ISSN 1392 – 124X INFORMACINĖS TECHNOLOGIJOS IR VALDYMAS, 2004, Nr. 1(30)

PARALLEL GMRES SOLUTION OF CONVECTIVE TRANSPORT PROBLEMS ON DISTRIBUTED MEMORY COMPUTERS

Arnas Kačeniauskas

Parallel Computing Laboratory, Vilnius Gediminas Technical University
Saulėtekio al. 11, LT-2040, Vilnius, Lithuania

Abstract. Parallel GMRES algorithms for solution of convective transport problems on distributed memory computers are investigated in this paper. The space-time FEM stabilised by the GLS formulation is applied as the most accurate and stable. The parallel algorithms based on the domain decomposition and iterative GMRES solver are implemented in order to reduce long computing time associated with space-time GLS method. The developed data structures, the static load balancing and the inter-processor communication algorithms are particularly suited for homogeneous distributed memory PC clusters. The benchmark problem of the rotating cone is solved in order to validate efficiency of the developed software. The performance of parallel computations, the speed-up and the efficiency is measured on several PC clusters as well as on the IBM SP2 supercomputer.

1. Indroduction

Many important problems involve convective transport phenomena. Various computational methods are developed for discretising the governing equations and for efficient solution of convective transport problems. The software based on the finite element method (FEM) [20] has the flexible and universal structure allowing to solve complex problems in an integrated fashion and making it particularly interesting to the designers of advanced technologies as well as to the software developers. However, it has always been the problem of the FEM codes that larger computational times have been associated with them. Parallel computing is thus perceived as a promising avenue for future advances in this applied area of science.

Domain decomposition [2] is the most efficient parallelisation technique used for finite element algorithms on distributed memory computers. The basic idea of this technique is the partitioning the computational domain into sub-domains, each being assigned to a processor. A non-overlapping domain decomposition [18] of the grid is generated that adjacent processors share nodes on the sub-domain interfaces. The sub-domains exchange data with each other through their boundaries. Inter-processor communication is necessary only when the solution on boundary elements of a sub-domain is sought. Such computation and communication arrangement enables data parallelism to be executed efficiently and is particularly suitable for platforms of distributed memory computers.

The most time consuming parts of FEM software are assembling of finite element coefficient matrices and solving large systems of linear equations. The domain decomposition strategy significantly reduces the first time consuming part, because coefficient matrices are assembled locally on each processor. Iterative Krylov subspace methods are believed to be very fast in solving large linear equation systems. The GMRES [15] is popular method for parallel iterative solution of such systems. The parallel implementation of restarted GMRES(m) algorithm and the reduction of global communication overhead on distributed memory computers were studied by De Sturler and Van der Vorst [17]. Various preconditioners [10, 13, 16, 19] are used for improving the convergence of iterative Krylov solvers. Despite this there is little theory available to guide the design of efficient and parallel preconditioners for the various types of matrices. The diagonal [3] and block-diagonal [9] preconditioners based on the simple iteration of Jacobi type are believed to be the simplest and very well parallelised preconditioners.

The main purpose of this work is construction of parallel algorithms for efficient solution of convective transport problems on distributed memory computers. The paper is organised as follows. Section 2 outlines the mathematical model of convective transport phenomena. Section 3 presents the parallel algorithms for distributed memory PC clusters. In section 4, the results of parallel efficiency analysis are discussed. Conclusions are drawn in Section 5.

2. Mathematical model

Convective transport problems are governed by the hyperbolic partial differential equation:

$$\frac{\partial \psi}{\partial t} + u_j \frac{\partial \psi}{\partial x_j} = 0 , \qquad (1)$$

here, ψ is transported variable and u_j is the j-th component of the velocity vector. Equation (1) together with initial conditions and prescribed ψ values on inflow boundaries forms the mathematical model of investigated problems.

The standard Galerkin method yields oscillatory solutions when it is applied to convection dominant problems in conjunction with classical time-stepping algorithms. The Galerkin Least Squares Method (GLS) [4] belongs to the family of the stabilised methods based on adding a stabilisation term to the Galerkin method. This stabilisation term is the least square form of the residual of the equation (1) evaluated elementwise and multiplied by a stabilisation parameter. In this work GLS method is used together with space-time approach [5, 12], and the temporal derivatives are computed using central weighted space-time finite elements in every space-time slab. The employed implicit numerical schema [7] is unconditionally stable. The resulting coefficient matrices are unsymmetrical in spite of the symmetry of the stabilising term.

3. Parallel GMRES solver

Parallel algorithms were implemented in the FEMTOOL program [14] created in the Swiss Federal Institute of Technology and developed in Vilnius Gediminas Technical University. FEMTOOL is a Finite Element Method Toolbox, which allows implementation of any partial differential equation with minor expenses. Space-time finite element integration in time and the high order shape functions generated automatically make FEMTOOL applicable to efficient solution of convective transport problems. Parallelisation of the FEMTOOL is based on the domain decomposition, which is the most efficient strategy used on distributed memory computers. Finite element meshes are partitioned into sub-domains by ParMETIS libraries [6]. The multilevel graph partitioning schemes and parallel multilevel k-way graph partitioning algorithms are employed for creating finite element mesh partitions of high quality. Partitions of roughly equal size ensure the static load balancing [11] on the homogeneous parallel machines.

Parallel sparse matrix-vector operations need information about the degrees of freedom processed on neighbouring sub-domains. Every processor creates local arrays for communication between neighbouring processors. Data structures shown on Figure 1 illustrate how the processor 1 addresses and stores the local data necessary for neighbouring processors 0, 2 and 3. Local variable *nr_of_nb* stores the number of neighbouring processors (sub-domains) for a considered processor. The array ***nb2proc***(*nr_of_nb*) contains identification numbers of neighbouring processors. The variable *tot_bnd_var* stores a local number of degrees of freedom that are also processed on the other processors. Only these unknowns are needed for inter-processor communication in parallel FEM codes. The array ***nb2var_ptr***(*nr_of_nb+1*) contains the pointers to the array ***nb2var***(*tot_bnd_var*). These pointers are used for convenient access of data belonging to any processor. The last cell in array ***nb2var_ptr*** is used for identifying the end of the array ***nb2var***. The main integer array ***nb2var***(*tot_bnd_var*) defines the position of a considered variable in assembled coefficient matrix or vector of unknowns ***y***(*neqtot*). The variable *neqtot* stores the local number of degrees of freedom. The necessary data is addressed in the local vector ***y*** and is sent to the neighbouring processors by the MPI subroutines. Parallel matrix-vector operations based on the MPI inter-processor communication forms the basis of the GMRES solver.

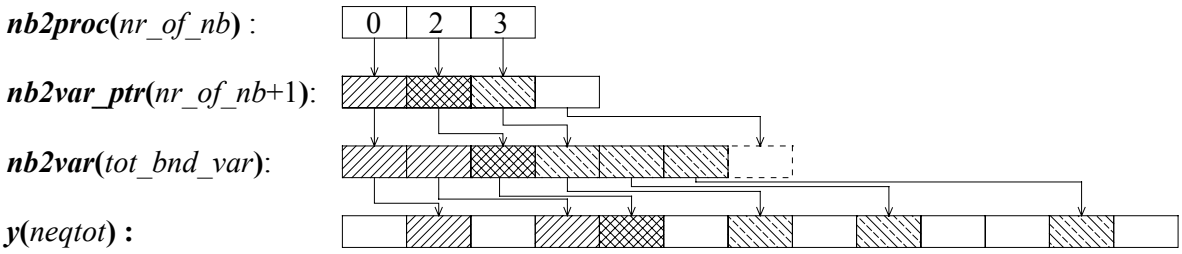

Figure 1. Data structures for inter-processor communication

The GMRES [15] is a popular method for iterative solution of linear equation systems with non-symmetric coefficient matrices. The GMRES solver is based on the matrix and vector operations, therefore, it can be parallelised using the algorithm described above. The following GMRES algorithm (Figure 2) was implemented in the FEMTOOL software. The iterative solver is based on the orthogonal Krylov subspaces $\{\boldsymbol{r}, \boldsymbol{Ar}, \boldsymbol{A}^2\boldsymbol{r}, \ldots, \boldsymbol{A}^{m-1}\boldsymbol{r}\}$ that can be changed to preconditioned sub-spaces $\{\boldsymbol{r}, \boldsymbol{K}^{-1}\boldsymbol{Ar}, (\boldsymbol{K}^{-1}\boldsymbol{A})^2\boldsymbol{r}, \ldots, (\boldsymbol{K}^{-1}\boldsymbol{A})^{m-1}\boldsymbol{r}\}$. The outer loop i starts from the computing residual vector $\boldsymbol{r}$ and the first vector of the Krylov subspace $\boldsymbol{v}_1$. The inner loop j constructs the orthogonal Krylov sub-space $\{\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_m\}$. The orthogonalisation is performed by a modified Gram-Smidt process in the third loop k. Hessenberg matrix $\boldsymbol{H}_{i,j}$ is

also constructed in the second loop j. The projected system matrix $\boldsymbol{H}_{i,j}$ is transformed by Givens rotations $\boldsymbol{J}$ to an upper triangular matrix with the last row equal to zero. The $\boldsymbol{s}_{j+1}$ estimates the norm of residuals $\left\|\mathbf{b}-\mathbf{A}\tilde{\mathbf{x}}\right\|_2$, therefore, computations exit the inner loop if $\boldsymbol{s}_{j+1}$ is smaller than the prescribed accuracy ε. When the inner loop is completed, the solution of the projected system $\boldsymbol{Hy}=\boldsymbol{s}$ is obtained and the solution $\boldsymbol{x}$ of the global system of equations is updated. If the norm of residuals is not small enough, computations are restarted and the next iteration of the outer loop is performed. In the outlined algorithm, the outer iterations are limited by a large number n for practical reasons.

do $i = 1, 2, \dots, n$
  $\boldsymbol{Kr} = \boldsymbol{b} - \boldsymbol{Ax}$
  $\mathbf{v}_1 = \mathbf{r} / \left\|\mathbf{r}\right\|_2$
  $\mathbf{s}_1 = \mathbf{e}_1 \left\|\mathbf{r}\right\|_2$
  do $j = 1, 2, \dots, m$
    $\boldsymbol{Kv}_{j+1} = \boldsymbol{Av}_j$
    do $k = 1, 2, \dots, j$
      $\boldsymbol{H}_{k,j} = (\boldsymbol{v}_{j+1}, \boldsymbol{v}_k)$
      $\boldsymbol{v}_{j+1} = \boldsymbol{v}_{j+1} - \boldsymbol{H}_{k,j}\, \boldsymbol{v}_i$
    endo
    $\mathbf{H}_{j+1,j} = \left\|\mathbf{v}_{j+1}\right\|_2$
    $\boldsymbol{v}_{j+1} = \boldsymbol{v}_{j+1} / \boldsymbol{H}_{j+1,j}$
    Construction of rotation matrix $\boldsymbol{J}$
    Transformation of $\boldsymbol{H}$ *and* $\boldsymbol{s}$ *by* $\boldsymbol{J}$
    if ($\boldsymbol{s}_{j+1} < \varepsilon$) *exit from loop*
  enddo
  Solution of $\boldsymbol{Hy} = \boldsymbol{s}$
  $\boldsymbol{x} = \boldsymbol{x} + \boldsymbol{v}_1\boldsymbol{y}_1 + \dots + \boldsymbol{v}_m\boldsymbol{y}_m$
  if ($\boldsymbol{s}_{j+1} < \varepsilon$) *exit from loop*
enddo

Figure 2. GMRES algorithm

The iteration number of the GMRES solver depends on a preconditioner $\boldsymbol{K}$. The simple diagonal preconditioner [3] is implemented in the software. The lack of robustness and universality of the diagonal preconditioner is compensated by the trivial parallel implementation and the negligible amount of interprocessor communication. The need for better parallel preconditioner which is robust and dramatically reduce the number of iterations while not significantly degrading scalability and parallel performance is a key open problem for the most PDEs applications. The most appropriate preconditioning strategies based on the approximate Shur complement [1], multilevel approaches [10] or overlapped Schwarcz splitting [13] requires significant changes in the data structures used in the standard FEM software. The local ILUT preconditioning [16, 19] combined with non-overlapped domain decomposition is implemented and tested in this work. The sequential version of the ILUT preconditioner is very efficient, but the parallel implementation on the distributed memory computers is very complex. Slightly overlapping sub-domains combined with local preconditioning produce large increase of the iteration number and huge inter-processor communication overhead. This unsuccessful attempt only confirms the fact, that design of universal and efficient parallel preconditioner for ill-conditioned matrices reveals a great challenge.

4. Parallel tests and efficiency analysis

The parallel FEMTOOL software presented in this paper is tested on several distributed memory architectures. The main tests are performed on BEOWULF clusters: PC cluster VILKAS (OS Linux, 20 Intel Pentium III processors, 1.4GHz, 0.5GB RAM for a processor, 1Gbit/s network) and Transtec PC cluster (OS Linux, 32 Intel Xeon processors, 2.2GHz, 0.5GB RAM for a processor, 1Gbit/s network). Nodes are interconnected with a hub running 1 Gbit Ethernet. The performance achieved on the PC clusters is compared with that of obtained on IBM SP2 supercomputer (OS AIX, 4 processors, 120MHz, 128MB RAM for a processor, 360Mbit/s network). The parallel performance of the developed code is judged by measurements of speed-up S_p and the efficiency E_p:

$$S_p = \frac{t_1}{t_p},\ E_p = \frac{S_p}{p}, \qquad (2)$$

here, t_1 is the program execution time for a single processor; t_p is the wall clock time for a given job to execute on p processors. Parallel efficiency is measured by fixing the problem size and increasing the number of processors used. In practice, perfect efficiency is not naturally attained because of an inherent serial part of the algorithm, parallel communication overhead and load imbalance.

The rotating cone problem is chosen as the benchmark problem widely used to illustrate the effectiveness of the algorithms in case of convection dominated flows. The 2D square solution domain [-0.5; 0.5]x[-0.5; 0.5] is discretised by different structural finite element meshes of 10000, 40000, 90000 and 160000 finite elements (10201, 40401, 90601 and 160801 degrees of freedom, respectively).

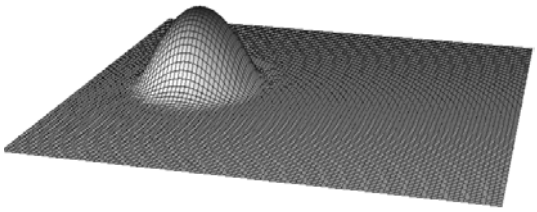

Figure 3. The rotating cone after 2π period of time and finite element mesh

The concentration cone of radius 0.15 is positioned at (-0.25; 0.0). The concentration maximum equals 1.0 in the centre of the cone and decreases to zero as a sinusoidal curve. The velocity field $u=-y$, $v=x$ corresponds to a rotational flow. The problem is numerically difficult to solve not only because of the pure advection but also because of the numerical

diffusion attributed to the Cartesian grid discretising the rotational flow field.

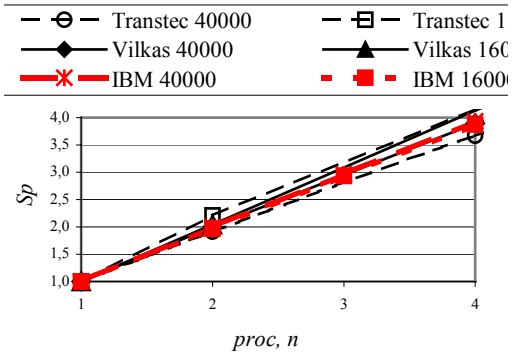

Figure 4. The comparison of the speed-up obtained on the PC cluster VILKAS, on the Transtec PC cluster and on the IBM SP2 supercomputer

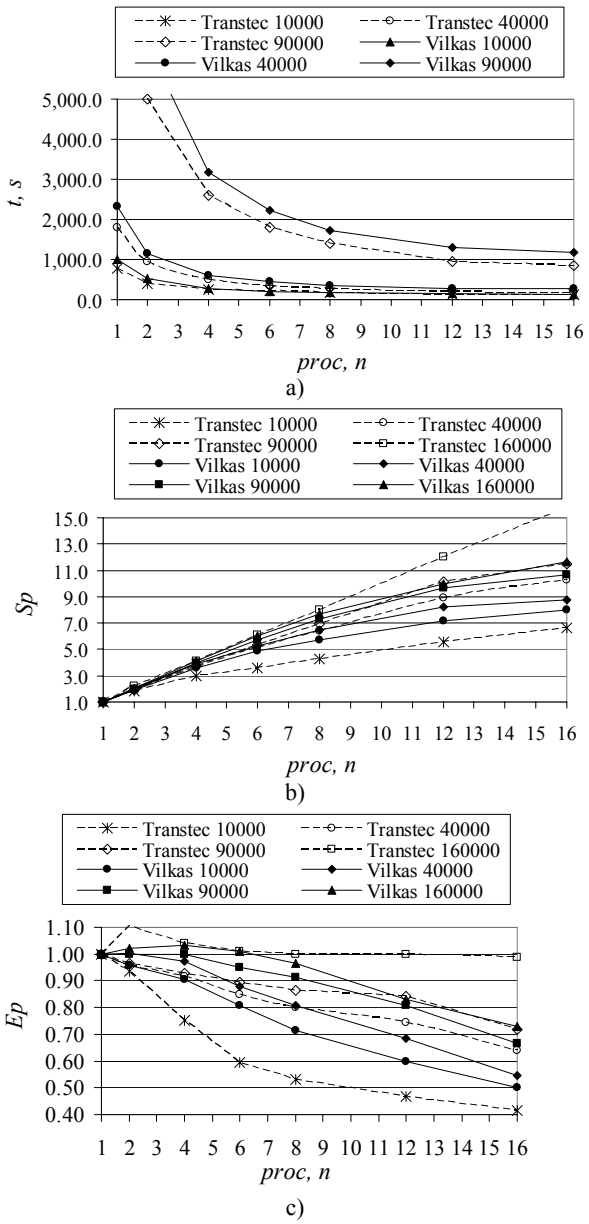

Figure 5. Performance tests on the Transtec PC cluster and on the PC cluster VILKAS:
(a) the run time, (b) the speed-up, (c) the efficiency

The accuracy and the efficiency of the developed implementation of the space-time GLS method significantly outperform those of other investigated methods [7]. A position of the rotating cone after 2π period of time is shown in Figure 3.

The results of parallel efficiency analysis are illustrated in Figures 4-5. The speed-up obtained on the PC clusters is compared with that of measured on the IBM SP2 supercomputer in Figure 4. The speed-up is close to linear on all computers, when the number of processors is small. The resulting curves obtained on the IBM SP2 are very close to each other, because of perfect management of parallel processors on this distributed memory architecture. The PC processors are newer and faster, therefore, program execution time is significantly shorter on the PC clusters. The reduction of the efficiency owing to communication overhead is obtained for a larger number of processors (Figure 5). The communication cost is quite small for relatively large problems (meshes of 90000 and 160000 elements), where a large number of finite elements are used per processor. In case of the benchmark problem with 160000 finite elements, even the super-linear speed-up is obtained. It is caused by occurred advantageous cashing and by the lack of RAM on a single PC. The processors of Transtec PC cluster are faster, but the speed of the network is the same as that of the cluster VILKAS. This is the reason why Transtec PC cluster solves the small problem of 10000 elements with lower efficiency. The implemented domain decomposition strategy and parallel GMRES solver are well designed for solving convective transport problems [8], because of the long time necessary for assembling GLS finite element coefficient matrices. The implemented domain decomposition performs this time consuming task without any inter-processor communication.

5. Conclusions

In this paper, parallel GMRES solution of convective transport problems on distributed memory computers has been investigated. The space-time FEM conception has allowed evident increase in accuracy, but it has required additional computing resources. The introduced parallel algorithms have significantly reduced the computing time. The universal domain decomposition strategy has been successfully applied to solve convection transport problems on distributed memory PC clusters. The static load balancing on the homogeneous parallel machines has been ensured by mesh partitioning code METIS incorporated in the pre-processor of the software. The iterative GMRES solver has been successfully parallelised developing the data structures and the communication algorithm particularly suited for distributed memory computers. The solution of the benchmark problem and performed efficiency analysis has illustrated high efficiency of the parallel algorithms. This has occurred due to the long time necessary for assembling GLS finite

element coefficient matrices. The implemented domain decomposition has perfectly parallelised this part of computations without any inter-processor communications. The solution of linear equation systems requires extensive communications among processors, which automatically decrease the desirable speed-up. The favourable time ratio of assembling finite element coefficient matrices to solving linear equation systems has allowed the achieving of the best efficiency on the distributed memory BEOWULF clusters.

References

[1] **C. Farhat, F.X. Roux.** Implicit parallel processing in structural mechanics. *Computational Mechanics Advances*, *Vol.* 2(1), 1994, 15-26.

[2] **R. Glowinsky, B. Golub, G. Meurant, J. Periaux.** (eds.) Proceedings Of First International Symposium On Domain Decomposition Methods For PDEs. *Philadelphia: SIAM Publications*, 1987.

[3] **V. Haroutunian, M.S. Engelman, I. Hasbani.** Segregated Finite Element Algorithms For The Numerical Solution Of Large-scale Incompressible Flow Problems. *Int. J. Numerical Methods in Engineering, Vol.* 17(2), 1993, 323-348.

[4] **T.J.R. Hughes, L.P. Franca, G.M. Hulbert.** A New Finite Element Formulation For Computational Fluid Dynamics: VIII. The Galerkin/Least-Squares Method For Advective-Diffusive Equations. *Computer Methods in Applied Mechanics and Engineering*, *Vol.* 73(2), 1989, 173-189.

[5] **C. Johnson, V. Navert. J. Pitkaranta.** Finite Element Methods For Linear, Hyperbolic Problems. *Computer Methods in Applied Mechanics and Engineering, Vol.* 45, 1984, 285-312.

[6] **G. Karypis, V. Kumar.** A Parallel Algorithm for Multilevel Graph Partitioning and Sparse Matrix Ordering. *J. Parallel and Distributed Computing, Vol.* 48(1), 1998, 71-95.

[7] **A. Kačeniauskas.** Parallel solution of convective transport equation by GLS stabilised space-time FEM. *Mechanika*, *Vol.* 4(42), 2003, 61-66.

[8] **A. Kačeniauskas, P. Rutschmann.** Parallel FEM software for CFD problems. Submitted to *Informatica*, 2004.

[9] **A. Klawonn.** Block-triangular preconditioners for saddle point problems with a penalty term. *SIAM J. Scientific Computing*, *Vol.* 19(1), 1998, 172-184.

[10] **J. Mandel.** Two level domain decomposition preconditioning for the *p*-version finite element method in three dimensions. *Int. J. Numerical Methods in Engineering*, *Vol.* 29(4), 1990, 1095-1108.

[11] **J. Mandel.** Balancing domain decomposition. *Communications in Numerical Methods in Engineering*, *Vol.* 9(2), 1993, 233-241.

[12] **A. Masud, T.J.R. Hughes.** A Space-Time Galerkin/ Least-Squares Finite Element Formulation Of The Navier-Stokes Equations For Moving Domain Problems. *Computational Methods in Applied Mechanics and Engineering, Vol.* 146(1-2), 1997, 91-126.

[13] **L.F. Pavarino.** Indefinite Overlapping Schwarz Methods For Time-dependent Stokes Problems. *Computational Methods in Applied Mechanics and Engineering, Vol.* 187(1-2), 2000, 35-51.

[14] **P. Rutschmann.** FE Solver with 4D finite elements in space and time. *Report, Swiss Federal Institute of Technology*, 1995, 1-10.

[15] **Y. Saad, M. Schultz.** GMRES: A Generalized Minimal Residual Algorithm For Solving Nonsymmetric Linear Systems. *SIAM J. Scientific and Statistical Computing, Vol.* 7(3-4), 1986, 856-859.

[16] **Y. Saad, J. Zhang.** BILUTM: a domain-based multilevel block ILUT preconditioner for general sparse matrices. *SIAM J. Matrix Analysis in Applications*, *Vol.* 21(1), 1999, 279-299.

[17] **E. De Sturler, H.A. Van der Vorst.** Reducing The Effect Of Global Communication In GMRES(*m*) And CG On Parallel Distributed Memory Computers. *Applied Numerical Mathematics, Vol.* 18(4), 1995, 441-459.

[18] **P. Le Tallec, A. Patra.** Non-overlapping Domain Decomposition Methods For Adaptive *hp* Approximations Of The Stokes Problem With Discontinuous Pressure Fields. *Computational Methods in Applied Mechanics and Engineering, Vol.* 145(3-4), 1997, 361-379.

[19] **J. Zhang.** A Multilevel Dual Reordering Strategy For Robust Incomplete LU Factorization Of Indefinite Matrices. *SIAM J. on Matrix Analysis in Applications, Vol.* 22(3), 2001, 925-947.

[20] **O.C. Zienkiewicz, R.L. Taylor.** The Finite Element Method. *Vol.* 1-2, *the fourth edition*, *McGraw-Hill*, London, 1991.