SPE-172984-MS

Spectral Effects of Large Matrices from Oil Reservoir Simulators on Performance of Scalable Direct Solvers

Ahmet Duran and Mehmet Tuncel, Istanbul Technical University, Department of Mathematics, Istanbul 34469, Turkey; Istanbul Technical University, National Center for High Performance Computing of Turkey (UHeM), Istanbul 34469, Turkey

This paper was prepared for presentation at the SPE Large Scale Computing and Big Data Challenges in Reservoir Simulation Conference and Exhibition held in Istanbul, Turkey, 15–17 September 2014.

Abstract

It is important to estimate the elapsed time to solve large sparse linear systems for time-restricted real life decision making applications such as oil and gas reservoir simulators. Challenging matrices should be distinguished and handled separately because they may lead to performance bottleneck. Therefore, we need to examine the spectral effects of large matrices on the performance of scalable direct solvers by using eigenvalues. In this work, we check whether there is relationship between the eigenvalue distribution of a matrix and the performance of the solver. We try to examine the eigenvalue distribution of various sparse matrices. We may find all eigenvalues in order to obtain the distribution graph of eigenvalues, if possible. However, it is very expensive to find all eigenvalues. Therefore, Gerschgorin's theorem may be used to bound the spectrum of square matrices. Several behaviors such as being disjoint, overlapped or clustered of Gerschgorin circles may give clue regarding the distribution of the eigenvalues and the performance of the solver for that matrix.

In this paper, we consider a portfolio of test matrices which include randomly populated sparse matrices and various patterned matrices coming from reservoir modeling from single porosity single permeability to dual porosity dual permeability models (see [10]). We examined our modified HELM2D03LOWER_20K matrix and EMILIA_923 matrix from the University of Florida sparse matrix collection (see [17]), in addition to the patterned matrices from 3 phase black-oil model and 7 component EOS model.

We define an optimal minimum number of cores as the number of cores that provides the minimum wall clock time for a given size of problem, where a right match occurs between the problem size, the spectral effects of matrix and the available resources such as memory, in presence of communication overhead. We find that the optimal minimum number of cores required depends on the sparsity level and size of the matrix. As the sparsity level of matrix decreases and the order of matrix increases, we expect that the optimal minimum number of cores increases slightly.

Introduction

We design a novel hybrid algorithm and solver for large sparse linear systems. First, we consider scalable direct solvers because of their robustness and examine the SuperLU_DIST 3.3 (see Li et al. [1]) for distributed memory parallel machines among several sparse direct solvers (see Li et al. [1], Li and Demmel [2], Amestoy et al. [3], Schenk and Gartner [4, 5], Duran and Saunders [6], Duran et al. [7] and references contained therein). Duran et al. [8] discussed the advantages and limitations of the SuperLU solvers and tested the code of SuperLU_DIST 3.0 (see Li et al. [1]) in order to measure the performance scalability for various sparse matrices (see [9] for the theoretical foundation regarding the distribution of eigenvalues for some sets of random matrices). SuperLU_DIST needs to be improved for certain types of challenging sparse matrices.

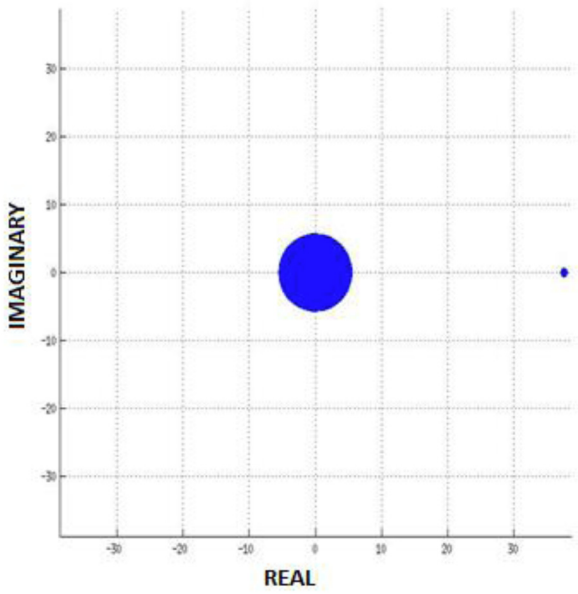

Figure 1—Distribution of eigenvalues for matrix RAND_30K_75

We believe that the approach for exception handling of challenging matrices via Gerschgorin circles is beneficial and practical to stabilize the performance of the solvers. Nearly defective matrices are among the challenging matrices. Clustered eigenvalues observed via Gerschgorin circles may be used to detect nearly defective matrix.

The presence of repeated eigenvalues can be one of the sources of challenges. The repeated eigenvalue may have fewer eigenvectors than the multiplicity of eigenvalue. While such eigenvalue is called defective eigenvalue, the corresponding matrix is referred as a defective matrix (see [12]). If the matrix of eigenvectors is singular, then the matrix cannot be diagonalizable and the matrix is defective. We observe that it takes longer time to solve sparse linear system having defective or nearly defective matrix than regular matrix. Moreover, defective matrix may lead to memory restriction due to the appearance of more fill-ins than that of diagonalizable matrix.

The remainder of this work is organized as follows. First, the test matrices are described. Later, the computation for spectral properties is presented and several illustrative examples are given. The final section concludes this work.

Methods and results

The selected eigenvalues of large matrices are computed using the Scalable Library for Eigenvalue Problem Computations (SLEPc) software (see [13]), which is developed based on the Portable, Extensible Toolkit for Scientific Computation (PETSc) (see [14]). The code has been tested up for all sparse matrices in the list on HP Integrity Superdome SD32B (see [15]), a computing server with shared memory architecture at UHeM (see [18]). The software package includes implementations of a set of methods for the solution of large sparse eigenproblems on parallel computers. It is applicable to both symmetric and nonsymmetric matrices. In our computations, we used the Krylov-Schur method available in the package.

We can compute all eigenvalues of the small randomly populated matrices and show the distribution of eigenvalues for RAND_30K_75 in Figure 1. We observe that nearly all eigenvalues can be found within the circle except for the largest eigenvalue that is indicated by an isolated point in figure. The distribution of eigenvalues for a randomly populated matrix is a good reference for other patterned matrices in order to understand the deviations between them (see [24]). We describe the test matrices in Table 1.

Table 1—Description of the test matrices

Matrices	Order	NNZ	NNZ/N	Origin	Kind of problem
RAND_30K_75	30000	2250000	75	UHeM	Randomly populated
Matrix300k	900000	13362067	14,85	Reservoir simulation	Black-oil model
spe5Ref_dpdp_a	2058000	66808700	32,46	Reservoir simulation	7 component EOS model
spe5Ref_dpdp_b	2058000	71260352	34,62	Reservoir simulation	7 component EOS model
spe5Ref_dpdp_c	2058000	68930222	33,49	Reservoir simulation	7 component EOS model
spe5Ref_dpdp_d	2058000	68930222	33,49	Reservoir simulation	7 component EOS model
spe5Ref_dpdp_e	2058000	67189220	32,65	Reservoir simulation	7 component EOS model
EMILIA_923	923136	40373538	43,74	UFSMC	Geomechanical structural
HELM2D03LOWER_20K	392257	1939353	4,94	UHeM	Patched matrix obtained from HELM2D03
M_UHEM3	1425825	17037638	11,94	UHeM	Patched matrix obtained from parabolic_fem
mC_8M	8000000	39988000	4,999	UHeM	CFD

For the large sparse matrices we compute the extreme eigenvalues. We try to see a rough picture of the distribution for the rest of the eigenvalues by using Gerschgorin's theorem. We show the Gerschgorin's circles of the patched matrix M_UHEM3 (see Duran et al. [16]), five matrices from 7 component EOS model, matrix Emilia_923, and matrix HELM2D03LOWER_20K in Figures 2 – 9, respectively. As the matrix becomes more patterned, the spectral space changes and the eigenvalues take place within disjoint, overlapped or clustered of Gerschgorin circles.

For example, when we examine the spectral properties of HELM2D03LOWER_20K, the real parts of the eigenvalues range between 2.294563 and 4.944602 with many repeated eigenvalues. Those clustered eigenvalues can be observed via Gerschgorin circles. Therefore, HELM2D03LOWER_20K is a nearly defective matrix. We used the SuperLU_DIST 3.3 with tunings of super-nodal storage parameters. However, it runs slowly for the matrix HELM2D03LOWER_20K compared to EMILIA_923, because HELM2D03LOWER_20K is a challenging matrix. It takes approximately 7,5 times longer than EMILIA_923, although HELM2D03LOWER_20K's order, total number of non-zeros and the number non-zeros per row are less than that of EMILIA_923.

SuperLU_MCDT is a distributed direct solver and the software will be uploaded to website (see [17]) after academic permissions from Istanbul Technical University. Here, we used symbolic factorization, ParMETIS (see [11]) for column permutation and Intel MKL (see [20]) as the BLAS library, among several options. The tuning of super-nodal storage parameters is important for the performance and we selected the tuned parameters relax:100 and maxsuper:110 (see [14]).

We define an optimal minimum number of cores as the number of cores that provides the minimum wall clock time for a given size of problem, where a right match occurs between the problem size and the available resources such as memory, in presence of communication overhead (see Duran et. al [23]). We find that the optimal minimum number of cores required depends on the sparsity level and size of the matrix. As the sparsity level of matrix decreases and the order of matrix increases, we expect that the optimal minimum number of cores increases slightly.

Table 2 illustrates the time for the factorisation and the total time for each matrix based on the optimal minimum number of cores. We observe that the optimal minimum number of cores can be different depending on the matrix properties.

We imbedded direct solvers (kernel class) such as SuperLU_DIST 3.3 and SuperLU_MCDT in addition to the solvers provided by OpenFOAM (see [23]). Since future exascale systems are expected to have heterogeneous and many-core distributed nodes, we believe that our SuperLU_MCDT software is a good candidate for future systems. We tested the performance of the solver at TGCC Curie (a Tier-0 system) at CEA, France (see [25] and [23]). SuperLU_MCDT worked up to 16384 cores for the large penta-diagonal matrices for 2D problems and hepta-diagonal matrices for 3D problems, coming from the

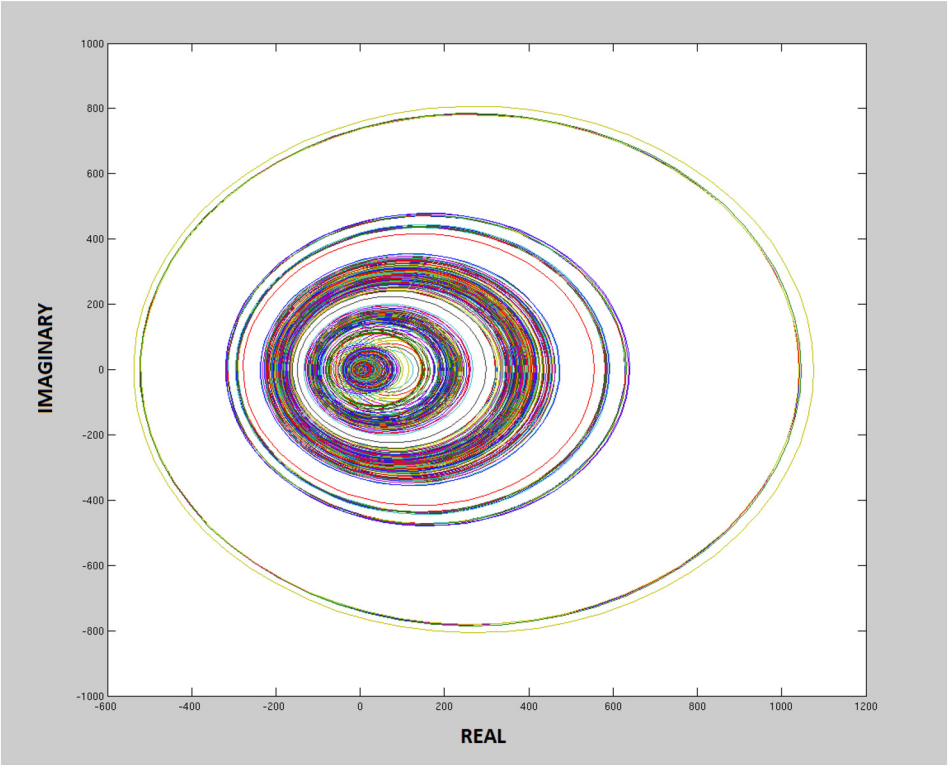

Figure 2—Gerschgorin's circles of M_UHEM3

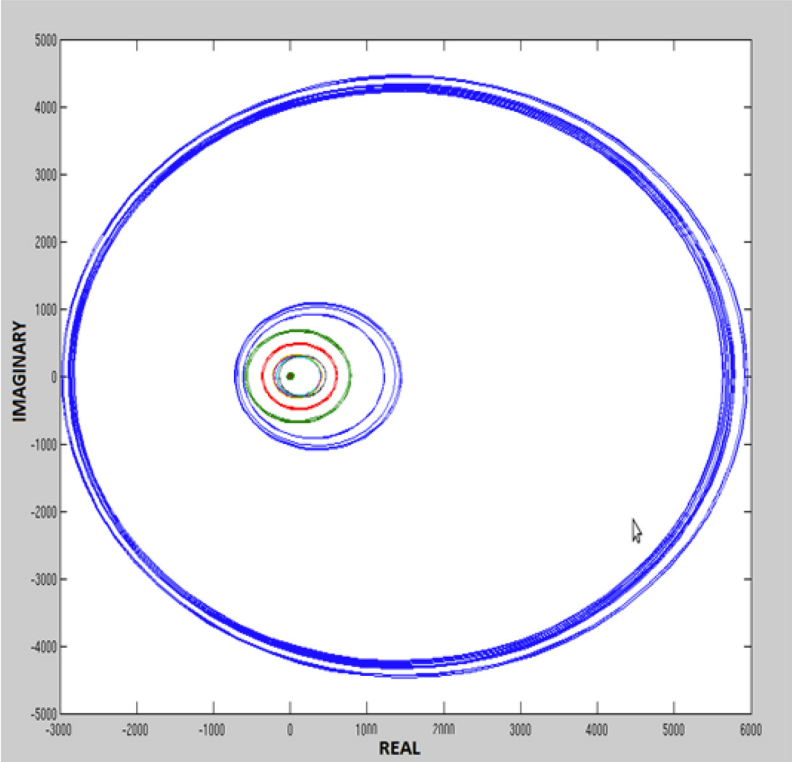

Figure 3—Gerschgorin's circles of spe5Ref_dpdp_a

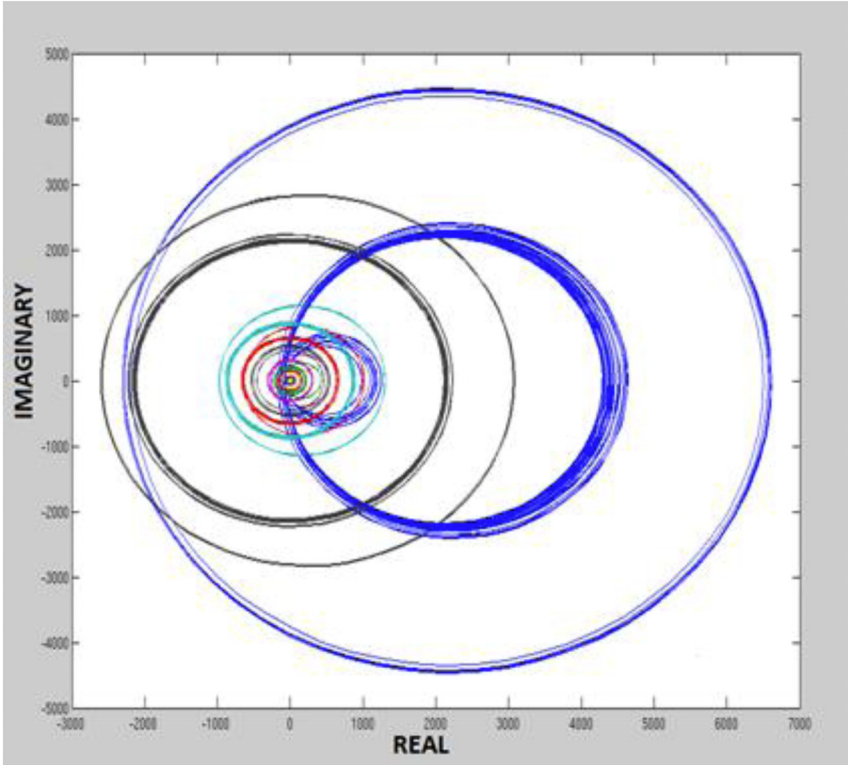

Figure 4—Gerschgorin's circles of spe5Ref_dpdp_b

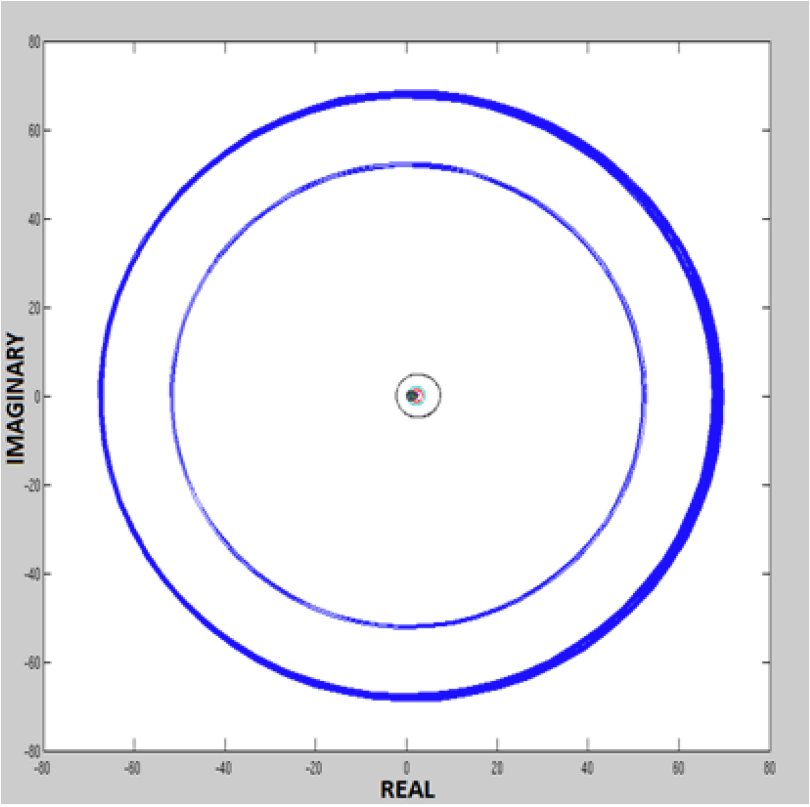

Figure 5—Gerschgorin's circles of spe5Ref_dpdp_c

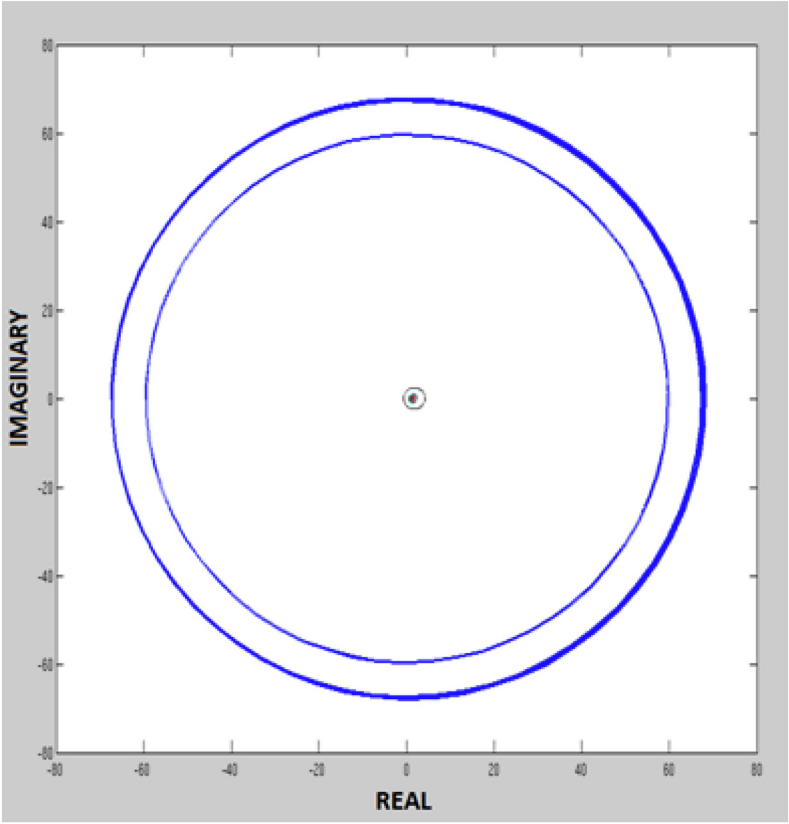

Figure 6—Gerschgorin's circles of spe5Ref_dpdp_d

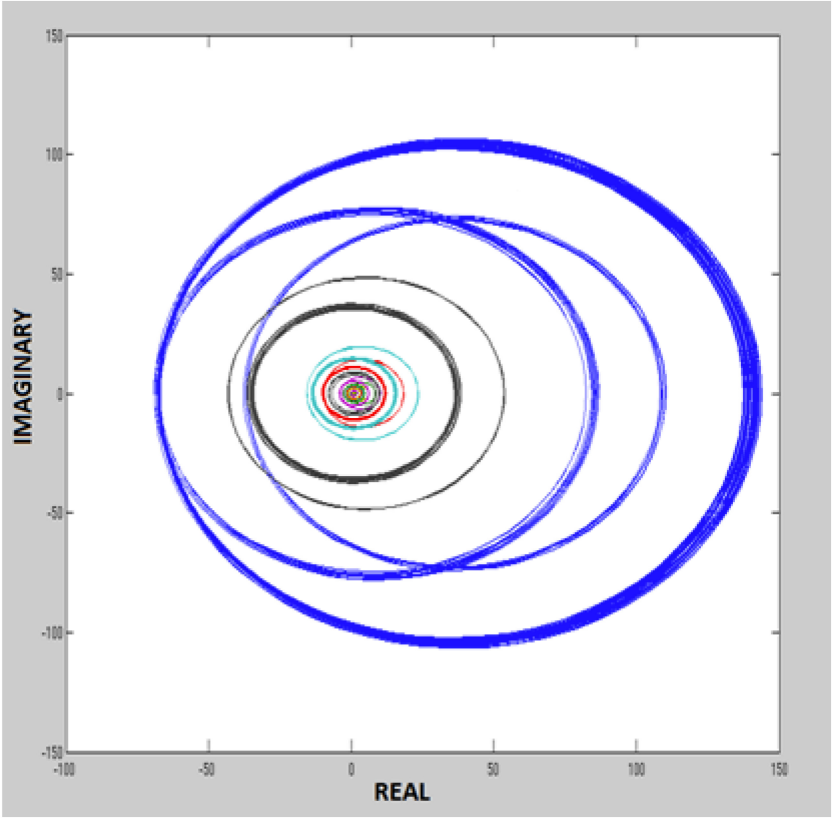

Figure 7—Gerschgorin's circles of spe5Ref_dpdp_e

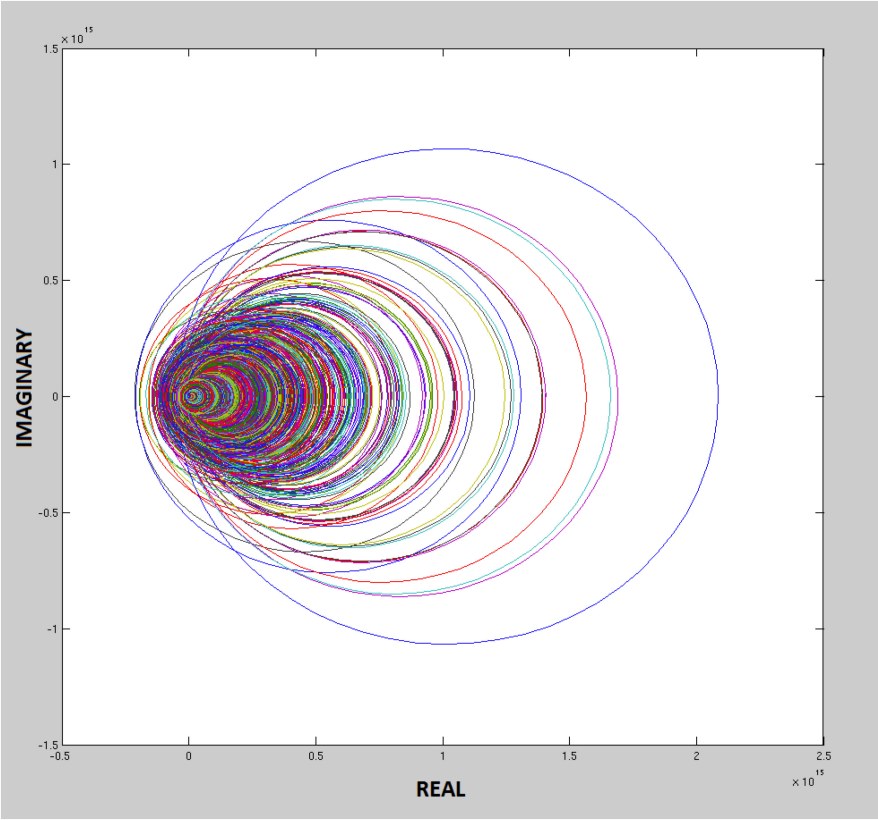

Figure 8—Gerschgorin's circles of matrix Emilia_923

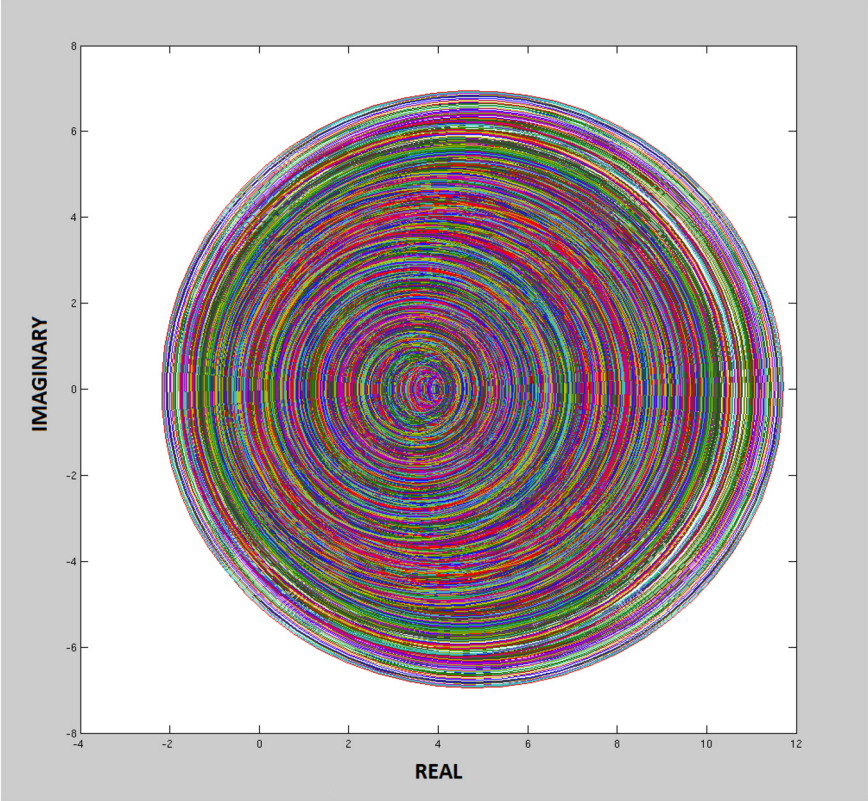

Figure 9—Gerschgorin's circles of matrix HELM2D03LOWER_20K

Table 2—Optimal wall clock times (s) of *SuperLU_MCDT* for the Matrix300k from the black-oil model and five matrices from 7 component EOS model described in Table 1.

Matrices		Optimal time (s)	Optimal minimum number of cores (meshes)
Matrix300k	Factor Time	10,46	1024
	Total Time	24,48	(256×4)
spe5Ref_dpdp_a	Factor Time	52,79	16384
	Total Time	208,27	(4096×4)
spe5Ref_dpdp_b	Factor Time	49,29	16384
	Total Time	220,91	(4096×4)
spe5Ref_dpdp_c	Factor Time	193,60	1024
	Total Time	242,34	(256×4)
spe5Ref_dpdp_d	Factor Time	193,54	1024
	Total Time	242,11	(256×4)
spe5Ref_dpdp_e	Factor Time	51,43	16384
	Total Time	216,49	(4096×4)

Table 3—Distribution of wall clock time (s) for mC_8M matrix using ParMETIS for column permutation, at TGCC Curie (a Tier-0 system) at CEA, France (see [25])

# of cores (mesh)	256 (16 × 16)	512 (16 × 32)	1024 (32 × 32)	2048 (32 × 64)	4096 (64 × 64)	8192 (64 × 128)	16384 (128 × 128)
Nonzeros in L	736867161	**80858737**	759889256	765376719	692260216	700475156	690287571
Nonzeros in U	736867161	**80858737**	759889256	765376719	692260216	700475156	690287571
nonzeros in L+U	1465734322	**160717474**	1511778512	1522753438	1376520432	1392950312	1372575142
nonzeros in LSUB	102386047	**11558966**	106262844	108045660	94662608	97338383	96491385
# of super-nodes	204238	**26847**	207025	208620	215465	214535	217216
Equil time	0,39	0,27	0,53	1,41	2,07	2,23	6,05
RowPerm time	2,18	0,27	2,17	2,18	2,18	2,2	2,17
ColPerm time	5,54	**8,63**	**31,12**	**66,29**	**102,04**	**139,54**	**301,12**
SymbFact time	3,92	0,41	4,07	4,1	3,57	3,66	3,63
Distribute time	1,07	0,24	0,75	0,76	0,69	0,92	1,68
Factor time	9,34	1,79	13,64	13,87	25,33	43,46	90,98
Solve time	3,33	0,01	1,59	1,88	1,59	1,85	2,05
Refinement time	19,76	1,06	7,84	6,59	7,75	8,1	10,85
||X-Xtrue||/||X||	1,18E-012	4,06E-011	1,80E-012	2,35E-012	1,12E-012	1,08E-012	1,10E-012
Total time (s)	45,53	**12,68**	61,71	97,08	145,22	201,96	418,53

incompressible blood flow simulation, without any problem. For example, Table 3 shows the distribution of wall clock time (s) for mC_8M matrix and the impact of number of super-nodes and the communication "overhead coming from ParMETIS on the performance. We obtained similar results for the other matrices in Table 1. SuperLU_MCDT uses dense block structures, called super-nodes to get advantages of BLAS3 (see [19]) with the common technique of array padding, like SuperLU_DIST 3.3. Super-node detection differs as process mesh size and its square or rectangular shape. So we observe sometimes more efficient case matched to the super-node detection strategies of the algorithm where the optimal minimum number of cores for the matrix mC_8M is 512.

Conclusions

The existing versions of SuperLU are sensitive to challenging matrices and need exception handling. Apart from the solver, spectral analysis can be done and tuned parameters may be used accordingly. We released the first SuperLU MCDT (Many Core Distributed) version (1.0) with several novelties based on the direct solver SuperLU DIST 3.3. Our benchmark tests show that SuperLU MCDT can run on up to 16348 cores.

There is no unique solver that fits all our needs for every matrix because of the rich pattern spectrum of matrices and the NP-complete problem of best reordering for minimum fill-in. We observe that the optimal minimum number of cores can be different depending on the matrix properties. The existence of optimal minimum number of cores requires a rule base to make a decision.

We believe that expert systems (see [22]), knowledge-based computer programs with a set of inference rules ('if then' type statements) in a rule base, are among the most promising subfields in artificial intelligence for big data discovery and decision making applications such as oil and gas reservoir simulators in a timely and reliable fashion. We plan that expert system tools for real time decision making based on the spectral properties and the super-node detection strategies of various large patterned matrices coming from reservoir modeling and the exception handling for the challenging matrices will be among the new properties of SuperLU_MCDT version (2.0). We will use an expert system with forward chaining as a reasoning method to reach conclusions in our learning algorithm.

Acknowledgements

The authors thank Saudi Aramco for sponsoring a part of the research and Saudi Aramco visit of the authors. The authors are grateful to Dr. Ali Dogru for project coordination, valuable comments and suggestions. The authors thank Larry Fung for helpful communications, constructive feedback and providing the reservoir simulation test matrices. Computing resources used in this work were provided mostly by Istanbul Technical University (ITU) - the National Center for High Performance Computing of Turkey (UHeM) (see [21]). Also, the work was achieved partially using the PRACE Research Infrastructure resource at CEA, France (see [25]). The authors are thankful to ITU-UHeM and CEA.

References

1. X. S. Li, J. W. Demmel, J. R. Gilbert, L. Grigori, M. Shao, and I. Yamazaki, *SuperLU Users' Guide,* Tech. Report UCB, Computer Science Division, University of California, Berkeley, CA, 1999, update: 2011.
2. X. S. Li and J. W. Demmel, Superlu-dist: A scalable distributed-memory sparse direct solver for unsymmetric linear systems, *ACM Trans. Math. Softw.,* **29** (2003), pp. 110–140.
3. P. R. Amestoy, I. S. Duff, J.-Y. L'Excellent, and J. Koster, A fully asynchronous multifrontal solver using distributed dynamic scheduling, *SIAM J. Matrix Anal. Appl.,* **23** (2001), pp. 15–41.
4. O. Schenk and K. Gartner, Solving unsymmetric sparse systems of linear equations with PARDISO, *Future Generation Computer Systems,* **20** (2004), pp. 475–487.
5. O. Schenk and K. Gartner, On fast factorization pivoting methods for sparse symmetric indefinite systems, *Electronic Transactions on Numerical Analysis,* **23** (2006), pp. 158–179.
6. A. Duran and B.D. Saunders, *Gen_SuperLU package (version 1.0, August 2002), referenced as GSLU also, a part of LinBox package*. GSLU contains a set of subroutines to solve a sparse linear system A*X=B over any field. http://web.itu.edu.tr/~aduran/Gen_SuperLU.pdf
7. A. Duran, B. D. Saunders and Z. Wan, Hybrid algorithms for rank of sparse matrices, Proceedings of the SIAM International Conference on Applied Linear Algebra (SIAM-LA), Williamsburg, VA, July 15-19, 2003.
8. A. Duran, M.S. Celebi, M. Tuncel and B. Akaydın, Design and implementation of new hybrid algorithm and solver on CPU for large sparse linear systems, PRACE-2IP white paper, Libraries, WP 43, July 13, 2012, http://www.prace-ri.eu/IMG/pdf/wp43-newhybridalgorithmfo_lsls.pdf
9. V. A. Marchenko and L. A. Pastur. Distribution of eigenvalues for some sets of random matrices. *Math. USSR-Sb* **83**(7) (1967) pp 457–486.
10. Al-Shaalan, T. M., Fung, L. S. K., Dogru, A.H., "A Scalable Massively Parallel Dual-Porosity Dual-Permeability Simulator for Fractured Reservoirs with Super-K Permeability," paper SPE 84371, SPE Annual Technical Conference and Exhibition, Denver, Colorado, 5-8, Oct. 2003.

11. (Par)METIS homesite: http://www.lrz.de/services/software/mathematik/metis
12. G. Strang. *Linear Algebra and Its Applications* (3rd ed.) (1988). San Diego: Harcourt.
13. http://www.grycap.upv.es/slepc
14. http://www.mcs.anl.gov/petsc
15. nPartition Administrator's Guide, *HP part number: 5991-1247B,* 1st Edition, February 2007, Hewlett-Packard Development Company.
16. A. Duran, M.S. Celebi, M. Tuncel, and B. Akaydin, Scalability of SuperLU solvers for large scale complex reservoir simulations, SPE and SIAM Conference on Mathematical Methods in Fluid Dynamics and Simulation of Giant Oil and Gas Reservoirs, Istanbul, Turkey, September 3-5, 2012. SPE Conference, 2012
17. T.A. Davis, *University of Florida sparse matrix collection*. http://www.cise.ufl.edu/research/sparse/matrices/.
18. http://www.uybhm.itu.edu.tr/eng/inner/duyurular.html#karadeniz
19. http://www.netlib.org/blas
20. http://software.intel.com/en-us/intel-mkl
21. http://www.uybhm.itu.edu.tr
22. Feigenbaum, E.A., *Expert systems: principles and practice*. KSL-91-79, 1992
23. A. Duran, M.S. Celebi, S. Piskin, and Mehmet Tuncel, Scalability of OpenFOAM for Bio-medical Flow Simulations, Partnership for Advanced Computing in Europe (PRACE) white paper, Exploitation of HPC Tools and Techniques, WP 162, June 9, 2014
24. A. Duran, M.S. Celebi, M. Tuncel and F. Oztoprak, Structural Analysis of Large Sparse Matrices for Scalable Direct Solvers, PRACE-2IP white paper, Scalable Algorithms, WP 82, August 20, 2013
25. http://www-hpc.cea.fr/en/complexe/tgcc-curie.htm

Document details

‹ Back to results | ‹ Previous 8 of 19 Next ›

Export Download Print E-mail Save to PDF Add to List More... ›

Locate LinkSource

SPE Large Scale Computing and Big Data Challenges in Reservoir Simulation Conference and Exhibition 2014
2015, Pages 140-149
SPE Large Scale Computing and Big Data Challenges in Reservoir Simulation Conference and Exhibition 2014; Istanbul; Turkey; 15 September 2014 through 17 September 2014; Code 116926

Spectral effects of large matrices from oil reservoir simulators on performance of scalable direct solvers (Conference Paper)

Duran, A. , Tuncel, M.

Department of Mathematics, National Center for High Performance Computing of Turkey (UHeM), Istanbul Technical University, Istanbul, 34469, Turkey

Abstract

View references (25)

It is important to estimate the elapsed time to solve large sparse linear systems for time-restricted real life decision making applications such as oil and gas reservoir simulators. Challenging matrices should be distinguished and handled separately because they may lead to performance bottleneck. Therefore, we need to examine the spectral effects of large matrices on the performance of scalable direct solvers by using eigenvalues. In this work, we check whether there is relationship between the eigenvalue distribution of a matrix and the performance of the solver. We try to examine the eigenvalue distribution of various sparse matrices. We may find all eigenvalues in order to obtain the distribution graph of eigenvalues, if possible. However, it is very expensive to find all eigenvalues. Therefore, Gerschgorin's theorem may be used to bound the spectrum of square matrices. Several behaviors such as being disjoint, overlapped or clustered of Gerschgorin circles may give clue regarding the distribution of the eigenvalues and the performance of the solver for that matrix. In this paper, we consider a portfolio of test

matrices which include randomly populated sparse matrices and various patterned matrices coming from reservoir modeling from single porosity single permeability to dual porosity dual permeability models (see [10]). We examined our modified HELM2D03LOWER-20K matrix and EMILIA-923 matrix from the University of Florida sparse matrix collection (see [17]), in addition to the patterned matrices from 3 phase black-oil model and 7 component EOS model. We define an optimal minimum number of cores as the number of cores that provides the minimum wall clock time for a given size of problem, where a right match occurs between the problem size, the spectral effects of matrix and the available resources such as memory, in presence of communication overhead. We find that the optimal minimum number of cores required depends on the sparsity level and size of the matrix. As the sparsity level of matrix decreases and the order of matrix increases, we expect that the optimal minimum number of cores increases slightly.

SciVal Topic Prominence

Topic: Multifrontal Method | LU Factorization | Sparse Matrix

Prominence percentile: 73.945

Indexed keywords

Engineering controlled terms:

Big data | Computation theory | Decision making | Eigenvalues and eigenfunctions | Linear systems | Low permeability reservoirs | Mechanical permeability | Petroleum reservoir engineering | Petroleum reservoir evaluation | Petroleum reservoirs | Porosity | Real time systems

Engineering uncontrolled terms

Communication overheads | Dual-porosity dual-permeability models | Eigenvalue distributions | Gerschgorin circles | Large sparse linear systems | Oil and gas reservoir | Performance bottlenecks | University of Florida

Engineering main heading:

Matrix algebra

Decomposition and parallelization of strongly coupled fluid-structure interaction linear subsystems based on the Q1/P0 discretization

Toma, M. , Oshima, M. , Takagi, S.
(2016) Computers and Structures

A novel TDFEM scheme based on high-performance parallel solver PaStiX

Wu, X.
(2020) International Journal of Numerical Modelling: Electronic Networks, Devices and Fields

View all related documents based on references

Find more related documents in Scopus based on:

Authors > Keywords >