Available online at www.prace-ri.eu

Partnership for Advanced Computing in Europe

Structural Analysis of Large Sparse Matrices for Scalable Direct Solvers

Ahmet Duran[a,b]*, M. Serdar Celebi[a,c], Mehmet Tuncel[a,c], Figen Oztoprak[a,c]

[a]*Istanbul Technical University, National Center for High Performance Computing of Turkey (UHeM), Istanbul 34469, Turkey*
[b]*Istanbul Technical University,Department of Mathematics, Istanbul 34469, Turkey*
[c]*Istanbul Technical University, Informatics Institute, Istanbul 34469, Turkey*

July 8, 2013

Abstract

It is significant to perform structural analysis of large sparse matrices in order to obtain scalable direct solvers. In this paper, we focus on spectral analysis of large sparse matrices. We believe that the approach for exception handling of challenging matrices via Gerschgorin circles and using tuned parameters is beneficial and practical to stabilize the performance of sparse direct solvers. Nearly defective matrices are among challenging matrices for the performance of solver. We observe that the usage of super-nodal storage parameters affects the number of fill-ins and memory usage accordingly.

1. Introduction

We design and implement a new hybrid algorithm and solver for large sparse linear systems. We consider scalable direct solvers because of their robustness and examine the SuperLU_DIST 3.3 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] and Celebi et al. [9] discussed the advantages and limitations of the SuperLU solvers and tested the code of SuperLU_DIST 3.0 in order to measure the performance scalability for various patterned sparse matrices and randomly populated sparse matrices (see [10] for the theoretical foundation regarding the distribution of eigenvalues for some sets of random matrices). Although the existing versions of SuperLU work well for many matrices, they need to be improved for certain types of sparse matrices.

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 and financial applications (see [11]). Challenging matrices should be distinguished and handled separately because they may lead to performance bottleneck. Therefore, structural analysis of large sparse matrices for scalable direct solvers are needed. In this work, we focus on spectral analysis of large sparse matrices and check whether there is relationship between the eigenvalue distribution of 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

* Corresponding author. *E-mail address*: aduran@itu.edu.tr.

Gerschgorin circles may give clue regarding the distribution of the eigenvalues and the performance of the solver for that matrix.

The presence of repeated eigenvalues can be one of the source 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 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. The exception handling is one of the new properties of SuperLU_MCDT (Multi Core Distributed) solver (see Duran et al. [8] and Celebi et al. [9]).

The remainder of this work is organized as follows. In Section 2, the test matrices including randomly populated matrices and patterned matrices are described. Later, the computation for spectral properties is presented and several illustrative examples are given. Section 3 concludes this work.

2. Methods and results

We consider a portfolio of test matrices containing randomly populated sparse matrices in addition to patterned matrices. We generate 30 different randomly populated matrices RAND_30K_3, ..., RAND_30K_100 for each. We describe the matrices in Table 1 and Table 2, respectively.

2.1. Description of matrices

Table 1. Description of patterned matrices

Patterned matrices Name	Order	NNZ	NNZ/N	Nonzero pattern symmetry	Numeric value symmetry	Origin	Kind of problem
EMILIA_923	923136	40373538	43,74	100%	100%	UFSMC	Geomechanical structural
HELM2D03LOWER_20K	392257	1939353	4,94	0%	0%	UHeM	
M_UHEM3	1425825	17037638	11,94	77%	54%	UHeM	

Table 2. Description of randomly populated matrices

Randomly populated matrices Name	Order	Number of nonzeros (NNZ)	NNZ per row (NNZ/N)	Condition number	Origin
RAND_30K_3	30000	90000	3	1,20E+006	UHeM
RAND_30K_5	30000	150000	5	4,22E+006	UHeM
RAND_30K_7	30000	210000	7	1,76E+006	UHeM
RAND_30K_9	30000	270000	9	2,51E+006	UHeM
RAND_30K_11	30000	330000	11	8,82E+005	UHeM

RAND_30K_30	30000	900000	30	1,13E+006	UHeM
RAND_30K_50	30000	1500000	50	7,03E+005	UHeM
RAND_30K_75	30000	2250000	75	1,16E+006	UHeM
RAND_30K_100	30000	3000000	100	3,39E+006	UHeM
RAND_10K_3	10000	30000	3	7,10E+005	UHeM
RAND_20K_3	20000	60000	3	3,19E+005	UHeM
RAND_30K_3	30000	90000	3	1,20E+006	UHeM
RAND_40K_3	40000	120000	3	3,90E+006	UHeM
RAND_50K_3	50000	150000	3	1,20E+006	UHeM
RAND_60K_3	60000	180000	3	2,14E+006	UHeM

2.2. Computation for spectral properties

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. 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. The termination criterion is set as the norm of the residual be under a specified level, i.e. $||Av-\lambda v||<\varepsilon$, and ε is chosen as the smallest value that allows termination in a reasonable time.

Gerschgorin's Theorem provides a range for each eigenvalue of a matrix. These ranges can be computed very easily. We observed that for some sparse matrices in our test the ranges suggested by the theorem provide meaningful information. The theorem can formally be stated as follows. Let A be an n x n matrix, and Λ be the set of the eigenvalues of A. Then, $\Lambda \subseteq \bigcup_{i=1}^{n} R_i$ where $R_i = \{\lambda: |\lambda - A_{ii}| \leq \sum_{j=1, j\neq i}^{n} |A_{ij}|\}$.

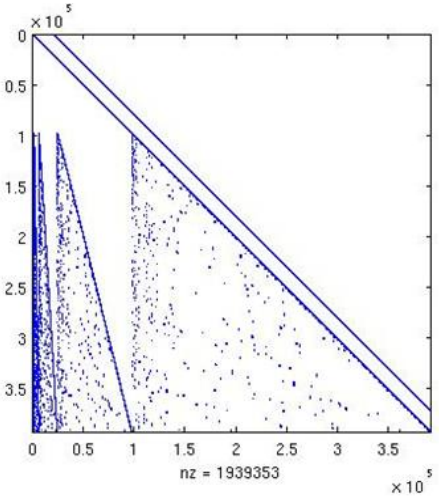

Fig. 1. Matrix picture of HELM2D03LOWER_20K

The computation of all eigenvalues may not be feasible for large sparse matrices, mainly due to memory constraints. Therefore, we followed two strategies to get an idea about the eigenvalue distribution of the test matrices:

1. 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. For example, we show the Gerschgorin's circles of matrix Emilia_923, matrix HELM2D03LOWER_20K, and the patched matrix M_UHEM3 (see Duran et al. [16]) in Figure 3, 2, and 4, respectively.
2. We can compute all eigenvalues of the small randomly populated matrices and show the distribution of eigenvalues for RAND_30K_100 in Figure 5. We observe that nearly all eigenvalues can be found within the circle except for the largest eigenvalue that is indicated by cross in figure.

Although the existing versions of SuperLU work well for many reasonable matrices, they need to be improved for certain types of sparse matrices. For example, we generated a new unsymmetric matrix HELM2D03LOWER_20K (see Duran et al. [8]), shown in Figure 1, which consists of the lower triangular part of a symmetric matrix HELM2D03 from the University of Florida sparse matrix collection [17] and an upper subdiagonal with 20000 distance from the main diagonal. We reported in our PRACE WP43 paper (see Duran et al. [8]) that SuperLU_DIST 3.0 failed for HELM2D03LOWER_20K due to symbolic factorization error, although it works well for the matrix HELM2D03 on the Linux Nehalem Cluster (see [18]) available at UHeM. Later, the bug in the factorization routine was fixed in April 2013.

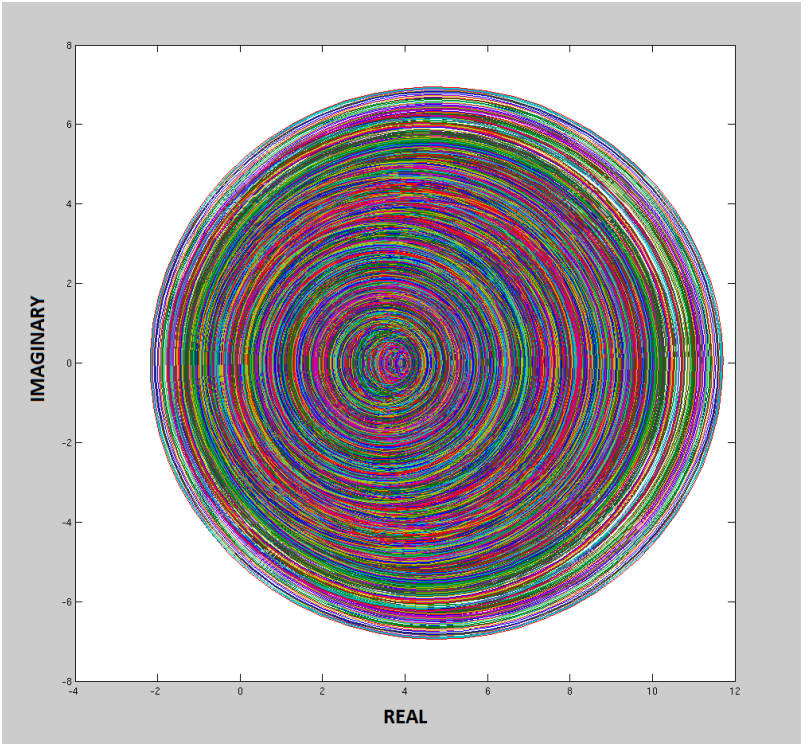

Fig. 2. Gerschgorin's circles of matrix HELM2D03LOWER_20K

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 (see Table 3), 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. Table 3 shows the performance of the SuperLU_DIST 3.3 for HELM2D03LOWER_20K and EMILIA_923 by using standard BLAS [19] and Intel's Math Kernel Library (MKL) [20] which is a kind of optimized BLAS on the Linux Nehalem Cluster with 64 (8x8 mesh) cores.

The tunings of super-nodal storage parameters are important. For example, the usage of tuned parameters (relax:100 and maxsuper:110) outperforms at least 1.8 times faster than that of default parameters (relax:12 and maxsuper:60) for HELM2D03LOWER_20K using the SuperLU_DIST 3.3. Moreover, the usage of super-nodal storage parameters affects the number of fill-ins. For instance, there are 3 208 629 380 non-zeros in L+U with the default parameters compared to 3 477 287 771 non-zeros of L+U in presence of the tuned parameters.

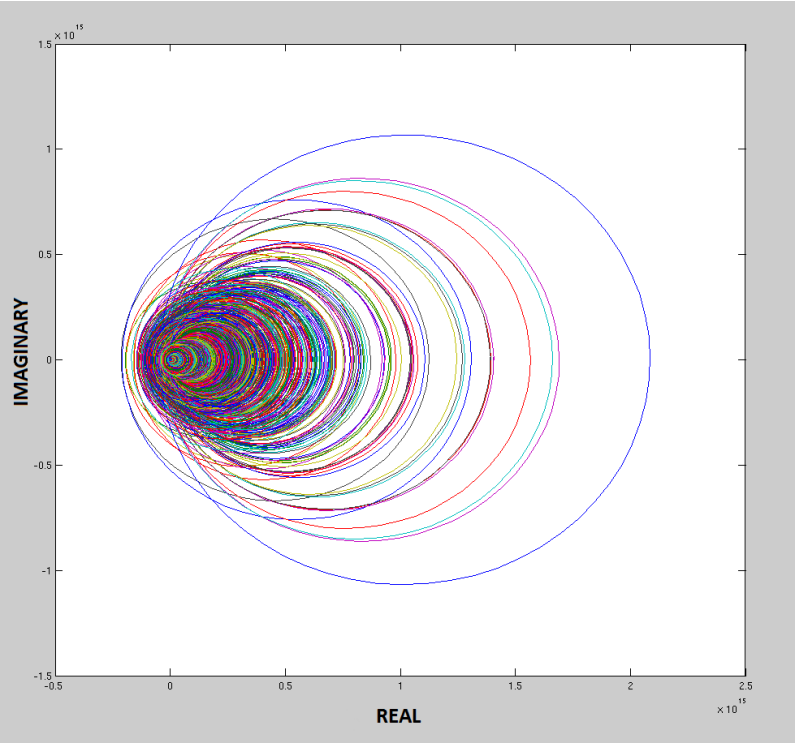

Fig. 3. Gerschgorin's circles of matrix Emilia_923

Table 3. The performance of the SuperLU_DIST 3.3 for HELM2D03LOWER_20K and EMILIA_923.

Wall clock time (s)	BLAS		MKL	
Patterned matrices	Default Parameters	Tuned Parameters	Default Parameters	Tuned Parameters
HELM2D03LOWER_20K	5594,72	3047,56	5310,04	2324,00
EMILIA_923	743,29			

When we examine the spectral properties of HELM2D03LOWER_20K in Figure 2, 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 as in Figure 2. Therefore, HELM2D03LOWER_20K is a nearly defective matrix.

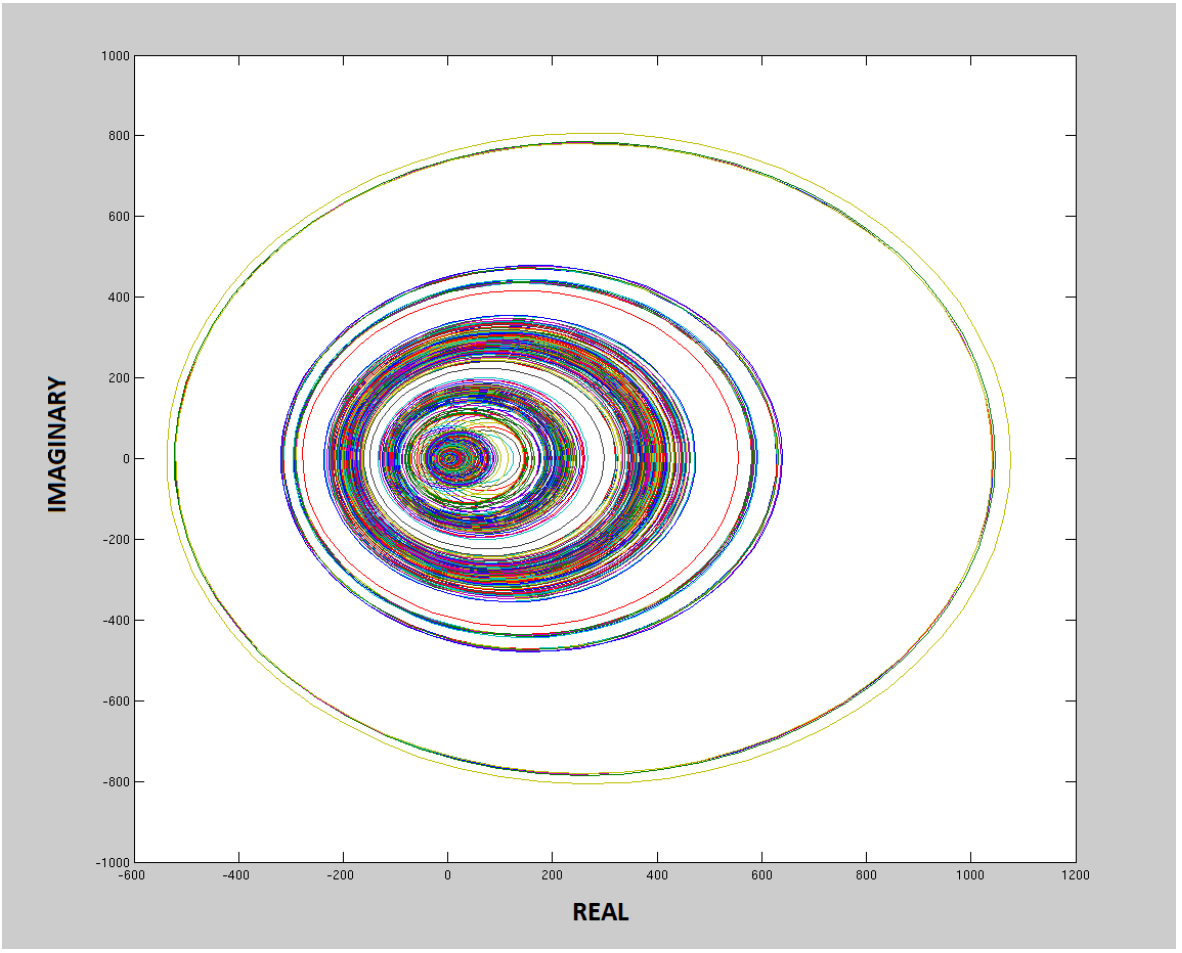

Fig. 4. Gerschgorin's circles of matrix M_UHEM3.

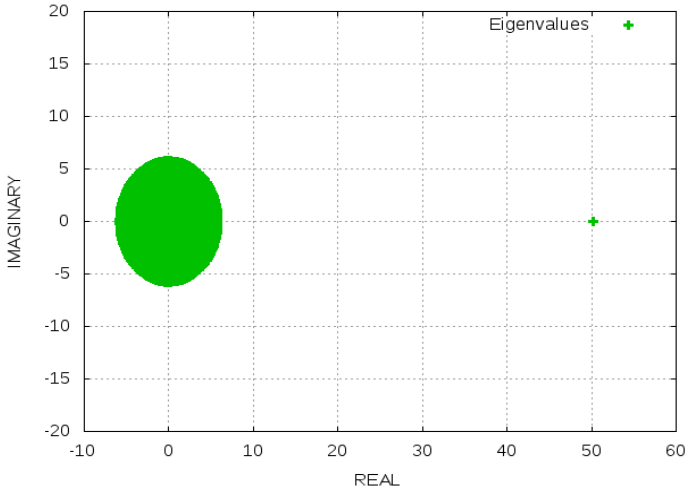

Fig. 5. Distribution of eigenvalues for matrix RAND_30K_100

3. Conclusions

We believe that the approach of exception handling of challenging matrices via Gerschgorin circles and using tuned parameters is beneficial and practical to stabilize the performance of sparse direct solvers.

Nearly defective matrices are among challenging matrices. Such matrices should be handled separately in order to get rid of potential performance bottleneck. Clustered eigenvalues observed via Gerschgorin circles may be used to detect nearly defective matrix.

We reported in our PRACE WP43 paper (see Duran et al. [8]) that SuperLU_DIST 3.0 failed for HELM2D03LOWER_20K due to symbolic factorization error. Later, the bug in the factorization routine was fixed in April 2013. We noticed that the SuperLU_DIST 3.3 with tunings of super-nodal storage parameters works for HELM2D03LOWER_20K but slowly.

The tunings of super-nodal storage parameters are important. For example, the usage of tuned parameters outperforms at least 1.8 times faster than that of default parameters for HELM2D03LOWER_20K using the SuperLU_DIST 3.3. Moreover, we observe that the usage of super-nodal storage parameters affects the number of fill-ins and memory usage.

Acknowledgements

This work was financially supported by the PRACE project funded in part by the EUs 7th Framework Programme (FP7/2011-2013) under grant agreement no. RI-283493. Computing resources used in this work were provided by the National Center for High Performance Computing of Turkey (UHeM) (see [21]) under grant number 1001682012.

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.

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. M.S. Celebi, A. Duran, M. Tuncel and B. Akaydın, Scalable and improved SuperLU on GPU for heterogeneous systems, PRACE-2IP white paper, Libraries, WP 44, July 13, 2012, http://www.prace-ri.eu/IMG/pdf/scalablesuperluongpu.pdf

10.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.

11. A. Duran and M.J. Bommarito, A profitable trading and risk management strategy despite transaction cost, Quantitative Finance, 11(6), 2011, pp. 829-848.

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

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

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

19. http://www.netlib.org/blas

20. http://software.intel.com/en-us/intel-mkl

21. http://www.uybhm.itu.edu.tr