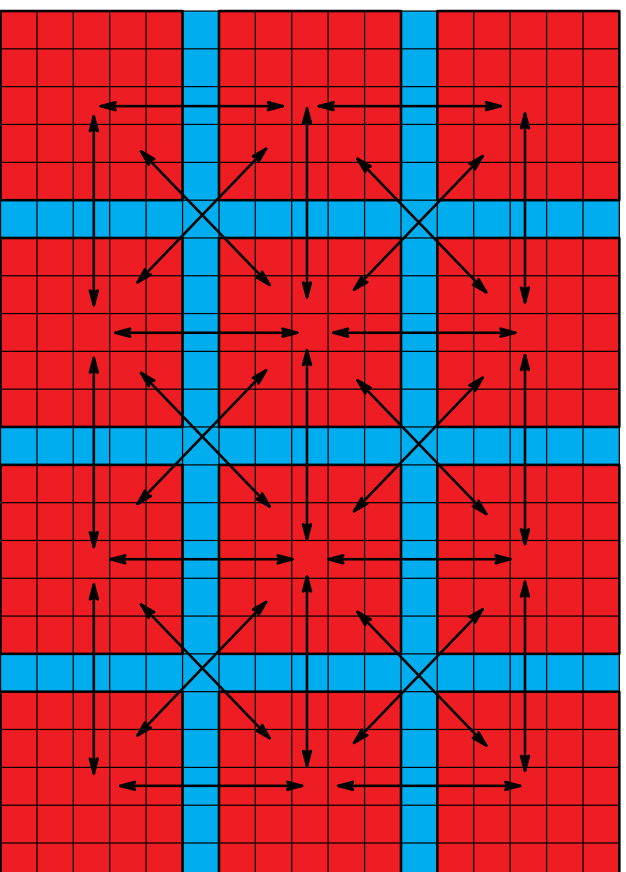
NUMERICAL IMPLEMENTATION

- Three-dimensional advection–dispersion equation with first order reactions
- Galerkin finite–elements with hexahedral (8–node) elements
 - 27–point stencil →27–diagonal non–symmetric matrix
 - upstream weighted formulation
 - variable weighted finite–difference for time derivative
- Logically rectangular grid structure assumed using “natural ordering” of nodes
- Krylov solvers (diagonal preconditioning for all solvers)
 - BICGSTAB
 - GMRES(m)
 - ORTHOMIN(k)
 - CGS

PARALLEL IMPLEMENTATION

- **Two-dimensional domain decomposition**
 - communication with at most 8 neighboring processors
 - natural node ordering for individual processor regions
- **Explicit message passing required to exchange information at processor boundaries especially during assembly and matrix-vector product stages**
 - NX communication library for Intel architectures
 - MPI for other architectures

PARALLEL DOMAIN DECOMPOSITION



Plan View of Two-Dimensional Domain Decomposition
(showing a 4x3 processor decomposition)

PARALLEL ARCHITECTURES USED IN THIS STUDY

- **Intel Paragon XPS/150 (CCS – ORNL)**
 - 1024 nodes
 - 64 Mb/node
 - 80 Mflop/node peak (in single threaded mode)
- **SGI Power Challenge Array (NCSA)**
 - 162 nodes total (max 32 allowed at one time)
 - 360 Mflop/node peak (R10000 processors)
 - 256 Mb/node
- **Convex Exemplar (NCSA)**
 - 64 nodes total (max 32 allowed at one time)
 - 360 Mflop/node peak (R10000 processors)
 - 128 Mb/node
- **SGI/Cray Origin 2000 (NCSA)**
 - 128 nodes total (max 64 allowed at one time)
 - 390 Mflop/node peak
 - 256 Mb/node

MODEL PROBLEMS

● Comparison of solvers based on two hypothetical model problems

● Model Problem 1: Single extraction well remediating a cylindrical plume

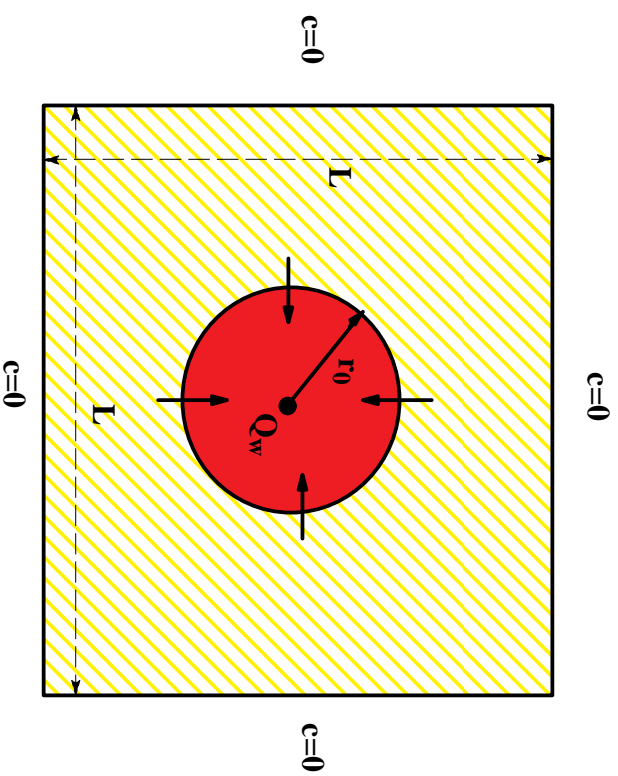
Used for tests involving scalability, parallel performance, and effect of problem size

Size varied from 61x61x11 (1 processor) to 1921x1921x11 (1024 processors)

● Model Problem 2: Contamination from a rectangular patch source used for tests involving roughness of coefficients, Courant and Peclet numbers

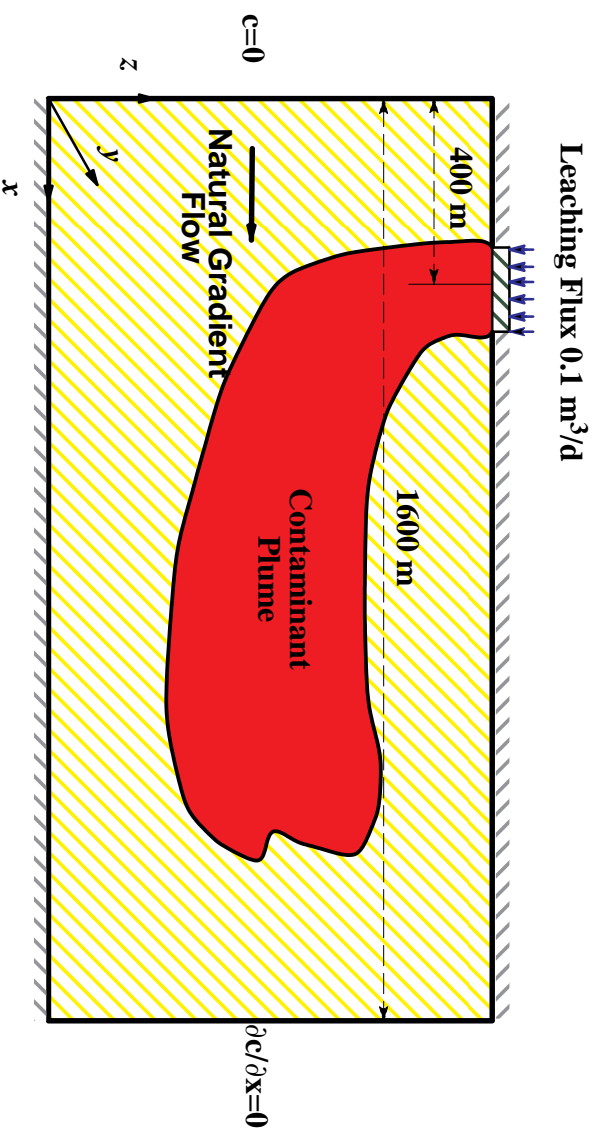
Size fixed at 801x401x21 using 242 processors ($mn=6.75M$, $np=242$)

MODEL PROBLEM 1



Plan view of Model Problem 1
(for scalability and performance tests)

MODEL PROBLEM 2



Vertical Cross-Section of Model Problem 2

BASIC MODEL PARAMETERS

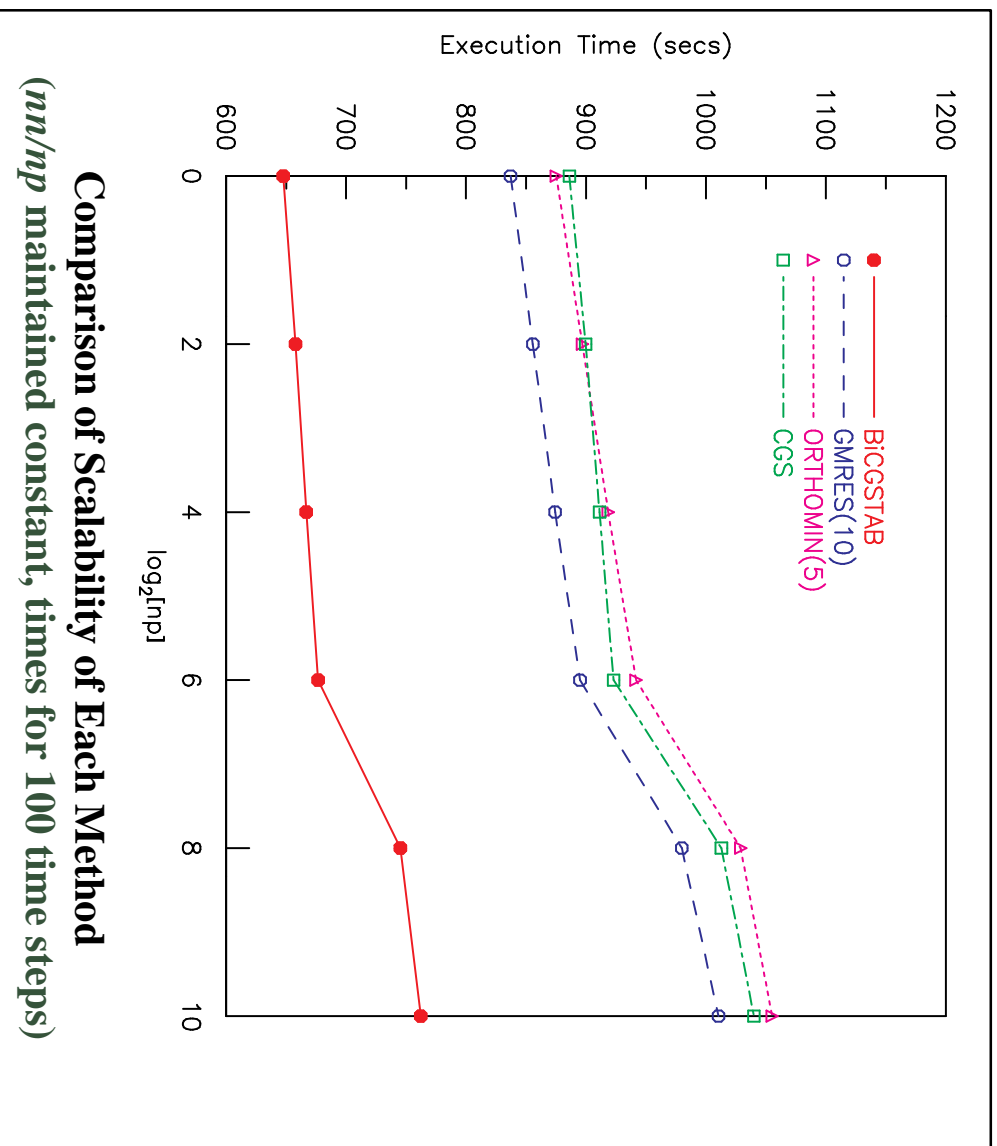
Parameters related to p.d.e

- Crank–Nicolson time stepping ($\omega=0.5$)
- Upstream weighting ($\alpha=0.5$)
- Constant Material Properties:
 - Porosity $\theta = 0.3$
 - Bulk density $\rho = 1.0$
 - Decay coefficient $\lambda = 0.005$
 - Adsorption distribution coefficient $K_d = 1.0$

Parameters related to solvers

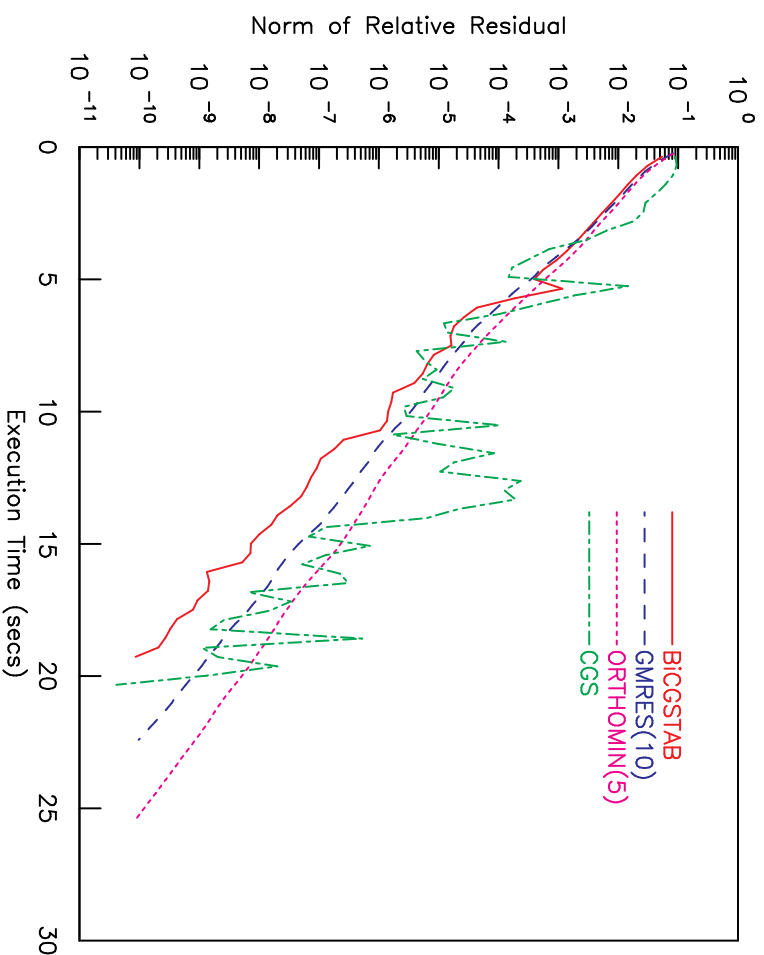
- Convergence tolerance = $1.e-10$ (two-norm of relative residual)
- Restart parameter for GMRES = 10, ORTHOMIN = 5

SCALABILITY



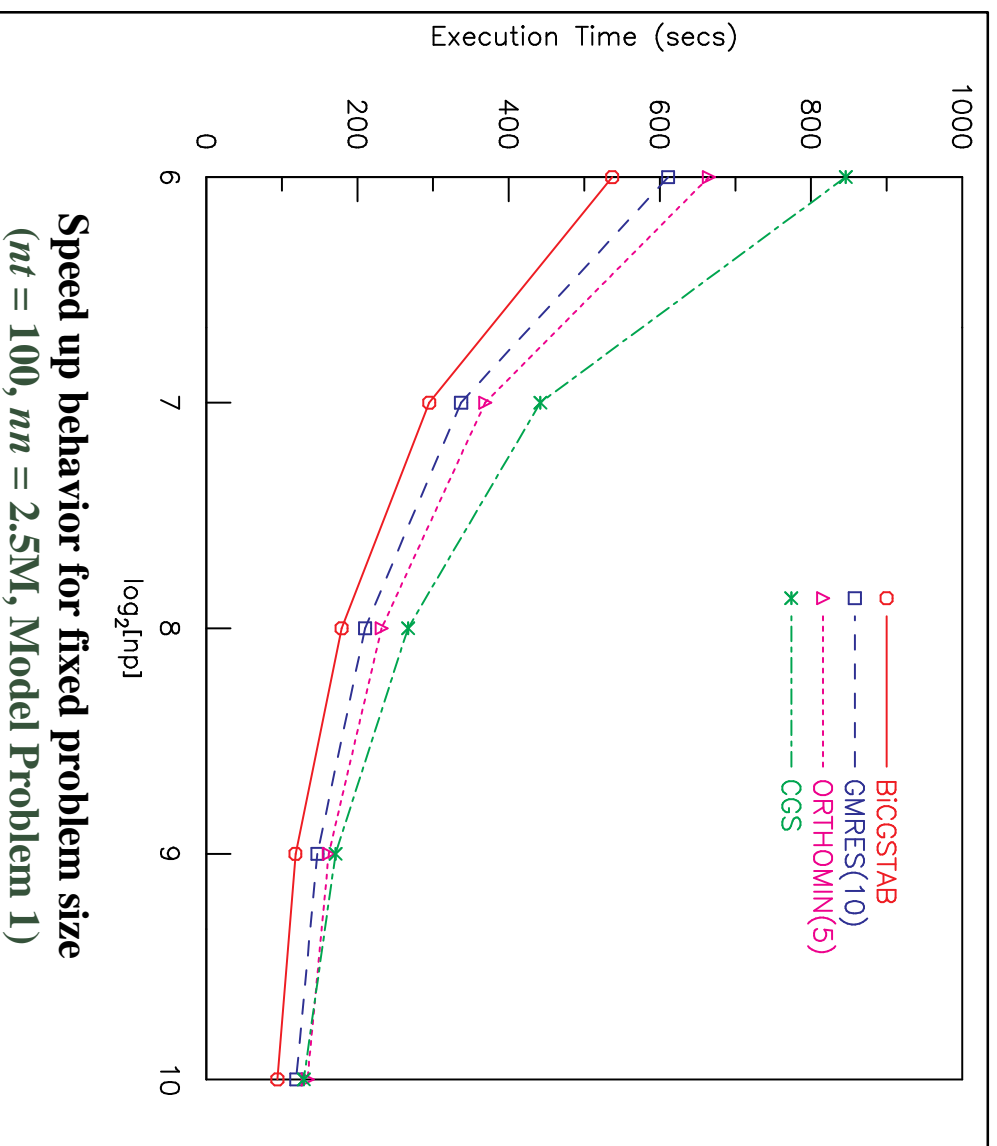
Comparison of Scalability of Each Method
(*mpi* maintained constant, times for 100 time steps)

CONVERGENCE BEHAVIOR



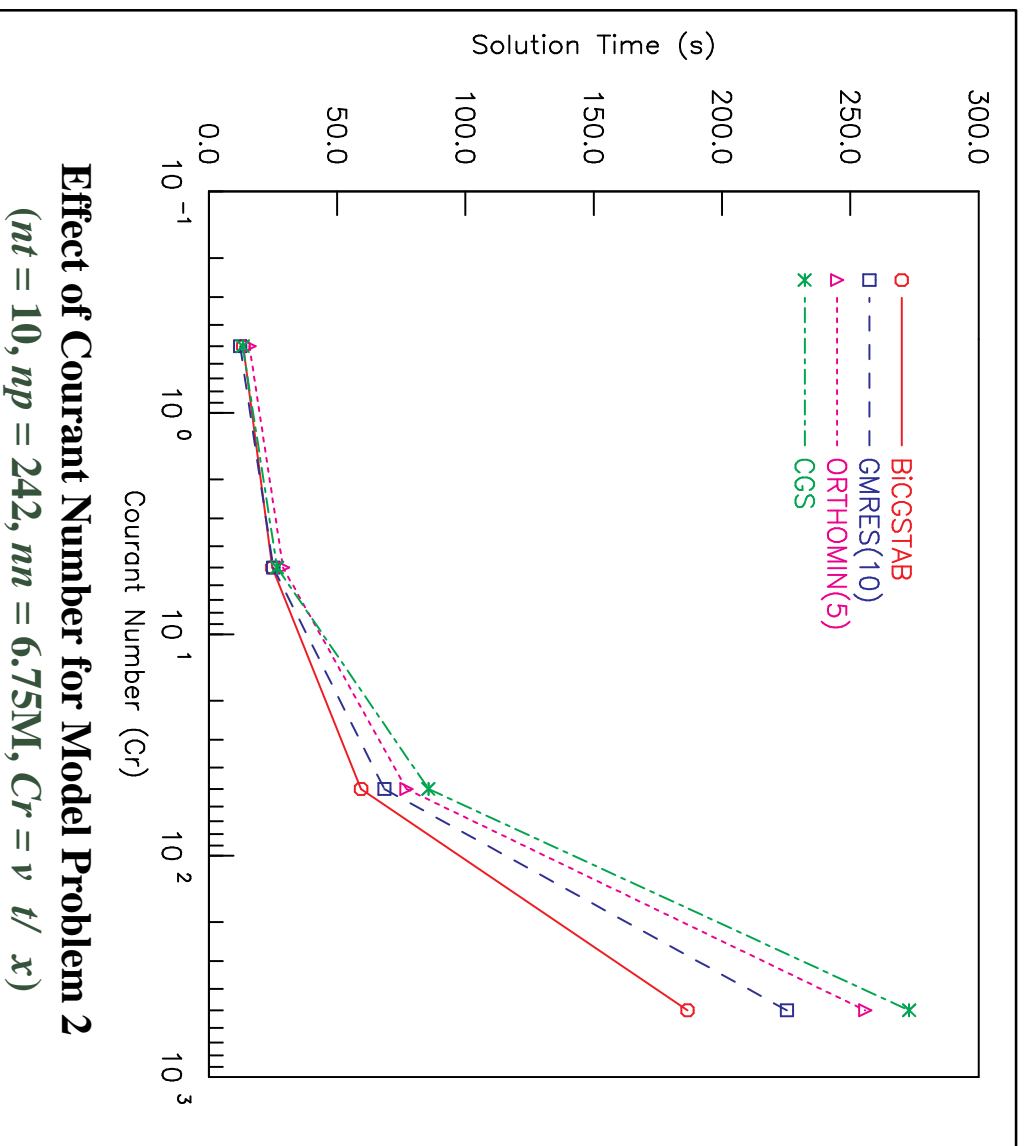
Convergence behavior for Model Problem 2
($np = 242$, first time step corresponding to $\tau = 4.0$)

SPEED UP BEHAVIOR FOR FIXED PROBLEM SIZE



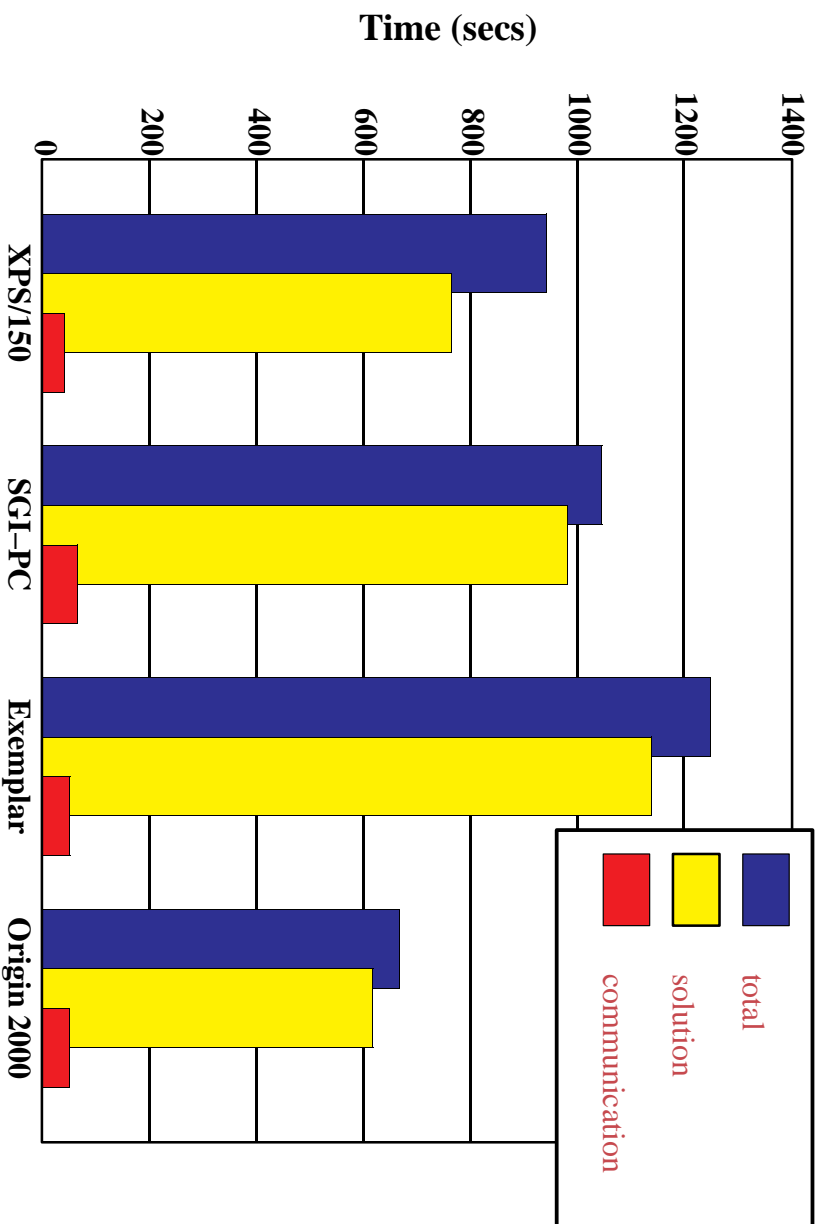
Speed up behavior for fixed problem size
($nt = 100$, $nm = 2.5M$, Model Problem 1)

EFFECT OF COURANT NUMBER



Effect of Courant Number for Model Problem 2
($mt = 10$, $np = 242$, $m = 6.75M$, $Cr = v \ t / x$)

COMPARISON AMONG PARALLEL PLATFORMS



Performance of BiCGSTAB on four parallel platforms
($mp = 16$, $m = 640K$, $nt = 100$)

EFFECT OF K-FIELD HETEROGENEITY

- The K-field heterogeneity () for the flow equation was increased from 0 to 4 resulting in increased variability in the velocity field; i.e., rough coefficients for the transport equation

	BIGSTAB	GMRES(10)	OMIN(5)	CGS				
	Iter	Time	Iter	Time	Iter	Time	Iter	Time
0.0	109	40.2	195	47.5	194	51.4	113	41.1
1.0	257	91.6	454	109.1	457	119.8	298	103.7
2.0	356	125.8	648	155.1	649	165.3	479	164.8
3.0	449	157.7	807	189.7	813	206.1	664	227.4
4.0	515	180.8	861	219.5	985	249.1	786	268.9

Effect of K-field Heterogeneity

(= degree of heterogeneity, $mn = 6.75M$, $np = 242$, $nt = 10$)

- Convergence behavior of all methods deteriorate with increasing

OTHER EFFECTS

- **Very little impact on convergence as problem size was increased. This is contrary to the steady state flow problem where the convergence behavior of conjugate gradient deteriorated as the problem size was increased.**
- **No significant effect of Grid Peclet Number ($Pe = x / L$) on convergence behavior**
- **No significant effect of K_d -field variability on convergence behavior. This is different from the result of K -field variability.**

FLOATING POINT PERFORMANCE

- Floating point performance for all the methods were in the range of 10–15 Mflops per processor on the Intel Paragon XPS/150. This is within the normal range for sparse matrix applications.
- The explicit communication times on XPS/150 was on the order of 5–10% of the total time. The communication time on other architectures were slightly higher.

CONCLUSIONS

- All methods exhibit good scalability up to 1024 processors. Largest problem consisted of more than 40 million nodes and took less than 3 seconds per time step.
- BICGSTAB performs slightly better than other methods for most problems tested.
- The performance of all the methods are within a factor of two of each other.
- The convergence behavior did not deteriorate as problem size was increased.
- The iterations per time step was in the order of < 20 iterations for most problems tested using BICGSTAB.
- We see very little advantage in using a method such as multigrid for this problem because the Krylov methods already perform very well.

ACKNOWLEDGEMENTS

This work was sponsored by the Center for Computational Sciences of the Oak Ridge National Laboratory managed by Lockheed Martin Energy Research Corporation for the U.S. Department of Energy under contract number DE-AC05-96OR22464.

The authors gratefully acknowledge the use of High Performance Computing Facilities at the Center for Computational Sciences and the National Center for Supercomputing Applications.