

Use of Parallel SuperLU in Large-Scale Scientific Calculations

X. Sherry Li

Lawrence Berkeley National Lab

June 11, 2003

Outline

- Algorithms and implementation in SuperLU_DIST
 - Compare with sequential SuperLU
- In quantum mechanics (linear systems)
 - With M. Baertschy, C. W. McCurdy, T. N. Rescigno, W. A. Isaacs
- In accelerator design (eigenvalue problems)
 - With P. Husbands, C. Yang

SuperLU (_MT)

[Demmel/Gilbert/Li]

- Partial pivoting
 - With diagonal preference
- Sparsity order: $A' * A$ -based
 - Fix column order
- “BLAS-2.5”
- Left-looking (fan-in)
 - More “read” than “write”

SuperLU_DIST

[Li/Demmel]

- “Static” pivoting
 - Based on A
- Sparsity order: $A' + A$ -based
 - Fix row & column order
- BLAS-3
- Right-looking (fan-out)
 - More parallelism ?
 - 2D block-cyclic layout

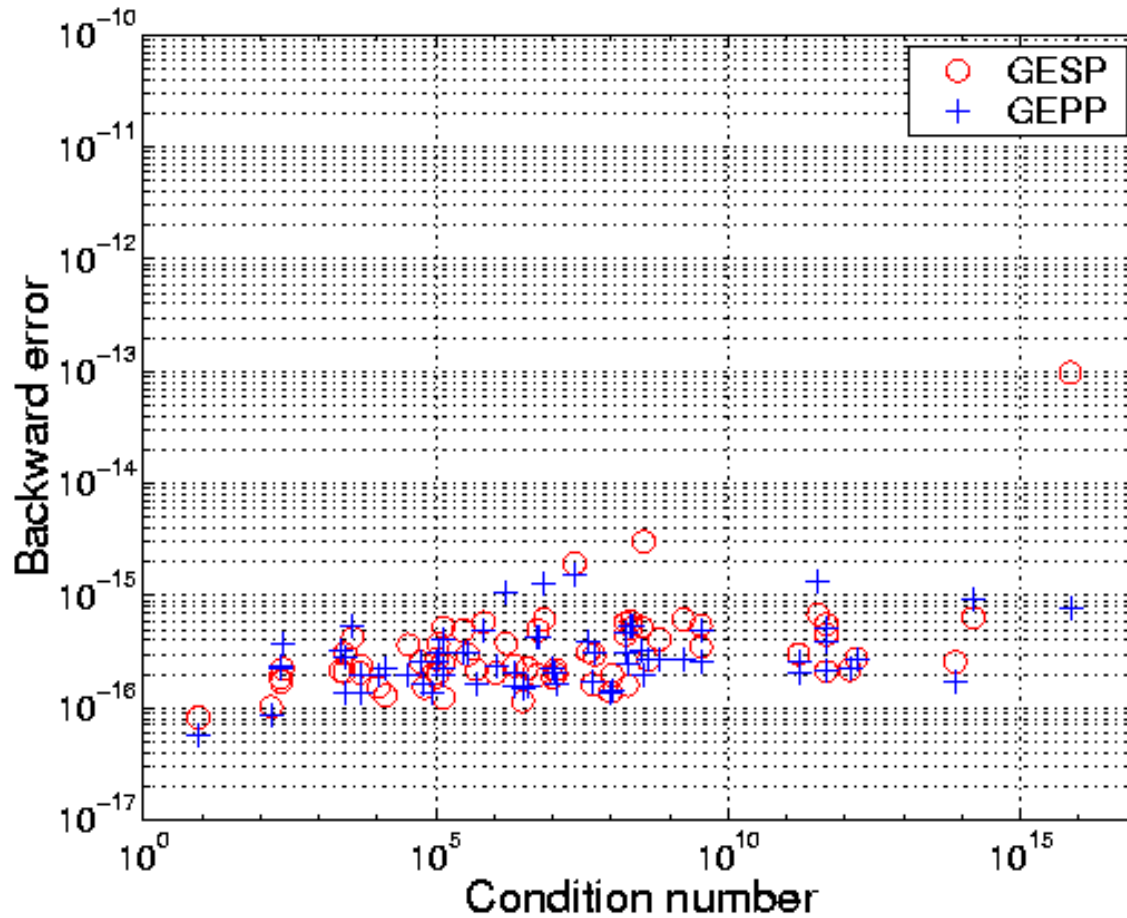
GE with Static Pivoting

- Before factorization, scale and permute A to maximize diagonal: $A \leftarrow P_r D_r A D_c$
 - MC64 [Duff/Koster '99]
 - Parallel auction algorithm on the way [Riedy]
- Find a permutation P_c to preserve sparsity:
 $A \leftarrow P_c A P_c'$
- While factorizing $A = LU$, replace tiny pivots by $\sqrt{\varepsilon} \|A\|$, without changing structures of L & U
 - Most matrices insensitive to this perturbation
- If needed, use iterative refinement or an iterative solver to improve first solution

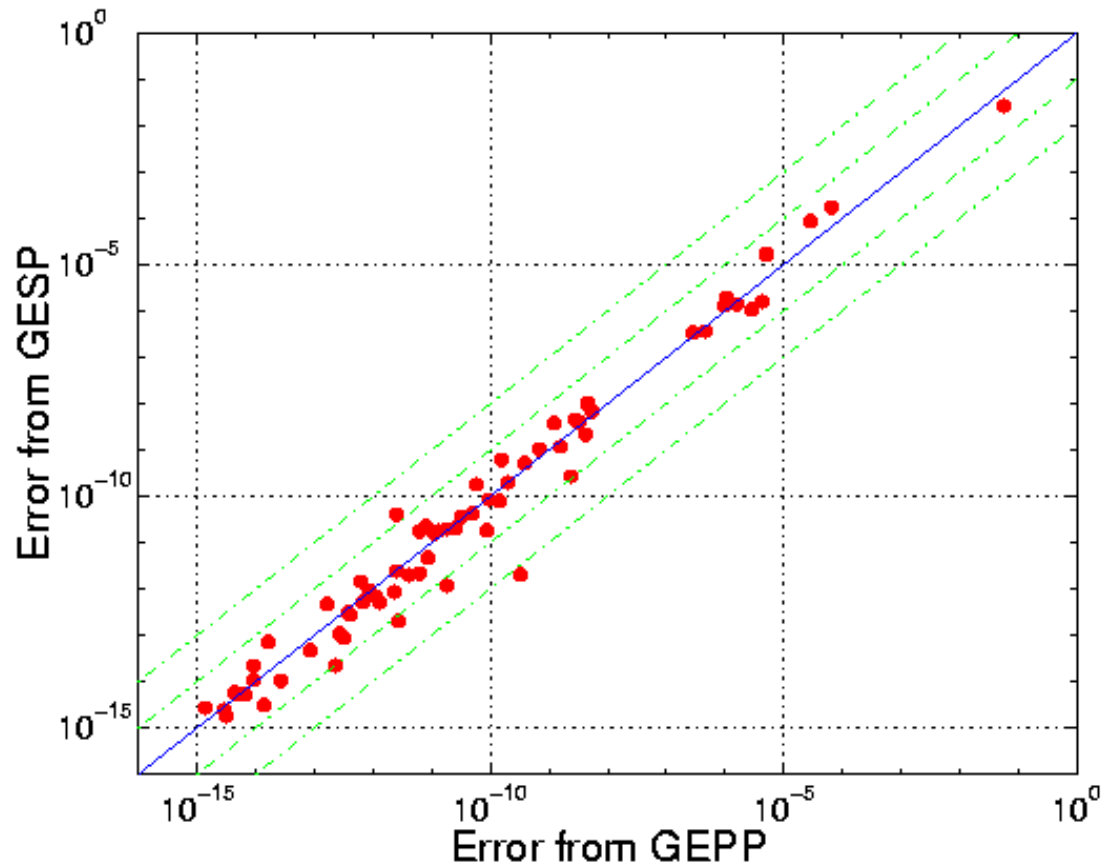
Numerical Results

- 68 unsymmetric matrices
- If no pivoting at all
 - 26 contain zeros on diagonal to begin with
 - 2 create new zeros on diagonal during elimination
 - o bbmat, orsreg_1
 - Most of the others get large errors due to pivot growth
- Working precision iterative refinement (64 bits)
- Detailed table at www.nersc.gov/~xiaoye/SuperLU/GESP

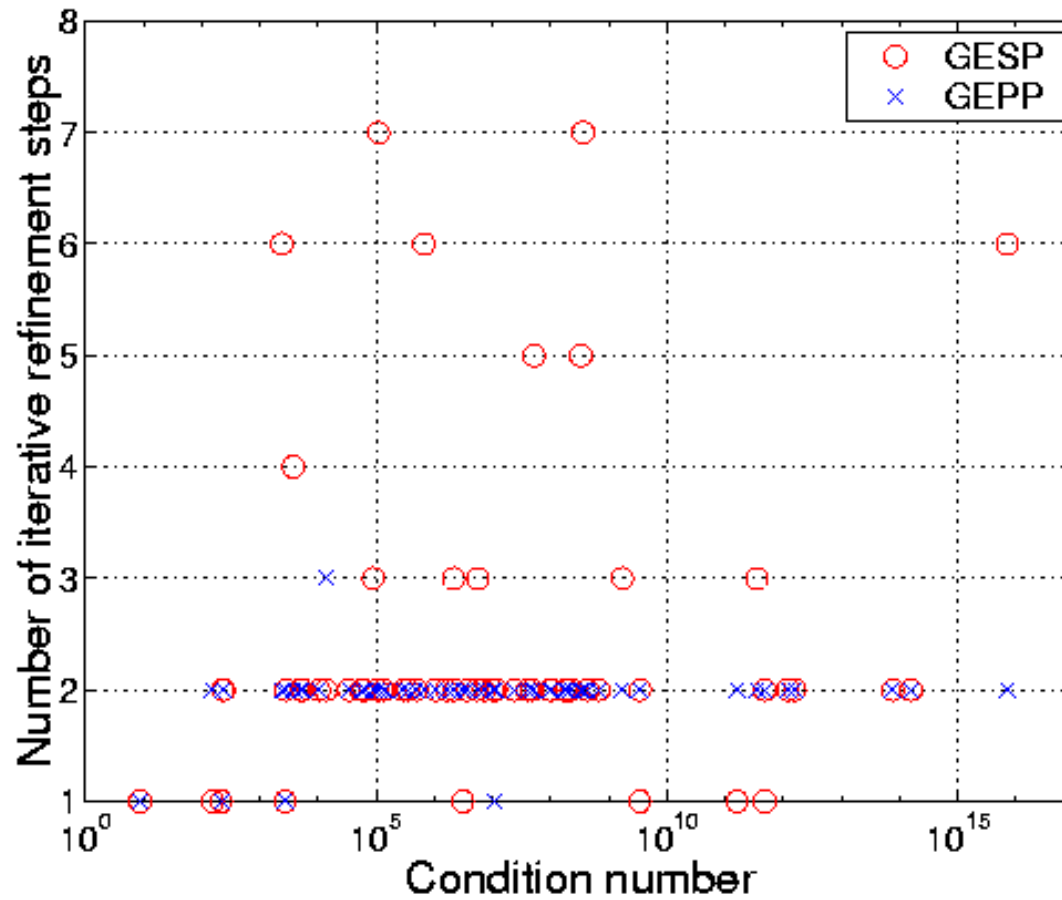
Sparse Backward Error



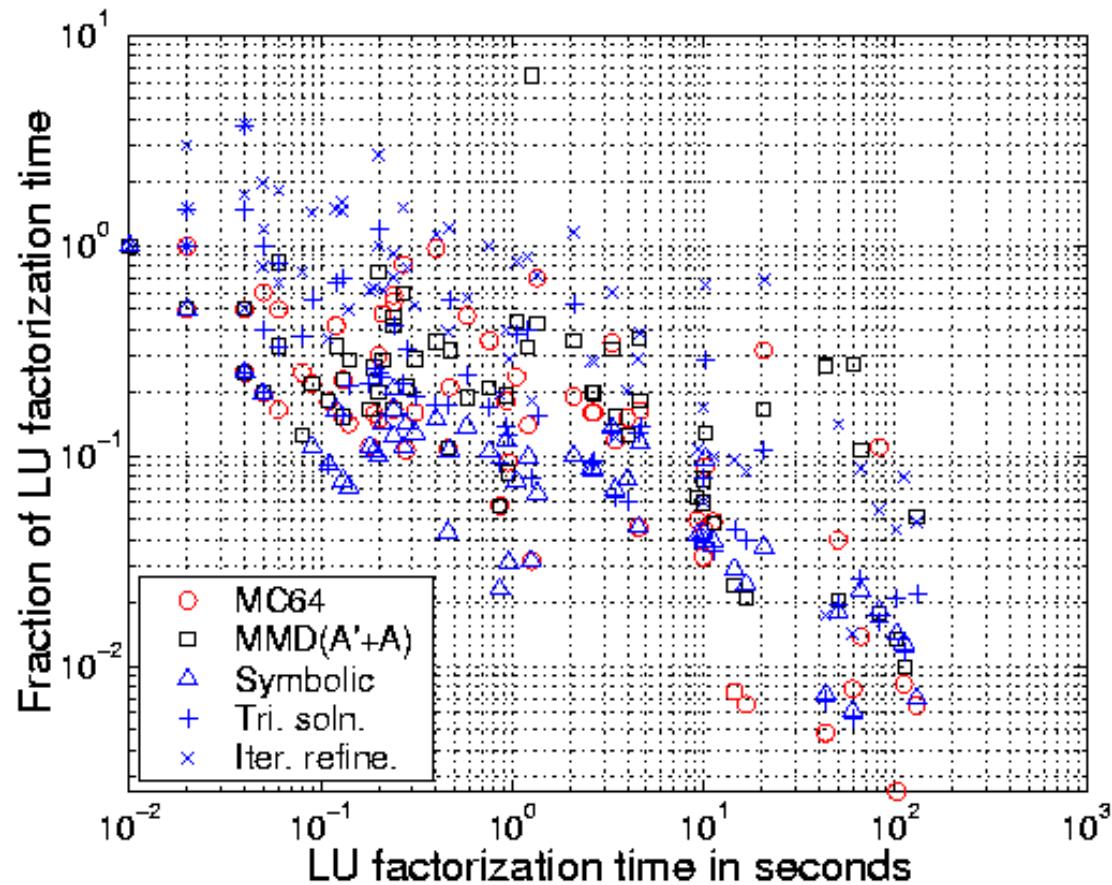
Accuracy: GESP vs GEPP



Refinement Steps



Times of Various Phases



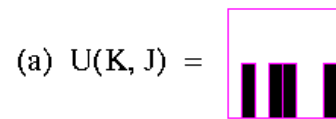
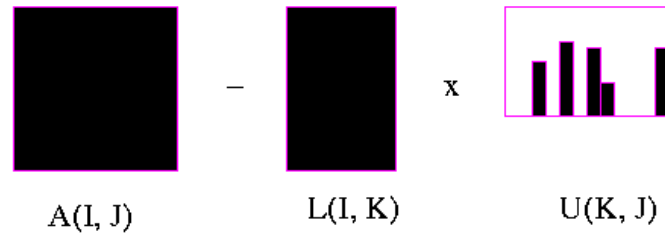
Better Sparsity Ordering

- SuperLU_DIST: fill-ins in L+U (10^6)

Matrix	COLAMD	MMD(A'+A)
BBMAT	49.1	41.1
ECL32	72.6	42.4
INV-EXTRUSION-1	62.7	29.1
MIXING-TANK	81.4	40.7
TWOTONE	18.3	11.4

- Diagonal Markowitz (**based on A**): 10-15% better
[Amestoy/Li/Ng '01]

BLAS 3 Kernel



- Copying is almost free
- 20-40% better than BLAS 2.5 on IBM SP

Default Setting May Not Be Good

- Diagonal perturbation is bad for 4 matrices
 - Ex11, fidap011, inaccura, raefsky4
- Work well by either one of the 2 methods
 - Turn off diagonal perturbation
 - Use LU as preconditioner for GMRES [SPARSKIT]

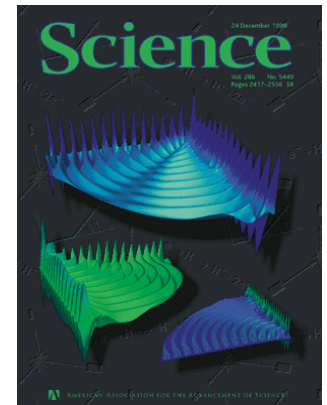
Matrix	N	Diag Perturbs	GMRES Steps
EX11	16614	8666	497
FIDAP011	16614	8602	530
INACCURA	16146	3	5
RAEFSKY4	19779	51	41

Some Open Problems

- Parameter setting is configurable by users
- Need an automatic strategy to choose a proper configuration
 - Run the default configuration first (mostly works)
 - Compute forward & backward error bounds
 - If large errors, invoke an iterative solver
- How does extra-precision iterative refinement help?
 - Investigating dense LU using XBLAS
- Guaranteed stability if variable precision is used during GE [Demmel]

Quantum Mechanics

- Scattering in a quantum system of three charged particles
- Simplest example is ionization of a hydrogen atom by collision with an electron: $e^- + H \rightarrow H^+ + 2e^-$
- Seek the particles' wave functions represented by the time-independent Schrodinger equation
- First solution to this long-standing unsolved problem [Recigno, et. al. Science, 24 Dec 1999]



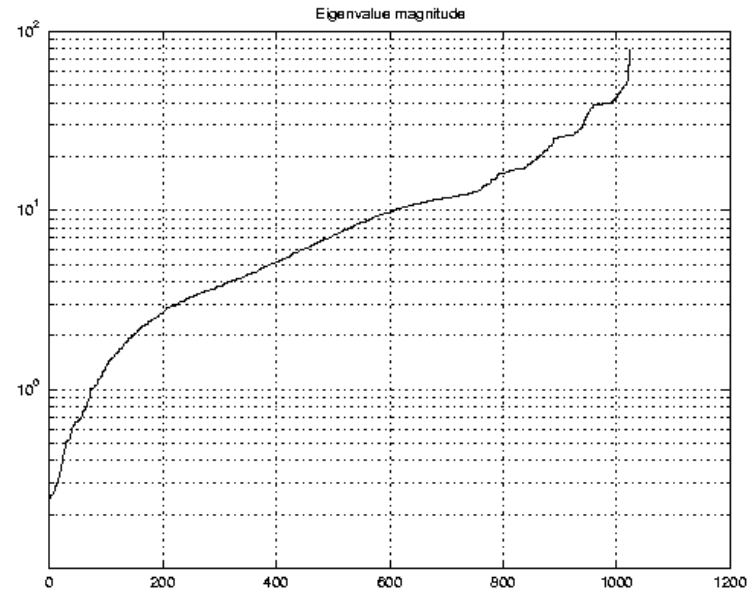
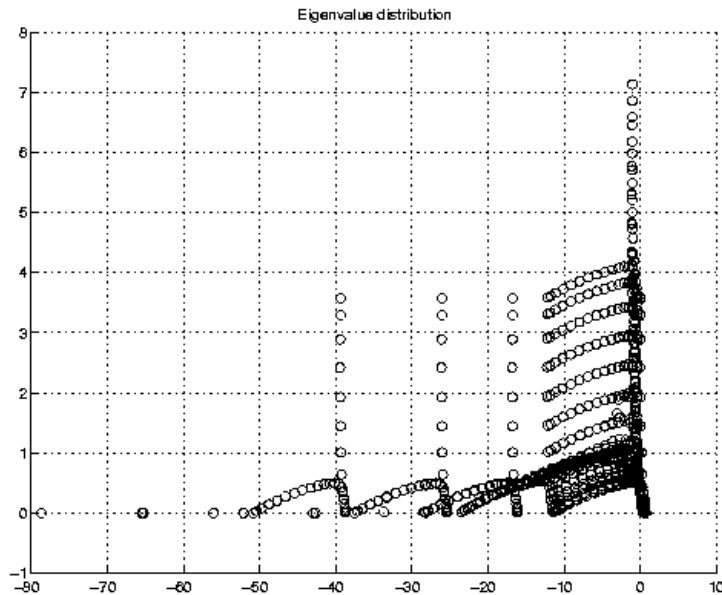
Quantum Mechanics, cont.

- New *Exterior Complex Scaling* formalism to represent scattering states which simplifies boundary conditions
 - Becomes computationally tractable
- Finite difference leads to **complex, unsymmetric** systems
 - Diagonal blocks have the structure of 2D finite difference Laplacian matrices
 - Very sparse: nonzeros per row ≤ 13
 - Off-diagonal block is a diagonal matrix
 - Between 6 to 24 blocks, each of order between 200K and 350K
 - Total dimension up to 8.4 M
- Too much fill if use direct method

A_{11}				...
	A_{22}			...
		A_{33}		...
			A_{44}	...
⋮	⋮	⋮	⋮	⋮

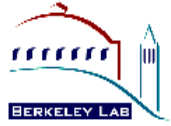
Matrix Problem

- Example: $N = 1024$, $NNZ = 12544$, Cond. Num. $4e+03$



- No iterative solvers without preconditioners or with simple preconditioners converge

SuperLU_DIST as Preconditioner



- SuperLU_DIST as block-diagonal preconditioner for CGS iteration

$$M^{-1}A x = M^{-1}b$$

$$M = \text{diag}(A_{11}, A_{22}, A_{33}, \dots)$$

- Run multiple SuperLU_DIST simultaneously

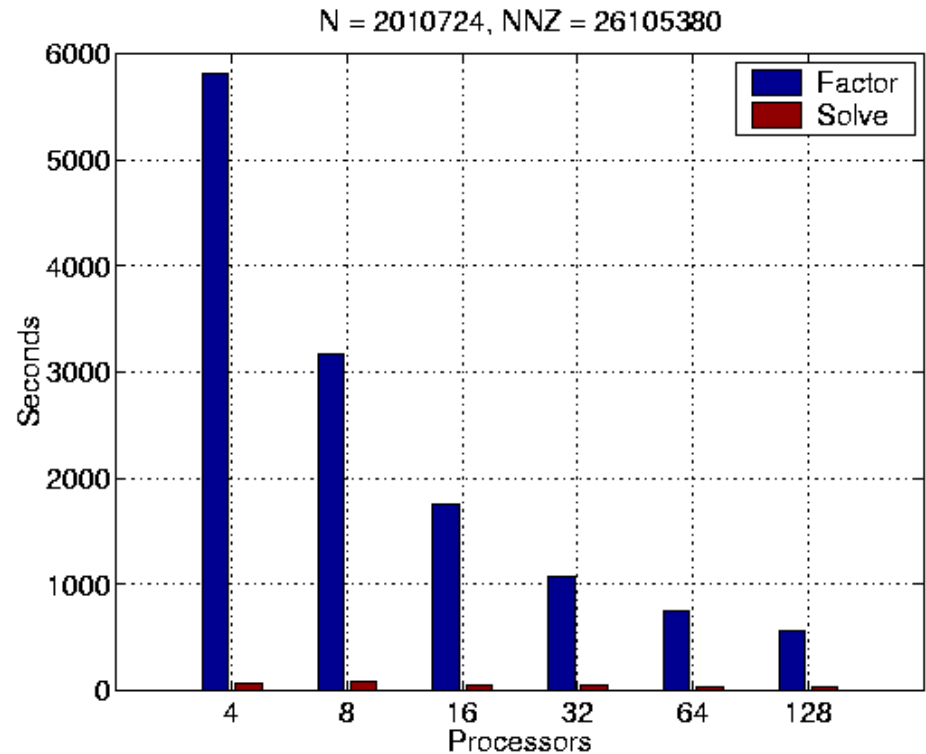
- **No pivoting, nor iterative refinement**

- 96 to 280 iterations @ 1 ~ 2 minute/iteration using 64 IBM SP processors

→ **Total time ~ 1 to 8 hours**

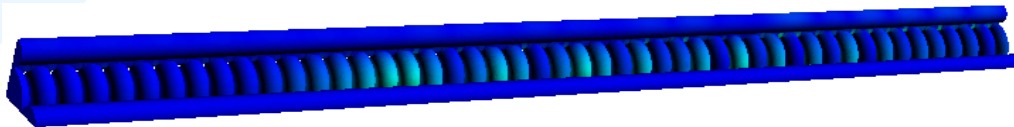
Timings on IBM SP

- $N = 2\text{ M}$, $NNZ = 26\text{ M}$
- Fill-ins using Metis: 1.3 G (50x fill)
- Factorization speed
 - 10x speedup (4 to 128 P)
 - Up to 30 Gflops



Accelerator Cavity Design

- Calculate cavity mode frequencies and field vectors
- Solve Maxwell equation in electromagnetic field
- Omega3P simulation code



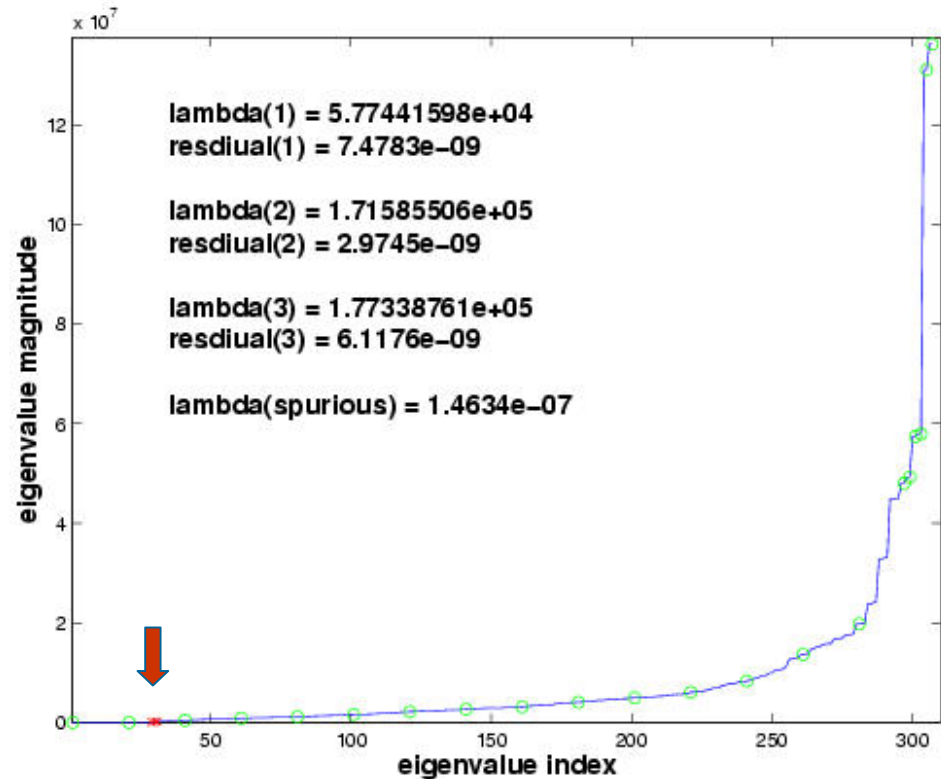
Omega3P model of a 47-cell section of the 206-cell Next Linear Collider accelerator structure



Individual cells used in accelerating structure

Accelerator, cont.

- Finite element methods lead to large sparse generalized eigensystem $Kx = \lambda Mx$
- Real symmetric for lossless cavities
Complex symmetric when lossy in cavities
- Seek interior eigenvalues (tightly clustered) that are relatively small in magnitude



Accelerator, cont.

- Speed up Lanczos convergence by shift-invert
 - ➔ Seek largest eigenvalues, well separated, of the transformed system

$$M (K - \sigma M)^{-1} x = \mu M x$$

$$\mu = 1 / (\lambda - \sigma)$$

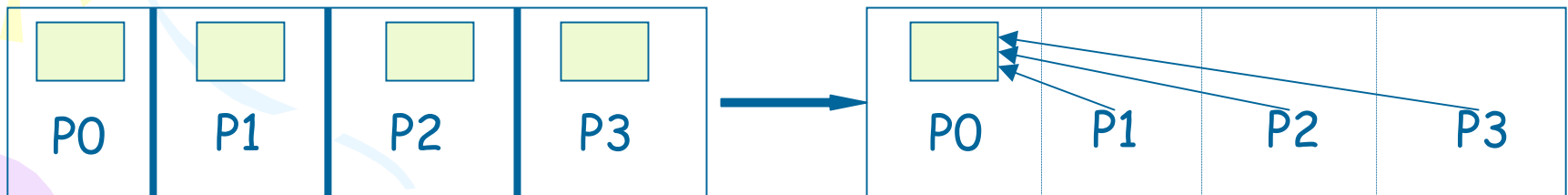
- The Filtering algorithm [Y. Sun]
 - Inexact shift-invert Lanczos + JOCC
- We added exact shift-invert Lanczos (ESIL)
 - PARPACK for Lanczos
 - SuperLU_DIST for shifted linear system
 - **No pivoting, nor iterative refinement**

SuperLU_DIST Enhancements

- 64-bit addressing
- Some integers need to be 64-bit for the largest problem (integer overflow)
- Will large memory SMP nodes help us?
 - IBM SP nodes
 - 16-way 375MHz POWER3
 - GB/node: 16, 32 (64 nodes) ,64 (4 nodes)
 - "Colony" switch

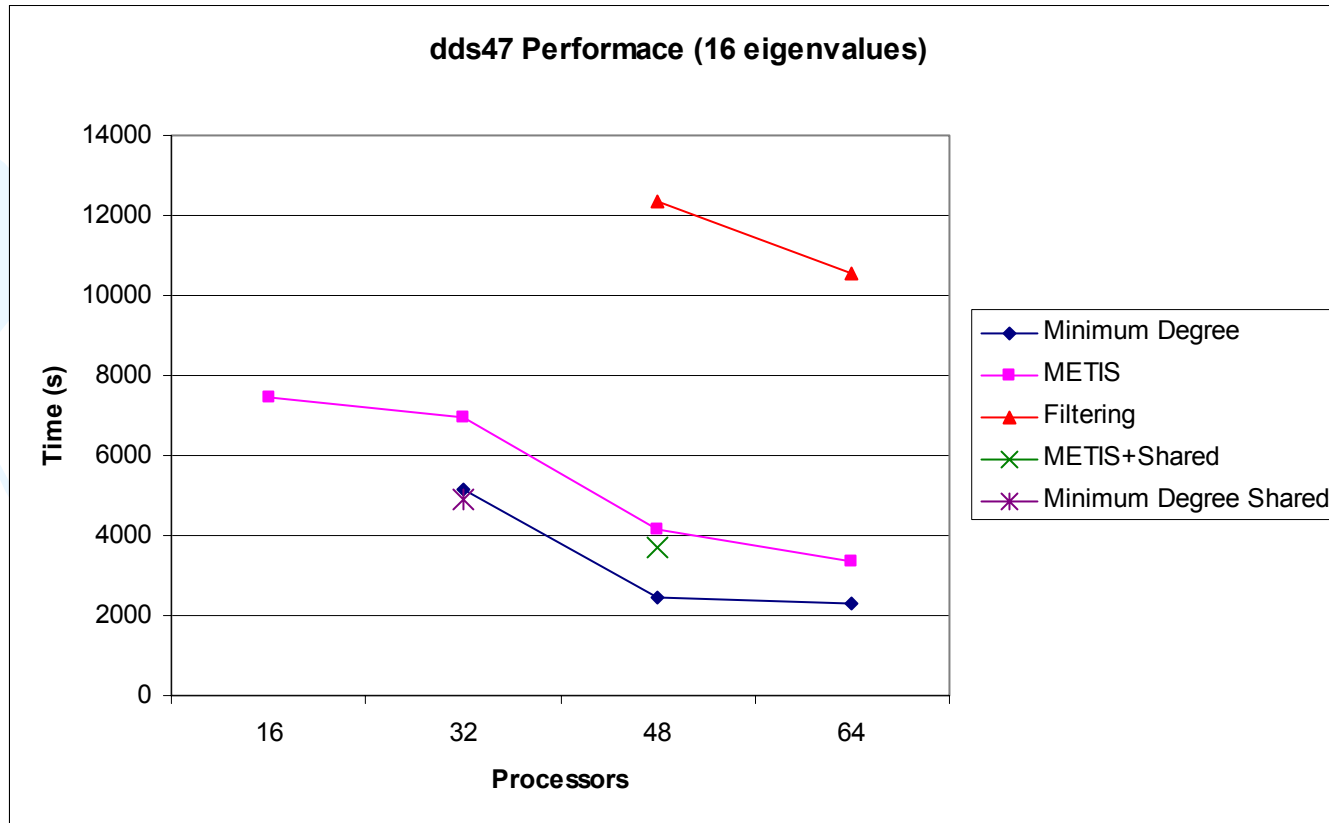
Use of Shared Memory

- Symbolic factorization (7 GB for largest problem) still serial and replicated on all processors
 - Save communication by not having to broadcast output
- Better to perform 1 `symbfact()` per SMP node and share the results [Husbands]
 - Also used in High Performance Linpack



DDS47, Linear Elements

➤ Total eigensolver time: $N = 1.3 \text{ M}$, $NNZ = 20 \text{ M}$



Largest Problem

➤ DDS47, quadratic elements

- $N = 7.5 \text{ M}$, $NNZ = 304 \text{ M}$
- 6 G fill-ins using Metis

➤ 24 processors (8x3)

- Factor: 3,347 s
- 1 Solve: 61 s
- Eigensolver: 9,259 s (~ 2.5 hrs)
 - o 10 eigenvalues, 1 shift, 55 solves

➤ Future: $N=22.3 \text{ M}$, $NNZ = 475 \text{ M}$, too big?

Future Work

- Parallel symbolic factorization to enhance memory scalability
- Better triangular solve (latency-bound)
- Include ILU capability
- Optimizations for SMP clusters
 - Hybrid approach with threads in each MPI task