TASK QUARTERLY 11 No 1–2, 117–128

PRECONDITIONING OF VOXEL FEM ELLIPTIC SYSTEMS

SVETOZAR MARGENOV AND YAVOR VUTOV

Institute for Parallel Processing, Bulgarian Academy of Sciences,
Acad. G. Bonchev, bl. 25A, 1113 Sofia, Bulgaria
{margenov, yavor}@parallel.bas.bg

(Received 29 December 2006; revised manuscript received 29 January 2007)

Abstract: The presented comparative analysis concerns two iterative solvers for large-scale linear systems related to μFEM simulation of human bones. The considered scalar elliptic problems represent the strongly heterogeneous structure of real bone specimens. The voxel data are obtained with high resolution computer tomography. Non-conforming Rannacher-Turek finite elements are used to discretize of the considered elliptic problem. The preconditioned conjugate gradient method is known to be the best tool for efficient solution of large-scale symmetric systems with sparse positive definite matrices. Here, the performance of two preconditioners is studied, namely modified incomplete Cholesky factorization, MIC(0), and algebraic multigrid. The comparative analysis is mostly based on the computing times to run the sequential codes. The number of iterations for both preconditioners is also discussed. Finally, numerical tests of a novel parallel MIC(0) code are presented. The obtained parallel speed-ups and efficiencies illustrate the scope of efficient applications for real-life large-scale problems.

Keywords: FEM, PCG, MIC(0), AMG, parallel algorithms

1. Introduction

These days computational science entails interdisciplinary research, tackling complex scientific and engineering problems under the unifying concept of computation. The explosive growth of computer performance and progress in numerical methods and interfaces extend the power of scientific computation to an ever larger set of problems, while suggesting new ideas for experimental research. The present study is motivated by the development and tuning of robust iterative solution methods, algorithms and software tools for μFE (micro finite element) simulation of human bones. The problem includes a voxel representation of bone structure based on micro computer tomography (CT) images. We consider an isotropic 3D elliptic partial differential equation. This scalar problem is an [[inherent brick]] in the development of efficient solvers for the related coupled problems involved in nonlinear elasticity and porous elasticity μFE simulation of bone structures.

The computational domain is a complicated heterogeneous composition of solid and fluid phases. The figure below presents the solid phase of a micro mesh detail of a 2.5 mm cubic portion of a bone specimen with 44μm voxels [1].

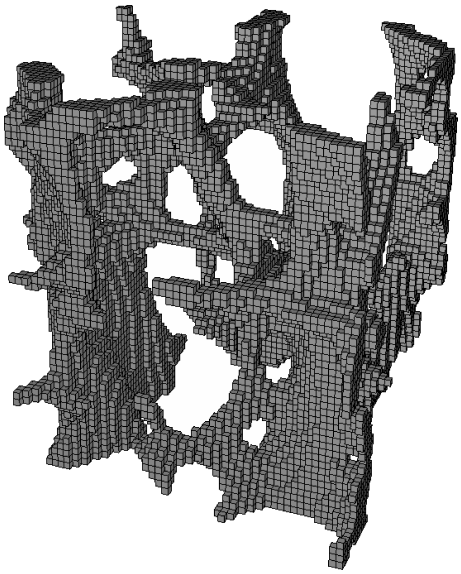

Figure 1. Micro mesh detail of the solid phase of a human trabecular bone

Non-conforming Rannacher-Turek FEs are used to discretize the problem. The obtained linear system is large with a sparse, symmetric and positive definite matrix. This implies the use of iterative solvers based on the preconditioned conjugate gradient (PCG) method [2]. Here, the performance of two PCG codes is studied: (a) modified incomplete factorization, MIC(0), and (b) algebraic multigrid, AMG. The former has been developed in IPP-BAS, Sofia, while the AMG code is the BoomerAMG module of the Hypre software system developed at LLNL, Livermore. The comparative analysis is focused on the number of iterations and the related computing times for real-life large-scale problems.

The paper is organized as follows. In Section 2 we describe the Finite Element Method (FEM) setting of the problem. Brief information about MIC(0) and BoomerAMG preconditioners is given in Section 3 followed by comparative numerical tests for the related sequential codes. Section 4 is devoted to the recently proposed parallel MIC(0) algorithm for 3D Rannacher-Turek FEM systems. Parallel numerical tests are included, performed on an IBM SP Cluster.

2. Non-conforming FEM formulation of the problem

Let us consider the elliptic boundary value problem:

$$\begin{aligned} Lu \equiv -\nabla\cdot(a(x)\nabla u(x)) &= f(x) && \text{in } \Omega, \\ u &= 0 && \text{on } \Gamma_D, \\ (a(x)\nabla u(x))\cdot n &= 0 && \text{on } \Gamma_N, \end{aligned} \tag{1}$$

where Ω is a parallelogram domain in $\mathbb{R}^3$ which can be decomposed into $n_1 \times n_2 \times n_3$ cubes. Each of these cubes corresponds to a voxel from the CT image of a bone specimen. The problem is isotropic. The $a(x)$ coefficient is assumed to be piece-wise constant with jumps aligned with the voxel mesh:

$$a(x) = \begin{cases} 1 & \text{for } x \in \Omega_s, \\ \zeta & \text{for } x \in \Omega \setminus \Omega_s, \end{cases} \tag{2}$$

where Ω_s stands for the solid phase of bone volume, $\Omega \setminus \Omega_s$ corresponds to the fluid phase, and ζ is a constant parameter in $(0,1]$.

The weak formulation of problem (1) reads as follows: given $f \in L^2(\Omega)$ find $u \in \mathscr{V} \equiv H^1_D(\Omega) = \{v \in H^1(\Omega) : v = 0 \text{ on } \Gamma_D\}$, satisfying:

$$\mathscr{A}(u,v) = (f,v) \quad \forall v \in H^1_D(\Omega), \text{ where } \quad \mathscr{A}_h(u,v) = \int_\Omega a(x) \nabla u(x) \cdot \nabla v(x) dx. \tag{3}$$

The variational problem (3) is then discretized using the finite element method, *i.e.*, the continuous space $\mathscr{V}$ is replaced by a finite dimensional subspace $\mathscr{V}_h$. Then, the finite element formulation is: find $u_h \in \mathscr{V}_h$, satisfying:

$$\mathscr{A}(u_h,v_h) = (f,v_h) \quad \forall v_h \in \mathscr{V}_h, \text{ where } \quad \mathscr{A}_h(u_h,v_h) = \sum_{e \in \mathscr{T}_h} \int_e a(e) \nabla u_h \cdot \nabla v_h dx. \tag{4}$$

Thus, the resulting discrete problem to be solved is a linear system of equations:

$$A \boldsymbol{u}_h = \boldsymbol{f}_h, \tag{5}$$

where $\boldsymbol{u}_h$ stands for the vector of unknown degrees of freedom, A and $\boldsymbol{f}_h$ being the corresponding global stiffness matrix and global right hand side, h being the discretization (mesh size) parameter for the underlying partition $\mathscr{T}_h$ of Ω. The aim of this paper is to investigate high performance preconditioners for solving system (5).

Non-conforming finite elements based on *rotated* multilinear shape functions were introduced by Rannacher and Turek [3] as a class of simple elements for the Stokes problem. More generally, the recent activities in the development of efficient solution methods for non-conforming finite element systems are inspired by their attractive properties as a stable discretization tool for ill-conditioned problems. Further details of non-conforming finite elements can be found in [4–6]. The $[-1,1]^3$ cube is used as a reference element, $\hat{e}$, to define the isoparametric rotated trilinear element $e \in \mathscr{T}_h$. Let $\Psi_e : \hat{e} \to e$ be the corresponding trilinear one-to-one transformation, and let the nodal basis functions be determined by the following relation:

$$\{\phi_i\}_{i=1}^6 = \{\hat{\phi}_i \circ \Psi_e^{-1}\}_{i=1}^6, \qquad \{\hat{\phi}_i\} \in \text{ span } \{1,x,y,z,y^2-x^2,x^2-z^2\}, \tag{6}$$

where $(x_1,x_2,x_3) \equiv (x,y,z)$, '$\circ$' denotes the composition of functions. For the MP (mid point) variant, $\{\hat{\phi}_i\}_{i=1}^6$ are found by the point-wise interpolation condition:

$$\hat{\phi}_i(b_e^j) = \delta_{ij}, \tag{7}$$

where b_e^j, j=1,6 are the centers of gravity of the walls of cube $\hat{e}$. Then:

$$\begin{aligned}
\hat{\phi}_1(x,y,z) &= \left(1-3x+2x^2-y^2-z^2\right)/6, \\
\hat{\phi}_2(x,y,z) &= \left(1+3x+2x^2-y^2-z^2\right)/6, \\
\hat{\phi}_3(x,y,z) &= \left(1-x^2-3y+2y^2-z^2\right)/6, \\
\hat{\phi}_4(x,y,z) &= \left(1-x^2+3y+2y^2-z^2\right)/6, \\
\hat{\phi}_5(x,y,z) &= \left(1-x^2-y^2-3z+2z^2\right)/6, \\
\hat{\phi}_6(x,y,z) &= \left(1-x^2-y^2+3z+2z^2\right)/6.
\end{aligned} \tag{8}$$

Alternatively, for the MV variant, the integral mean-value interpolation operator is applied in the following form:

$$|\Gamma_{\hat{e}}^j|^{-1} \int_{\Gamma_{\hat{e}}^j} \hat{\phi}_i d\Gamma_{\hat{e}}^j = \delta_{ij}, \tag{9}$$

$\Gamma^j_{\hat{e}}$, $j=1,6$ being the cube walls, and then:

$$\begin{aligned}
\hat{\phi}_1(x,y,z) &= \left(2-6x+6x^2-3y^2-3z^2\right)/12,\\
\hat{\phi}_2(x,y,z) &= \left(2+6x+6x^2-3y^2-3z^2\right)/12,\\
\hat{\phi}_3(x,y,z) &= \left(2-3x^2-6y+6y^2-3z^2\right)/12,\\
\hat{\phi}_4(x,y,z) &= \left(2-3x^2+6y+6y^2-3z^2\right)/12,\\
\hat{\phi}_5(x,y,z) &= \left(2-3x^2-3y^2-6z+6z^2\right)/12,\\
\hat{\phi}_6(x,y,z) &= \left(2-3x^2-3y^2+6z+6z^2\right)/12.
\end{aligned} \tag{10}$$

Both variants have similar properties with respect to the solution methods for the related FEM systems. In the following Section we present numerical tests for the MV case, due to the approximation advantages of this variant reported in the literature (see *e.g.* [3]).

3. PCG algorithms

The PCG is known to be the best algorithm for solving large sparse systems of linear equations with a positive definite matrix. Crucial for its performance is the preconditioning technique used. Here, we focus on two preconditioners: modified incomplete Cholesky factorization, MIC(0), and the algebraic multigrid method in the BoomerAMG variant, developed at Lowrance Livermore National Laboratory (LLNL).

3.1. MIC(0) preconditioning

In this section we briefly discuss modified incomplete factorization ([7], see also [8, 9]). Our presentation will follow [10]. Let us rewrite the real $N\times N$ matrix $A=(a_{ij})$ in the following form

$$A = D - L - L^T, \tag{11}$$

where D is the diagonal and $(-L)$ is the strictly lower triangular part of A. Then, we consider the approximate factorization of A of the following form:

$$C_{\mathrm{MIC}(0)} = (X-L)X^{-1}(X-L)^T, \tag{12}$$

where $X=\mathrm{diag}(x_1,\ldots,x_N)$ is a diagonal matrix determined by the condition of equal rowsums. We are interested in the case when $X>0$ and thus $C_{\mathrm{MIC}(0)}$ is positive definite for the purpose of preconditioning. If this holds, we speak about *stable* MIC(0) factorization. Concerning the stability of MIC(0) factorization, we have the following theorem.

THEOREM 1. Let $A=(a_{ij})$ be a symmetric real $N\times N$ matrix and let $A=D-L-L^T$ be the splitting of A. Let us assume that:

$$\begin{aligned}
L &\geq 0,\\
A\boldsymbol{e} &\geq 0,\\
A\boldsymbol{e}+L^T\boldsymbol{e} &> 0, \quad \boldsymbol{e}=(1,\ldots,1)^T\in R^N,
\end{aligned} \tag{13}$$

i.e. that A is a weakly diagonally dominant matrix with nonpositive offdiagonal entries and that $A+L^T=D-L$ is strictly diagonally dominant. Then the relation:

$$x_i = a_{ii} - \sum_{k=1}^{i-1} \frac{a_{ik}}{x_k} \sum_{j=k+1}^{N} a_{kj} > 0 \tag{14}$$

holds and the diagonal matrix $X = \mathrm{diag}(x_1, \ldots, x_N)$ defines stable MIC(0) factorization of A.

REMARK 1. The presented numerical tests have been performed using the perturbed version of the MIC(0) algorithm, where incomplete factorization is applied to the $\tilde{A} = A + \tilde{D}$ matrix. Diagonal perturbation, $\tilde{D} = \tilde{D}(\xi) = \mathrm{diag}(\tilde{d}_1, \ldots, \tilde{d}_N)$, is defined as follows:

$$\tilde{d}_i = \begin{cases} \xi a_{ii} & \text{if} \quad a_{ii} \geq 2w_i, \\ \xi^{1/2} a_{ii} & \text{if} \quad a_{ii} < 2w_i, \end{cases} \tag{15}$$

where $0 < \xi < 1$ is a parameter, while $w_i = \sum_{j>i} -a_{ij}$.

3.2. BoomerAMG

BoomerAMG contains sequential and parallel implementations of algebraic multigrid methods [11]. It can be used both as a solver and as a preconditioner. Various parallel coarsening techniques and relaxation schemes are available. (See [12, 13] for a detailed description of the coarsening algorithms, interpolation and numerical results.) The following coarsening techniques are available:

- Cleary-Luby-Jones-Plassman (CLJP) coarsening,
- various variants of the classical Ruge-Stüben (RS) coarsening algorithm and
- Falgout coarsening, which is a combination of CLJP and the classical RS coarsening algorithm.

The following relaxation techniques are available:

- Jacobi relaxation,
- the hybrid Gauss-Seidel / Jacobi relaxation scheme,
- the symmetric hybrid Gauss-Seidel / Jacobi relaxation scheme, and
- Gauss-Seidel relaxation.

3.3. Sequential numerical tests

The tests included in this section were performed on a computer with an Athlon64 processor running at 2.0GHz with 4GB of RAM. The structure of the solid phase for the $(32 \times 32 \times 32)$ and $(64 \times 64 \times 64)$ domains was extracted from a real CT image of a trabecular human bone. The voxel size was $40\,\mu$m. A mirror reflection technique was applied to get the larger problems (see Figure 2), which is a standard procedure (see, *e.g.*, [14] and the references therein) to construct a sequence of discrete problems when computational scalability is studied.

REMARK 2. It is well known that bone microstructure is not periodical. Here, a mirror reflection has been applied only for benchmarking the studied iterative solvers. It is important to note that we have used no periodical properties in the construction of the studied preconditioners.

Both MIC(0) and BoomerAMG are used as preconditioners with a relative stopping criteria in the following form:

$$(C^{-1} r^{N_{it}}, r^{N_{it}}) / (C^{-1} r^0, r^0) < 10^{-6}, \tag{16}$$

where r^i is the current residual and C stands for the used preconditioner. The size of discrete problem N and coefficient jump ζ varies, while $N = (n_1 + 1) n_2 n_3 +$

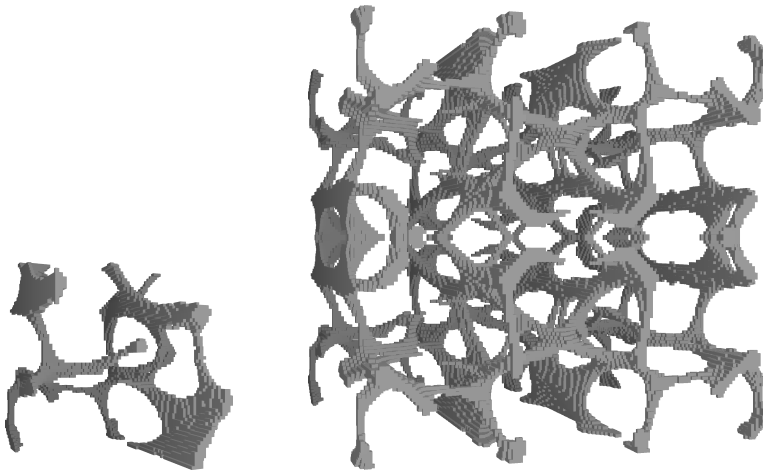

Figure 2. Structure of the solid phase: $64\times64\times64$ (left), and $128\times128\times128$ (right)

$n_1(n_2+1)n_3+n_1n_2(n_3+1)$, n_i is the number of mesh intervals along the x_i direction, $i\in\{1,2,3\}$.

Numerical tests for the MIC(0) and BoomerAMG preconditioners are respectively presented in Table 1 and Table 2. They include the obtained numbers of iterations, N_{it}, and the related total execution times, t, measured in seconds. The default settings of BoomerAMG were used: the Falgout coarsening consisted of the classical Ruge-Stueben coarsening, followed by CLJP using the interior coarse points generated by Ruge-Stueben coarsening as its first independent set. A V(1, 1)-cycle was performed with hybrid Gauss-Seidel smoothing. The related AMG strength threshold was 0.25.

Table 1. MIC(0)

		$\zeta=1$		$\zeta=0.1$		$\zeta=0.01$		$\zeta=0.001$	
$n_1\times n_2\times n_3$	N	N_{it}	t	N_{it}	t	N_{it}	t	N_{it}	t
$32\times32\times32$	101 376	27	0.86	46	1.55	121	3.22	187	5.11
$32\times32\times64$	201 728	26	1.58	36	2.27	115	6.08	296	15.3
$32\times64\times64$	401 408	26	3.04	41	5.51	117	13.1	379	47.8
$64\times64\times64$	798 720	35	7.84	56	14.5	166	41.3	417	103
$64\times64\times128$	1 593 344	33	14.6	52	28.1	141	69.3	453	220
$64\times128\times128$	3 178 496	33	27.7	53	54.2	142	134	467	440
$128\times128\times128$	6 340 608	47	76.8	72	122	197	337	575	981

Notably, $N_{it}=O(\sqrt{n_1})$ in the case of MIC(0), which is in full agreement with the existing theoretical estimates and with computational practice. For the largest problem, $128\times128\times128$, N_{it} increases more than 12 times when ζ decreases from 1 to 0.001. The iteration counts are much better for BoomerAMG. The multigrid iterations are almost optimal with respect to the problem size, N, increasing slightly with the coefficient jumps, while the AMG theory does not support the case of so strongly heterogeneous media.

It is important to note that the structure of the algorithm and the memory consumption of the MIC(0) code does not depend on the coefficients. This is not the

case with BoomerAMG, which uses about 4 times more memory than the MIC(0) for the Laplace equation ($\zeta = 1$) and about 10 times more in the presence of coefficient jumps. The shortage of the available RAM was the reason to skip the last two largest tests for BoomerAMG in the cases of $\zeta \in \{0.1, 0.01, 0001\}$. The number of iterations for the MIC(0) algorithm depends heavily on the perturbation parameter, ξ.

Table 2. BoomerAMG

		$\zeta = 1$		$\zeta = 0.1$		$\zeta = 0.01$		$\zeta = 0.001$	
$n_1 \times n_2 \times n_3$	N	N_{it}	t	N_{it}	t	N_{it}	t	N_{it}	t
$32 \times 32 \times 32$	101 376	7	3.65	8	5.42	9	5.79	11	6.18
$32 \times 32 \times 64$	201 728	7	6.83	8	9.55	8	10.0	10	10.0
$32 \times 64 \times 64$	401 408	7	13.4	8	27.3	10	29.2	11	30.5
$64 \times 64 \times 64$	798 720	8	28.5	9	58.9	12	57.1	13	52.7
$64 \times 64 \times 128$	1 593 344	8	58.1	9	145	11	153	15	181
$64 \times 128 \times 128$	3 178 496	8	123						
$128 \times 128 \times 128$	6 340 608	8	252						

The ultimate aim of this analysis is to compare the total computing times. The presented tests demonstrate that MIC(0) performs better in all cases excluding the strongest coefficient jump of $\zeta = 0.001$. The time required to construct the preconditioner (for recursive approximate factorization) is relatively long for the BoomerAMG implementation of the general purpose algebraic multigrid algorithms. The better times for BoomerAMG are indicated in Table 2 by gray background. Importantly, MIC(0) provides an alternative choice for efficient solution of voxel FEM elliptic systems in cases of small to moderate coefficient jumps. For stronger jumps and larger problems (if they fit the available RAM), BoomerAMG will outperform the tested MIC(0) code.

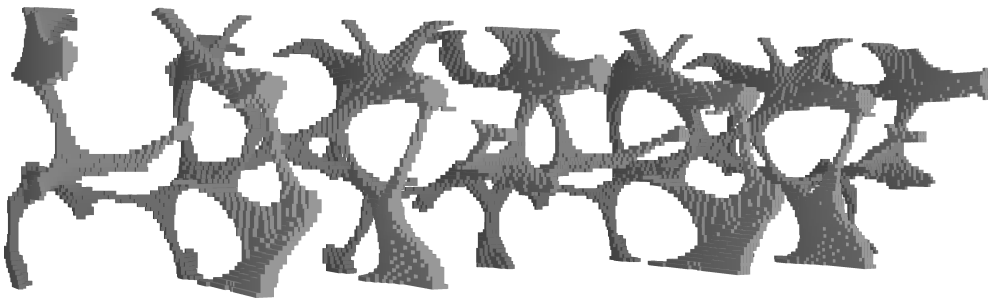

Figure 3. Structure of the solid phase: $64 \times 64 \times 256$

Let us consider the case when the domain has stronger geometrical anisotropy (see Figure 3). Such problems naturally appear in bone simulations. Let $n_1 = n_2$ and $n_3 = kn_1$ for $k \in \{1, 2, 4, 8\}$; the results for both solvers are presented in Tables 3 and 4.

Again, with varying n_3 for fixed ζ the number of iterations remains much the same, which is generally expected for the optimal BoomerAMG solver. More interesting is the obtained optimal convergence rate for the MIC(0) preconditioner. Both solvers scale relatively well in all cases. Doubling the problem size approximately

Table 3. MIC(0): domain anisotropy

		$\zeta = 1$		$\zeta = 0.1$		$\zeta = 0.01$		$\zeta = 0.001$	
$n_1 \times n_2 \times n_3$	N	N_{it}	t	N_{it}	t	N_{it}	t	N_{it}	t
$32 \times 32 \times 32$	101 376	27	0.86	46	1.55	121	3.22	187	5.11
$32 \times 32 \times 64$	201 728	26	1.58	36	2.27	115	6.08	296	15.3
$32 \times 32 \times 128$	402 432	29	3.94	42	5.28	124	15.3	357	43.3
$32 \times 32 \times 256$	803 840	27	6.78	41	10.3	118	27.2	286	65.8
$64 \times 64 \times 64$	798 720	35	7.84	56	14.5	166	41.3	417	103
$64 \times 64 \times 128$	1 593 344	33	14.6	52	28.1	141	69.3	453	220
$64 \times 64 \times 256$	3 182 592	39	36.4	54	49.1	148	131	449	399
$64 \times 64 \times 512$	6 361 088	37	66.6	57	98.3	161	265	504	835

Table 4. BoomerAMG: domain anisotropy

		$\zeta = 1$		$\zeta = 0.1$		$\zeta = 0.01$		$\zeta = 0.001$	
$n_1 \times n_2 \times n_3$	N	N_{it}	t	N_{it}	t	N_{it}	t	N_{it}	t
$32 \times 32 \times 32$	101 376	7	3.65	8	5.42	9	5.79	11	6.18
$32 \times 32 \times 64$	201 728	7	6.83	8	9.55	8	10.0	10	10.0
$32 \times 32 \times 128$	402 432	7	14.1	9	26.0	10	26.0	11	26.3
$32 \times 32 \times 256$	803 840	7	28.2	9	48.1	11	49.6	10	44.7
$64 \times 64 \times 64$	798 720	8	28.5	9	58.9	12	57.1	13	52.7
$64 \times 64 \times 128$	1 593 344	8	58.1	9	145	11	153	15	181
$64 \times 64 \times 256$	3 182 592	7	115						
$64 \times 64 \times 512$	6 361 088	8	246						

doubles the computational time. The scalability is even better expressed for MIC(0) in the case of stronger coefficient jumps. As in Table 2, the data missing for BoomerAMG are due to the lack of sufficient RAM. The better times for BoomerAMG are again markeed with gray background.

4. Parallel MIC(0)

As has been mentioned, BoomerAMG also has a parallel implementation. Less popular is a parallel algorithm for MIC(0) preconditioning of Rannacher-Turek FEM systems. The last section of this article is concerned with the recently developed parallel MIC(0) for 3D Rannacher-Turek problems. A separate article will be devoted to a comparative analysis of the parallel performance of MIC(0) and BoomerAMG.

4.1. The parallel algorithm

The idea of the algorithm is to apply MIC(0) factorization to an auxiliary matrix B. The matrix has a special block structure allowing scalable parallel implementation.

Following the standard FEM assembling procedure, we write A in the form $A = \sum_{e \in \omega_h} L_e^T A_e L_e$, where A_e is the element stiffness matrix, L_e stands for the restriction mapping of the global vector of unknowns to the local one corresponding to the current element e. Let us consider the following approximation B_e of A_e:

$$A_e = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} & a_{15} & a_{16} \\ a_{21} & a_{22} & a_{23} & a_{24} & a_{25} & a_{26} \\ a_{31} & a_{32} & a_{33} & a_{34} & a_{35} & a_{36} \\ a_{41} & a_{42} & a_{43} & a_{44} & a_{45} & a_{46} \\ a_{51} & a_{52} & a_{53} & a_{54} & a_{55} & a_{56} \\ a_{61} & a_{62} & a_{63} & a_{64} & a_{65} & a_{66} \end{bmatrix}, \qquad B_e = \begin{bmatrix} b_{11} & a_{12} & a_{13} & a_{14} & a_{15} & a_{16} \\ a_{21} & b_{22} & a_{23} & a_{24} & a_{25} & a_{26} \\ a_{31} & a_{32} & b_{33} & 0 & 0 & 0 \\ a_{41} & a_{42} & 0 & b_{44} & 0 & 0 \\ a_{51} & a_{52} & 0 & 0 & b_{55} & 0 \\ a_{61} & a_{62} & 0 & 0 & 0 & b_{66} \end{bmatrix}. \tag{17}$$

The local numbering follows the pairs of opposite nodes of the reference element. The diagonal entries of B_e are modified to hold the rowsum criteria. Assembling the locally defined matrices B_e, we obtain the global matrix: $B = \sum_{e \in \omega_h} L_e^T B_e L_e$.

The sparsity structure of matrices A and B is illustrated in Figure 4, with the use of lexicographic node numbering. An important property of matrix B is that its diagonal blocks are diagonal matrices. This allows well-scalable parallel implementation (see [15–17]), confirmed by the tests presented at the end of this section.

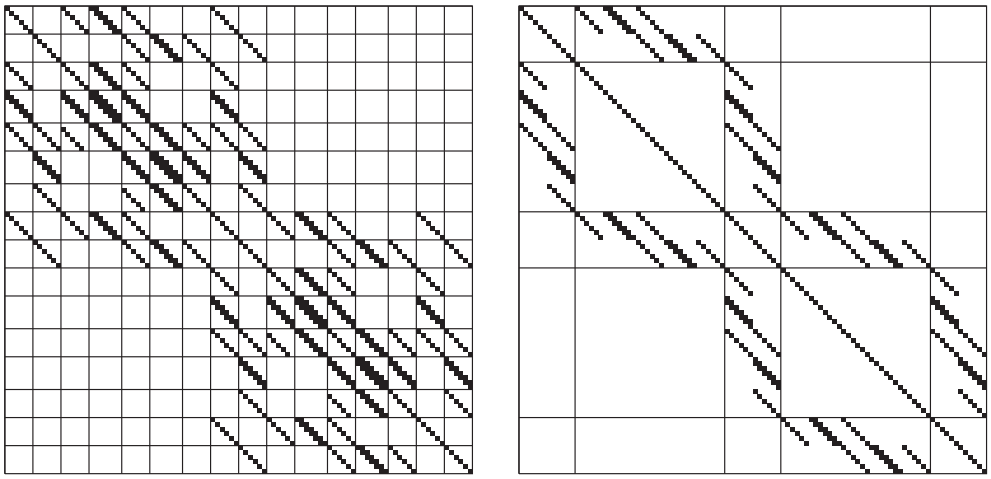

Figure 4. Sparsity structure of matrix A (left) and matrix B (right) for division of Ω into $2 \times 2 \times 6$ hexahedrons. Non-zero elements are marked as small squares

4.2. Convergence tests

The definition of B ensures a uniform spectral condition number estimate, $\kappa(B^{-1}A) = O(1)$. Here, we compare the convergence of the sequential MIC(0) and the parallel variant, *i.e.* MIC(0) applied to auxiliary matrix B. Diagonal compensation is used to remove the positive offdiagonal entries and ensure stable MIC(0) factorization of A in the sequential algorithm. The numbers of iterations are collected in Table 5 for both methods. Their convergence is very similar, but the computational complexity of one PCG iteration is cheaper for the second variant as B is considerably sparser than A.

Table 5. Number of iterations for MIC(0) and Parallel MIC(0)

$n_1 \times n_2 \times n_3$	MIC(0)	Parallel MIC(0)
$32 \times 32 \times 32$	23	27
$64 \times 64 \times 64$	33	35
$128 \times 128 \times 128$	46	47

As can be seen in Tables 1–5, the sequential tests presented in Section 3.3 have been performed using the modified (parallel) version of MIC(0).

4.3. Parallel tests

The parallel MIC(0) algorithm was programmed using the Message Passing Interface library (MPI). The presented tests were performed on four parallel platforms. Platform C1 was an IBM SP Cluster 1600 made of 64 nodes p5-575 interconnected with a pair of connections to the Federation HPS (High Performance Switch). Each p5-575 node contained 8 SMP Power5 processoer of 1.9 GHz and 16 GB of RAM. Platform C2 was a Cray XD1 cabinet, fully populated with 72 2-way nodes, with a total of 144 AMD 64-bit Opteron processors of 2.4 GHz. Each node had 4 GBs of memory. The CPUs of C2 were interconnected with the Cray RapidArray network. Platforms C3 and C4 provided different interconnections for one and the same cluster. They were made of 32 nodes each with 2 AMD Opteron processors of 2.4 GHz and 4 GBs of memory. The processors run in the 32 bit mode. The communication of C3 was over a normal gigabit ethernet, while C4 used an M3F-PCIXD-2 Myrinet/PCI-X network interface.

Basing on the conclusions from the previous subsections, we concentrate here on the parallel speedups and efficiencies presented in Table 6 and Table 7. Since the cost of constructing the MIC(0) preconditioner is roughly about 2/3 of the cost of one iteration – only times per iteration are given. The number of iterations for the parallel MIC(0) solver does not depend on the number of processors, p.

The first set of parallel experiments were performed with $n_1 = n_2 = n_3 = n$ (see Table 6).

Table 6. Parallel Tests I

		C1			C2			C3			C4		
p	n	T_p	S_p	E_p	T_p	S_p	E_p	T_p	S_p	E_p	T_p	S_p	E_p
1		0.0204	1.00	1.00	0.0268	1.00	1.00	0.0257	1.00	1.00	0.0236	1.00	1.00
2		0.0111	1.8	0.92	0.0161	1.66	0.83	0.0263	0.97	0.48	0.0157	1.50	0.75
4	31	0.0072	2.8	0.70	0.0115	2.33	0.58	0.0275	0.93	0.23	0.0116	2.04	0.51
8		0.0046	4.4	0.55	0.0067	4.01	0.50	0.0281	0.91	0.11	0.0082	2.88	0.36
16		0.0033	6.0	0.38	0.0046	5.72	0.36						
1		0.176	1.00	1.00	0.227	1.00	1.00	0.221	1.00	1.00	0.200	1.00	1.00
2		0.090	1.94	0.97	0.117	1.94	0.97	0.137	1.62	0.81	0.118	1.70	0.85
4	63	0.049	3.56	0.89	0.071	3.18	0.80	0.098	2.26	0.56	0.070	2.86	0.71
8		0.026	6.70	0.84	0.045	5.01	0.63	0.082	2.69	0.34	0.047	4.28	0.53
16		0.017	10.5	0.66	0.033	6.84	0.43						
1		1.57	1.00	1.00	2.09	1.00	1.00	1.82	1.00	1.00	1.92	1.00	1.00
2		0.81	1.94	0.97	1.08	1.92	0.96	1.08	1.69	0.84	1.11	1.73	0.87
4	127	0.44	3.56	0.89	0.55	3.83	0.95	0.62	2.94	0.74	0.53	3.62	0.90
8		0.22	7.01	0.88	0.30	6.97	0.87	0.39	4.64	0.58	0.29	6.59	0.82
16		0.12	13.16	0.82	0.19	11.12	0.69						

The parallel scalability of the solver is well expressed for the larger of the test problems. Slight differences in sequential times on platforms C3 and C4 can be explained by the use of different MPI libraries. Notably, even the sequential runs ($p = 1$) have been influenced by the related MPI library.

As the efficiency of the parallel MIC(0) algorithm depends mainly on the $n_2 n_3/p$ ratio, we repeated the parallel tests on platforms C3 and C4 in the case of domain anisotropy as introduced above. The results are presented in Table 7.

Table 7. Parallel Tests II

		C3			C4					C3			C4		
p	$n_1 \times n_2 \times n_3$	T_p	S_p	E_p	T_p	S_p	E_p	p	$n_1 \times n_2 \times n_3$	T_p	S_p	E_p	T_p	S_p	E_p
1		0.027	1.00	1.00	0.025	1.00	1.00	1		0.22	1.00	1.00	0.20	1.00	1.00
2		0.024	1.10	0.55	0.020	1.27	0.63	2		0.23	0.93	0.46	0.12	1.72	0.86
4	$32 \times 32 \times 32$	0.036	0.77	0.19	0.012	2.13	0.53	4	$64 \times 64 \times 64$	0.10	2.09	0.52	0.08	2.70	0.67
8		0.033	0.82	0.10	0.009	2.99	0.37	8		0.08	2.57	0.32	0.05	4.22	0.53
1		0.060	1.00	1.00	0.050	1.00	1.00	1		0.54	1.00	1.00	0.53	1.00	1.00
2		0.036	1.67	0.84	0.031	1.57	0.78	2		0.26	2.11	1.05	0.23	2.37	1.18
4	$32 \times 32 \times 64$	0.030	1.99	0.50	0.021	2.39	0.60	4	$64 \times 64 \times 128$	0.14	3.69	0.92	0.12	4.31	1.07
8		0.030	1.98	0.25	0.012	4.08	0.51	8		0.17	3.05	0.38	0.08	6.67	0.83
1		0.107	1.00	1.00	0.116	1.00	1.00	1		0.93	1.00	1.00	0.96	1.00	1.00
2		0.063	1.71	0.85	0.057	2.02	1.01	2		0.62	1.50	0.75	0.56	1.70	0.85
4	$32 \times 32 \times 128$	0.075	1.43	0.35	0.034	3.39	0.85	4	$64 \times 64 \times 256$	0.30	3.05	0.76	0.25	3.79	0.94
8		0.034	3.14	0.39	0.021	5.43	0.68	8		0.15	6.04	0.76	0.12	7.55	0.94
1		0.238	1.00	1.00	0.240	1.00	1.00	1		2.11	1.00	1.00	2.00	1.00	1.00
2		0.133	1.78	0.89	0.133	1.81	0.90	2		1.07	1.91	0.96	1.00	1.99	0.99
4	$32 \times 32 \times 256$	0.068	3.49	0.87	0.057	4.15	1.03	4	$64 \times 64 \times 512$	0.63	3.32	0.83	0.59	3.41	0.85
8		0.047	5.04	0.63	0.035	6.84	0.86	8		0.30	7.15	0.89	0.26	7.73	0.97

There is very good efficiency for larger n_3 on both types of interconnections, additionally supported by the implemented overlapping of computations and communications (for more details see [16]). As a result, the influence of latency and the networks' speed is strongly reduced with the increase of n_3. When $n_2 n_3/p > c$, c is a certain threshold depending on the interconnection, almost all communications are overlapped with the ongoing computations, and the efficiencies are close to 1. Efficiencies greater than 1 could be explained by better data localization and the consequent better memory cache utilization in some of the cases.

5. Concluding remarks

Computational scalability issues of high performance iterative solution methods, algorithms and codes have been studied and tested. Benchmarking has been focused on strongly heterogeneous scalar elliptic problems. The structure of the test problems has been taken from a CT image of a trabecular bone. The presented results enable further development of robust computational tools for μFEM analysis of bone structures. The planned next steps of this study include the application of scalar elliptic solvers in the construction of block-preconditioners for coupled elasticity and poroelasticity problems.

Acknowledgements

Partial support for this work with the EC INCO Grant BIS-21++ 016639/2005 and the Bulgarian NSF Grant VU-MI-202/2006 is gratefully acknowledged. The parallel numerical tests on platforms C1 and C2 were supported via EC Project HPC-EUROPA RII3-CT-2003-506079.

The authors would like to thank Harry van Lenthe, ETH Zurich, Switzerland, for providing the voxel finite element mesh used to construct the test problems.

References

[1] 2004 *ABAQUS Technology Brief,* **TB-03-HTB-1**

[2] Axelsson O 1994 *Iterative Solution Methods*, Cambridge University Press, Cambridge

[3] Rannacher R and Turek S 1992 *Numer. Methods Partial Differential Equations* **8** (2) 97
[4] Arnold D N and Brezzi F 1985 *RAIRO Model. Math. Anal. Numer.* **19** 7
[5] Bencheva G and Margenov S 2003 *J. Comput. Appl. Mech.* **4** (2) 105
[6] Braess D 1997 *Finite Elements. Theory, Fast Solvers, and Applications in Solid Mechanics*, Cambridge University Press
[7] Gustafsson I 1983 *Preconditioning Methods; Theory and Applications* (Evans D J, Ed.), Gordon and Breach, pp. 265–293
[8] Gustafsson I and Lindskog G 1998 *Numer. Linear Algebra Appl.* **5** (2) 123
[9] Gustafsson I 1979 *Stability and Rate of Convergence of Modified Incomplete Cholesky Factorization Methods*, Chalmers University of Technology, Report No. 79.02R
[10] Blaheta R 1994 *Numer. Linear Algebra Appl.* **1** 107
[11] Ruge J W and Stüben K 1987 *Frontiers in Applied Mathematics* (McCormick S F, Ed.), SIAM, Philadelphia, PA, **3**, pp. 73–130
[12] Henson V E and Yang U M 2002 *Appl. Num. Math.* **41** (5) 155 (also available as LLNL Technical Report UCRL-JC-141495)
[13] Yang U M 2005 *Numerical Solution of Partial Differential Equations on Parallel Computers* (Bruaset A M and Tveito A, Eds), Springer-Verlag, pp. 209–236 (also available as LLNL Technical Report UCRL-BOOK-208032)
[14] Arbenz P, van Lenthe G H, Mennel U, Müller R and Sala M 2006 *A scalable Multi-level Preconditioner for Matrix-free μ-Finite Element Analysis of Human Bone Structures*, Institute of Computational Science, ETH Zurich, Technical Report 543
[15] Arbenz P and Margenov S 2004 *Proc. IMET Conf.*, Prague, Czech Republic, pp. 12–15
[16] Arbenz P, Margenov S and Vutov Y 2007 *Comput. Math. Appl.* (to appear)
[17] Vutov Y 2007 *Lecture Notes in Computer Science*, NMA 2006, Borovets, Bulgaria, **4310** 114