On a parallel multilevel solver for linear elasticity problems *

Alexander Padiy †

Abstract

In this paper an application of the additive multilevel iteration method to parallel solving of large–scale linear elasticity problems is considered. The results are derived in the framework of the hierarchical basis finite–element discretization defined on a tensor product $\hat{T}^{(i)}_{xyz} = \hat{T}^{(i)}_{xy} \otimes \hat{T}_z$ of one–dimensional grid $\hat{T}_z$ and a sequence of nested triangulations $\hat{T}^{(i)}_{xy}$. The algorithm is tested on a number of model problems, arising from bridge foundation modeling. Parallel performance of the solver is reported for Cray T3E–600 and Sun ES/4000 computer systems.

KEY WORDS Linear elasticity problems, multilevel methods, non-uniform discretization meshes, parallel computing.

1 Introduction

Consider the elastic body Ω subjected by external surface forces $\underline{\gamma}$ and volume forces $\mathbf{g}$. The energy functional, which describes the potential energy of the elastic body under small deformations, can be expressed as

$$E(\mathbf{u}) = \int\limits_{\Omega} \left(\mu \sum_{i,j=1}^{3} (\varepsilon_{i,j}(\mathbf{u}))^2 + \frac{\lambda}{2} (\nabla \cdot \mathbf{u})^2 \right) d\Omega - \int\limits_{\Gamma_N} \underline{\gamma} \cdot \mathbf{u} \, d\Gamma - \int\limits_{\Omega} \mathbf{g} \cdot \mathbf{u} \, d\Omega. \qquad (1)$$

Here $\mathbf{u} = (u_1, u_2, u_3)$ represents the displacement field, $\varepsilon_{ij}(\mathbf{u})$ denotes the strain tensor

$$\varepsilon_{ij} = \frac{1}{2} \left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right),$$

and Γ_N denotes a part of the boundary where the external surface load $\underline{\gamma}$ is applied. The parameters λ and μ above are positive material coefficients related to the elasticity modulus E and the contractivity (Poisson) ratio ν by

$$\mu = \frac{E}{2(1+\nu)}, \quad \lambda = \mu \frac{2\nu}{1-2\nu}.$$

*This work was supported by the COPERNICUS project "High Performance Computing in Geosciences; Safety of Constructions with Respect to Rock Deformations and Movements HIPERGEOS", COP-94-00820. Parallel experiments on Cray T3E-600 were performed within the NCF/MP-010 project "Preconditioning methods for extremely large scale scientific models".

†Faculty of Mathematics and Informatics, University of Nijmegen, Toernooiveld 1, 6525 ED Nijmegen, The Netherlands, E-mail: padiy@sci.kun.nl and Institute of Numerical Mathematics, Russian Academy of Sciences, Gubkina str. 8, GSP-1, 117333 Moscow, Russia, E-mail: padiy@labnumat.inm.ras.ru.

We are interested in the equilibrium state displacement field $\mathbf{u}^*$, which can be found via solving the minimization problem

$$\text{Find } \mathbf{u}^* \ : \ E(\mathbf{u}^*) = \min_{\mathbf{u}\in V} E(\mathbf{u}). \tag{2}$$

If the function space V is defined as

$$V = \left\{ \mathbf{u} \in (H^1(\Omega))^3;\ \mathbf{u} = \mathbf{0} \text{ on } \Gamma_D = \Omega \backslash \Gamma_N, \int\limits_{\Omega} \nabla \times \mathbf{u}\, d\Omega = 0 \right\}, \tag{3}$$

then the functional $E(\mathbf{u})$ is proven to be convex (as can be found in [11], for example). This implies the existence and uniqueness of $\mathbf{u}^*$. Since in the majority of practical applications it is impossible to solve the problem (2) analytically, numerical simulations are used instead. Further we will search for the approximation $\mathbf{u}_h^*$ to $\mathbf{u}^*$ in the finite–element space $V_h \subset V$.

The minimization of $E(\mathbf{u})$ in the finite–element space V_h leads to the algebraic system

$$Ax = f \tag{4}$$

with sparse symmetric positive–definite matrix A. Even though the matrix A is guaranteed to be non-singular, the dimension of A can easily exceed millions of unknowns because of the complicated shape of Ω and due to the need to maintain acceptable approximation error in the regions where the solution is rapidly changing. This puts a very strong demand both on the numerical algorithms and on the computer hardware. Parallel supercomputers are often used to solve the systems arising from real–life engineering applications.

In principle, both direct and iterative methods can be used to solve (4). However, the use of direct methods is not feasible due to their sequential nature and very high memory and CPU requirements. Thus, one has to use an iterative procedure to compute the action of A^{-1} on f. In this context an efficient preconditioner for A becomes a very important issue. To compute the solution in a reasonable turn–around time, the preconditioning scheme has to be scalable with respect to both the size of the algebraic system and the number of processing elements used in the parallel environment. The preconditioner is also expected to be robust with respect to the jumps in the problem coefficients and work well on non-uniform discretization meshes.

In the present paper we use a SDC (separate displacement components) ordering approach to construct a block–diagonal preconditioner for the discrete linear elasticity problem. The preconditioner is based on the Korn inequality and on a newly developed modification to the additive AMLI method (we refer to [1], [2], [6], [7], [12] and references herein for the description of the AMLI framework). The method works on a hierarchical sequence of nested finite element spaces to solve the problem with arithmetic cost, nearly proportional to the number of degrees of freedom on the finest mesh. The solver is robust with respect to discontinuities of the problem coefficients and imposes only weak (and acceptable in practice) restrictions on the choice of the meshing procedure. It is particularly well suited for the case when the solution is known to be strongly varying in certain subregions of the domain and the mesh is locally pre–refined there to reduce the discretization error.

The performance of the method depends mainly on a single parameter σ, which is introduced in section 4. The parameter σ can be computed locally for each mesh element and, therefore, it does not depend on possible jumps of the coefficients between them. Moreover, there is no need to choose a very coarse mesh in the proposed algorithm. The coarsest mesh can be relatively fine and, hence, a large number of jumps in the coefficient function is allowed.

Parallel performance of the method is reported for Cray T3E-600 (DEC Alpha 21164, 300 Mhz, 3D–torus) and Sun ES/4000 (Sun UltraSparc–I, 167 Mhz, shared memory/bus) computer platforms. Both uniform and highly non–uniform discretization meshes are used in our experiments.

2 Variational formulation

Consider the problem of finding a minimizer to the energy functional $E(\mathbf{u})$ from (1) in the functional space V defined by (3). Since the functional $E(\mathbf{u})$ is convex, the variation of $E(\mathbf{u})$ equals zero at the minimization point. Thus, we have the following variational formulation equivalent to the minimization of $E(\mathbf{u})$:

$$\text{Seek } \ \mathbf{u} \in V \ \text{ such that } \ \delta E(\mathbf{u}) = 0 \ \text{ for all } \ \mathbf{v} \in V.$$

Taking into account the explicit form of $E(\mathbf{u})$, the variational problem can be rewritten as

$$\text{Seek } \ \mathbf{u} \in V \ \text{ such that } \ a(\mathbf{u},\mathbf{v}) = (\,\mathbf{g},\mathbf{v}\,) + [\,\underline{\gamma},\mathbf{v}\,] \ \text{ for all } \ \mathbf{v} \in V, \tag{5}$$

where

$$a(\mathbf{u},\mathbf{v}) = \int\limits_{\Omega} \left(2\mu \sum_{k=1}^{3} (\nabla u_k)\cdot(\nabla v_k) - \mu\,(\nabla\times\mathbf{u})\cdot(\nabla\times\mathbf{v}) + \lambda\,(\nabla\cdot\mathbf{u})(\nabla\cdot\mathbf{v}) \right) d\Omega, \tag{6}$$

$$(\,\mathbf{g},\mathbf{v}\,) = \int\limits_{\Omega} \mathbf{g}\cdot\mathbf{v}\, d\Omega, \qquad [\,\underline{\gamma},\mathbf{v}\,] = \int\limits_{\Gamma_N} \underline{\gamma}\cdot\mathbf{v}\, d\Gamma. \tag{7}$$

Consider the domain Ω being covered by a set T of finite elements (further we assume that the set T is based on the mesh $T^{(l)}$, defined by (10)). Then, by introducing the space $V_h \subset V$ consisting of vector functions with local support, associated with the vertices of T, we arrive to the following discrete variational problem:

$$\text{Seek } \ \mathbf{u}_h \in V_h \ \text{ such that } \ a(\mathbf{u}_h,\mathbf{v}_h) = (\,\mathbf{g},\mathbf{v}_h\,) + [\,\underline{\gamma},\mathbf{v}_h\,] \ \text{ for all } \ \mathbf{v}_h \in V_h. \tag{8}$$

In the sequel we assume that there are no incompressible materials involved in the problem, i.e. the Poisson ratio ν is bounded away from $1/2$. If this assumptions is satisfied, the finite–element solution $\mathbf{u}_h$ is proven to be converging to the solution of (5) when the mesh step-size h tends to zero. The case when the Poisson ratio is close to $1/2$ has to be treated differently, by means of mixed variables formulation or inexact integration techniques, for example. This issue will not be discussed further in the present paper, we refer to [4] for one of possible approaches.

3 The algebraic system and the preconditioner

The variational problem (8) leads to the following system of algebraic equations:

$$A\mathbf{u} = \mathbf{f},$$

where matrix A corresponds to the bilinear form $a(\cdot,\cdot)$ from (6) and $\mathbf{f}$ is generated by the bilinear forms $(\cdot,\cdot)$ and $[\cdot,\cdot]$ from (7). If we use the separate displacement components ordering of the unknowns, namely,

$$\mathbf{u} = \begin{bmatrix} \mathbf{u}^{(x)} \\ \mathbf{u}^{(y)} \\ \mathbf{u}^{(z)} \end{bmatrix}, \quad \text{where} \quad \mathbf{u}^{(\xi)} = \begin{bmatrix} u_1^{(\xi)} \\ \cdots \\ u_{n_\xi}^{(\xi)} \end{bmatrix}, \quad \xi = x, y, z,$$

then the matrix A admits the following block 3×3 structure

$$A = \begin{bmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{bmatrix}.$$

We will solve the system with A by the conjugate gradient method using a block–diagonal preconditioner $Q = \operatorname{diag}(Q_{11}, Q_{22}, Q_{33})$. The preconditioner Q is assumed to be generated by the bilinear form $q(\cdot,\cdot)$ defined as

$$q(\mathbf{u},\mathbf{v}) = \sum_{k=1}^{3} \tilde{q}(u_k, v_k), \qquad \tilde{q}(u_k, v_k) = \int_\Omega \mu(\nabla u_k) \cdot (\nabla v_k)\, d\Omega, \qquad \mathbf{u},\mathbf{v} \in V_h. \tag{9}$$

As follows from the Korn inequality, the condition number $\kappa(Q^{-1}A)$ can be estimated as

$$\kappa(Q^{-1}A) \le C_K \, \frac{1+\nu}{1-2\nu}.$$

Here C_K denotes the constant in the Korn inequality, which depends only on the shape of Ω and on the boundary conditions. Hence, the condition number $\kappa(Q^{-1}A)$ is independent of the number of degrees of freedom in the algebraic system. However, we still need an efficient solver for the blocks Q_{ii} when using Q as a preconditioner for A. Further we develop a new modification of the I–AMLI method to invert the blocks Q_{ii} iteratively. An advantage of the developed preconditioner is that it is not sensitive with respect to the problem parameters, is robust with respect to the discretization meshes and can be efficiently parallelized. A brief description of the method is given in the next section. As the way of constructing the preconditioner does not depend on a particular choice of the boundary conditions, we omit the subindex i further.

Since the algebraic systems with Q_{ii} are solved iteratively, up to a finite accuracy ε, the preconditioner Q is, in general, a variable–step preconditioner. This means that it is a nonlinear mapping $\mathbf{x} \to \Phi[\mathbf{x}]$. The use of variable–step preconditioners destroys the standard Krylov vector sequence and introduces also a nonsymmetry into the method. Therefore, the original preconditioned conjugate gradient method, which was primarily developed for symmetric positive definite matrices, is inapplicable for work with variable–step preconditioners. Fortunately, a method like the generalized conjugate gradient (GCG) method suggested in [5] is able to handle the nonlinearity, see also [2]. If we choose proper stopping criterion ε, then the following conditions are satisfied: a) the mapping $Q\Phi[\cdot]$ is coercive, i.e. there exists a positive constant δ_1, such that $(Q\Phi[\mathbf{v}],\mathbf{v}) \ge \delta_1(\mathbf{v},\mathbf{v})$ for all $\mathbf{v}$; b) the mapping $Q\Phi[\cdot]$ is continuous, i.e. there exists a positive constant δ_2, such that $||Q\Phi[\mathbf{v}]|| \le \delta_2||\mathbf{v}||$ for all $\mathbf{v}$. As shown in [5],

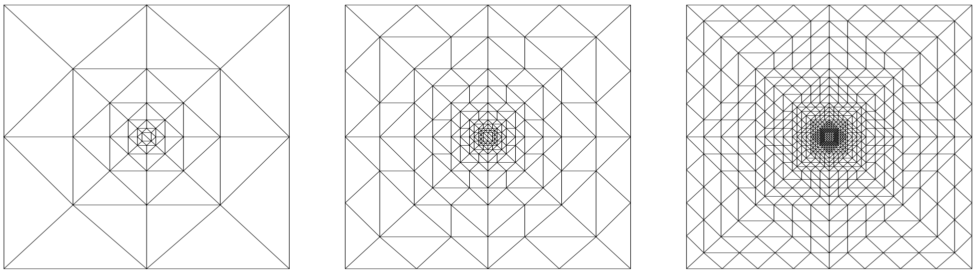

Figure 1: An example of the sequence $\{\hat{T}_{xy}^{(i)}\}$, used in our calculations

under the above assumptions the GCG method can not fail, it converges monotonically and at least with a rate given by the inequality

$$||\mathbf{r}^{(k)}||_0 \leq \sqrt{1-\left(\frac{\delta_1}{\delta_2}\right)^2}||\mathbf{r}^{(k-1)}||_0.$$

4 Multilevel preconditioner for the blocks Q_{ii}

4.1 Meshing procedure

The method is based on the hierarchical basis approach [13]. It is defined on a sequence

$$\{\hat{T}^{(i)} = \hat{T}_{xy}^{(i)} \otimes \hat{T}_z\}, \quad i = \overline{0,l}, \tag{10}$$

of three–dimensional meshes constructed as a tensor product of an arbitrary one–dimensional grid $\hat{T}_z$ and a sequence of nested two–dimensional triangulations $\hat{T}_{xy}^{(i)}$. By construction, the meshes $\hat{T}^{(i)}$ consist of prisms aligned along the z–direction.

We require the meshes $\hat{T}_{xy}^{(i)}$ to be regular, i.e. they should not contain triangles, which are close to the degenerate ones. Fortunately, this restriction is not very severe as it still allows us to have highly varying mesh step-sizes. This is illustrated in Figure 1. Since no conditions are imposed on the mesh $\hat{T}_z$, we can also take advantage of non–uniform discretization in the z–direction. This means that we can achieve a good quality of the solution using a moderate number of mesh nodes by pre–refining our three–dimensional mesh $\hat{T}^{(0)}$ in certain subregions of the domain Ω.

4.2 Operator splitting and hierarchical basis approach

Consider the following operator $\mathcal{L}$, which corresponds to the bilinear form $\tilde{q}(\cdot,\cdot)$ defined by (9):

$$\mathcal{L} = \mathcal{L}_{xy} + \mathcal{L}_z, \quad \mathcal{L}_{xy} = -\sum_{i=1}^{2} \frac{\partial}{\partial x_i}\mu(\mathbf{x})\frac{\partial}{\partial x_i}, \quad \mathcal{L}_z = -\frac{\partial}{\partial x_3}\mu(\mathbf{x})\frac{\partial}{\partial x_3}. \tag{11}$$

We assume that the coefficient function $\mu(\mathbf{x})$ in the bilinear form $\tilde{q}(\cdot,\cdot)$ is constant within each of the prisms forming $\hat{T}^{(0)}$ and can be discontinuous across them.

To construct the preconditioner, we discretize the operators $\mathcal{L}_{xy}$ and $\mathcal{L}_z$ by means of conforming piecewise–linear finite elements and finite–differences, respectively. We denote by $\hat{Q}_{xy}$ and $\hat{Q}_z$ the corresponding stiffness matrices. Further we develop a multilevel solution procedure for the matrix $\hat{Q} = \hat{Q}_{xy} + \hat{Q}_z$. Since the matrix Q_{ii} from the previous section and the matrix $\hat{Q}$ correspond to the different discretizations of the same bilinear form $\tilde{q}(\cdot,\cdot)$, they are spectrally equivalent. Note also that the condition number $\kappa(\hat{Q}^{-1}Q_{ii})$ does not depend on the size of the algebraic system.

Next, consider a sequence of matrices $\{\hat{Q}^{(i)}\}, i = \overline{0,l}$ generated by the operator $\mathcal{L}$ on the sequence of meshes $\{\hat{T}^{(i)}\}$. The matrices $\hat{Q}^{(i)}$ are assumed to be generated in the same way as the matrix $\hat{Q}$. Clearly, $\hat{Q}^{(l)} = \hat{Q}$. We also define a sequence $\{\tilde{Q}^{(i)} = \tilde{Q}^{(i)}_{xy} + \tilde{Q}^{(i)}_{z}\}, i = \overline{0,l}$ of the hierarchical basis (HB) stiffness matrices related to the sequence of meshes $\{\hat{T}^{(i)}\}$; for the description of the hierarchical basis approach we refer to [2], [8], [12] and [13]. Matrices $\tilde{Q}^{(i)}$ and $\hat{Q}^{(i)}$ are related by the following equation:

$$\tilde{Q}^{(i)} = {J^{(i)}}^T \hat{Q}^{(i)} J^{(i)}. \tag{12}$$

Here the matrices $J^{(i)}$ incorporate the couplings between the nodes of $\{\hat{T}^{(i)}\}$, which correspond to the different levels of discretization. An important feature of the matrices $J^{(i)}$ and ${J^{(i)}}^T$ is that both the memory required to store them and the arithmetic effort needed to compute their action on a vector are proportional to their size.

Consider the matrix $\tilde{Q}^{(i)}$. It admits the following 2×2 block partitioning:

$$\tilde{Q}^{(i)} = \begin{bmatrix} \tilde{Q}^{(i)}_{11} & \tilde{Q}^{(i)}_{12} \\ \tilde{Q}^{(i)}_{21} & \tilde{Q}^{(i)}_{22} \end{bmatrix} \begin{matrix} \} & \text{"new" mesh nodes} \\ \} & \text{"old" mesh nodes} \end{matrix}, \quad i = \overline{1,l}. \tag{13}$$

We define a two–level I–AMLI preconditioner $\tilde{M}^{(i)}$ to $\tilde{Q}^{(i)}$ as follows:

$$\tilde{M}^{(i)} = \begin{bmatrix} \tilde{B}^{(i)}_{11} & 0 \\ 0 & \tilde{W}^{(i-1)} \end{bmatrix}, \quad i = \overline{1,l},$$

where the matrices $\tilde{B}^{(i)}_{11}$ and $\tilde{W}^{(i-1)}$ are spectrally equivalent approximations to the matrices $\tilde{Q}^{(i)}_{11}$ and $\tilde{Q}^{(i-1)}$ respectively:

$$\tilde{B}^{(i)}_{11} \sim \tilde{Q}^{(i)}_{11}, \quad \tilde{W}^{(i-1)} \sim \tilde{Q}^{(i-1)}.$$

The construction of the approximations $\tilde{B}^{(i)}_{11}$ to the blocks $\tilde{Q}^{(i)}_{11}$ is described in detail in subsection 4.3. The constructing of the approximations $\tilde{W}^{(i-1)}$ to the matrices $\tilde{Q}^{(i-1)}$ is based on the multilevel approach, as described in [1], [2], [6], [7] and elsewhere. As shown in the above references, the condition number of the AMLI–preconditioned system can be made arbitrarily close to optimal, i.e. bounded independently with respect to the mesh step-size. Furthermore, the cost per I–AMLI iteration step can be made proportional to the number of degrees of freedom on the finest mesh. The only important condition [1], which has to be satisfied for the optimality of the method is the following:

$$\eta_{i_1,i_0} = \frac{1}{2} \frac{\log \sigma_{i_1,i_0}}{\log \rho_{i_1,i_0}} < 1 \tag{14}$$

where i_1 and i_0 denote the numbers of two consecutive regular levels in the I–AMLI multilevel structure, the parameter ρ_{i_1,i_0} denotes the ratio between the dimensions of $\tilde{Q}^{(i_1)}$ and $\tilde{Q}^{(i_0)}$ and, finally, the parameter σ_{i_1,i_0} is defined as:

$$\sigma_{i_1,i_0} = \left[\kappa \left(\left[\tilde{M}^{(i_1)} \right]^{-1} \tilde{Q}^{(i_1)} \right) \right]^{1/(i_1 - i_0)}. \tag{15}$$

In the case of the discretizations used in the present paper the condition (14) is satisfied, see subsection 4.5.

4.3 Approximation of the blocks $\tilde{Q}_{11}^{(i)}$

The splitting of the operator $\mathcal{L}$ in (11) induces the following splitting of the blocks $\tilde{Q}_{11}^{(i)}$:

$$\tilde{Q}_{11}^{(i)} = \tilde{Q}_{11_{xy}}^{(i)} + \tilde{Q}_{11_z}^{(i)},$$

where the blocks $\tilde{Q}_{11_{xy}}^{(i)}$ correspond to the operator $\mathcal{L}_{xy}$ and $\tilde{Q}_{11_z}^{(i)}$ correspond to $\mathcal{L}_z$. Let us also introduce a similar splitting of the approximations $\tilde{B}_{11}^{(i)}$ to the blocks $\tilde{Q}_{11_z}^{(i)}$:

$$\tilde{B}_{11}^{(i)} = \tilde{B}_{11_{xy}}^{(i)} + \tilde{B}_{11_z}^{(i)}.$$

As follows from [2], the condition number $\kappa\left(\tilde{Q}_{11_{xy}}^{(i)}\right)$ of the block $\tilde{Q}_{11_{xy}}^{(i)}$ is of order $O(1)$ as long as the mesh $\hat{T}_{xy}^{(i)}$ does not contain any degenerate triangles. Hence, we can approximate $\tilde{Q}_{11_{xy}}^{(i)}$ by its diagonal part:

$$\tilde{B}_{11_{xy}}^{(i)} = \mathrm{diag}(\tilde{Q}_{11_{xy}}^{(i)}).$$

Next, we define the matrix $\tilde{B}_{11_z}^{(i)}$ simply as

$$\tilde{B}_{11_z}^{(i)} = \tilde{Q}_{11_z}^{(i)}.$$

Clearly,

$$\tilde{Q}_{11}^{(i)} = \tilde{Q}_{11_{xy}}^{(i)} + \tilde{Q}_{11_z}^{(i)} \quad \sim \quad \tilde{B}_{11_{xy}}^{(i)} + \tilde{B}_{11_z}^{(i)} = \tilde{B}_{11}^{(i)}.$$

Since the matrix $\tilde{Q}_{11_z}^{(i)}$ couples the unknowns only in z–direction due to the structure of the operator $\mathcal{L}_z$, there exists a reordering of the unknowns, such that the matrix $\tilde{B}_{11}^{(i)}$ admits a block–diagonal partitioning with three–diagonal blocks. Thus, systems with $\tilde{B}_{11}^{(i)}$ can be easily solved by fast direct methods. Moreover, the inverting of the blocks $\tilde{B}_{11}^{(i)}$ can be efficiently done in parallel.

4.4 On the parallel data distribution

Consider the mesh $\hat{T}_{xy}^{(0)}$ being partitioned into a number of non-overlapping subdomains $\hat{T}_{xy_p}^{(0)}$ (clusters of triangles), distributed between the processing elements in the parallel environment. This implies the corresponding partitioning of $\hat{T}^{(i)}$

$$\hat{T}^{(i)} = \hat{T}_{xy}^{(i)} \otimes \hat{T}_z = \bigcup_{p=1}^{N} \hat{T}_{xy_p}^{(i)} \otimes \hat{T}_z = \bigcup_{p=1}^{N} \hat{T}_p^{(i)}, \quad i = \overline{1,l},$$

where $\hat{T}_p^{(i)}$ are constructed in parallel using uniform refinement procedure applied to each of the subdomains $\hat{T}_p^{(0)}$. Consider the bilinear forms $a(\cdot,\cdot)$ and $q(\cdot,\cdot)$ from (6) and (9) being assembled from their restriction on each of the subdomains $\hat{T}_p^{(0)}$. This leads to the corresponding distribution of the stiffness matrices, involved in the computations. Next, consider two types of the mesh nodes: 1) the nodes, which belong only to one subdomain (interior nodes); 2) the nodes, which are shared by two or more neighboring subdomains (interface nodes). When applying the preconditioner, the unknowns associated with the nodes from the first group can be handled without interprocessor communications. This follows from the block–tridiagonal structure of the blocks $\tilde{B}_{11}^{(i)}$. The multiplications with the matrices $J^{(i)}$ and $J^{(i)^T}$, defined by (12), can be also efficiently parallelized, the communications are required only through the interfaces between the subdomains.

This means that during the preconditioning step local computations dominate over the communications as long as the number of interior nodes is kept sufficiently large compared to the number of interface nodes. This issue is illustrated by the numerical experiments, presented in section 5.

4.5 Computation of the parameter σ

As was mentioned in subsection 4.1, the performance of the I–AMLI method depends strongly on the parameter σ defined by (15). Even though the value of σ is not required within the algorithm itself, it is needed to estimate the convergence rate of the method. Now we estimate σ for the class of problems considered in the present paper.

Consider the generalized eigenvalue problem

$$\begin{bmatrix} \tilde{Q}_{11}^{(i)} & \tilde{Q}_{12}^{(i)} \\ \tilde{Q}_{21}^{(i)} & \tilde{Q}_{22}^{(i)} \end{bmatrix} \begin{bmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \end{bmatrix} = \lambda \begin{bmatrix} \tilde{Q}_{11}^{(i)} & 0 \\ 0 & \tilde{Q}^{(i-1)} \end{bmatrix} \begin{bmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \end{bmatrix}. \tag{16}$$

We are interested in the extremal eigenvalues $\lambda^{min} = \min\{\lambda_k\}$ and $\lambda^{max} = \max\{\lambda_k\}$. Once they have been computed, the value of σ can be estimated as

$$\sigma_{i_1,i_0} \leq \sigma_{i,i-1} \leq \kappa\left(\tilde{B}_{11}^{(i)}, \tilde{Q}_{11}^{(i)}\right) \frac{\lambda^{max}}{\lambda^{min}}.$$

To estimate λ^{min} and λ^{max} we consider the following two auxiliary eigenvalue problems induced by the splitting of the operator $\mathcal{L}$:

$$\begin{bmatrix} \tilde{Q}_{11_{xy}}^{(i)} & \tilde{Q}_{12_{xy}}^{(i)} \\ \tilde{Q}_{21_{xy}}^{(i)} & \tilde{Q}_{22_{xy}}^{(i)} \end{bmatrix} \begin{bmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \end{bmatrix} = \lambda_{xy} \begin{bmatrix} \tilde{Q}_{11_{xy}}^{(i)} & 0 \\ 0 & \tilde{Q}_{xy}^{(i-1)} \end{bmatrix} \begin{bmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \end{bmatrix} \tag{17}$$

and

$$\begin{bmatrix} \tilde{Q}_{11_z}^{(i)} & \tilde{Q}_{12_z}^{(i)} \\ \tilde{Q}_{21_z}^{(i)} & \tilde{Q}_{22_z}^{(i)} \end{bmatrix} \begin{bmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \end{bmatrix} = \lambda_z \begin{bmatrix} \tilde{Q}_{11_z}^{(i)} & 0 \\ 0 & \tilde{Q}_z^{(i-1)} \end{bmatrix} \begin{bmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \end{bmatrix}. \tag{18}$$

After solving (17) and (18) we can estimate λ^{min} and λ^{max} using the following straightforward relations which follow from the Rayleigh quotient relation:

$$\lambda^{min} = \min\{\lambda_{xy}^{min}, \lambda_z^{min}\}, \quad \lambda^{max} = \max\{\lambda_{xy}^{max}, \lambda_z^{max}\}.$$

As shown in [10] (see also [1]),

$$\lambda_{xy}^{min} \geq 1 - \frac{\sqrt{3}}{2}, \quad \lambda_{xy}^{max} \leq 1 + \frac{\sqrt{3}}{2}.$$

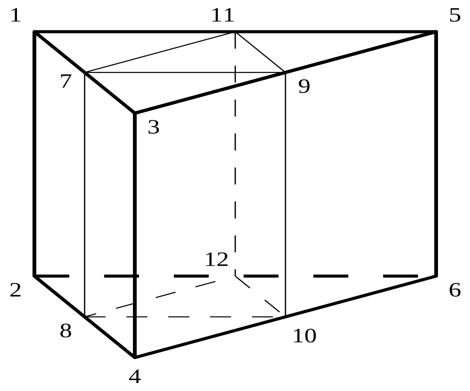

Figure 2: Superelement node numbering

In the present paper we show that the same estimates hold also for λ_z^{min} and λ_z^{max}.

Consider two consecutive levels of mesh refinement $i-1$ and i. They correspond to the meshes $\hat{T}^{(i-1)}$ and $\hat{T}^{(i)}$, such that each prism of the mesh $\hat{T}^{(i-1)}$ contains four congruent prisms of the mesh $\hat{T}^{(i)}$. We refer to the union of these four prisms as a "superelement" of the mesh $\hat{T}^{(i)}$ (see Figure 2). Next, consider the matrices $\tilde{Q}_z^{(i)}$ as a result of assembling the local matrices $\tilde{Q}_z^{(i,p)}$ corresponding to the superelements:

$$\tilde{Q}_z^{(i)} = \sum_{p=1}^{N_i} {L^{(i,p)}}^T \tilde{Q}_z^{(i,p)} L^{(i,p)}.$$

Here N_i denotes a number of superelements on the refinement level i and the matrix $L^{(i,p)}$ corresponds to mappings from the global numbering of the mesh nodes to their local numbering within the superelement p. Elementary computations show that λ_z^{min} and λ_z^{max} can be estimated as

$$\lambda_z^{min} = \min_p \{\lambda_z^{min,p}\}, \quad \lambda_z^{max} = \max_p \{\lambda_z^{max,p}\}.$$

Thus, instead of the "global" problem (18) we can consider the following "local" problem for the superelement p:

$$\begin{bmatrix} \tilde{Q}_{11_z}^{(i,p)} & \tilde{Q}_{12_z}^{(i,p)} \\ \tilde{Q}_{21_z}^{(i,p)} & \tilde{Q}_{22_z}^{(i,p)} \end{bmatrix} \begin{bmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \end{bmatrix} = \lambda_z^p \begin{bmatrix} \tilde{Q}_{11_z}^{(i,p)} & 0 \\ 0 & \tilde{Q}_z^{(i-1,p)} \end{bmatrix} \begin{bmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \end{bmatrix}. \tag{19}$$

All matrices above can be written explicitly:

$$\begin{bmatrix} \tilde{Q}_{11_z}^{(i,p)} & \tilde{Q}_{12_z}^{(i,p)} \\ \tilde{Q}_{21_z}^{(i,p)} & \tilde{Q}_{22_z}^{(i,p)} \end{bmatrix} = J^T A\, J,$$

$$\begin{bmatrix} \tilde{Q}_{11_z}^{(i,p)} & 0 \\ 0 & \tilde{Q}_z^{(i-1,p)} \end{bmatrix} = B,$$

where

$$A = \begin{bmatrix} R & 0 \\ 0 & 3R \end{bmatrix}, \quad B = \begin{bmatrix} 4R & 0 \\ 0 & 3R \end{bmatrix}, \quad J = \begin{bmatrix} I & 0 \\ S & I \end{bmatrix},$$

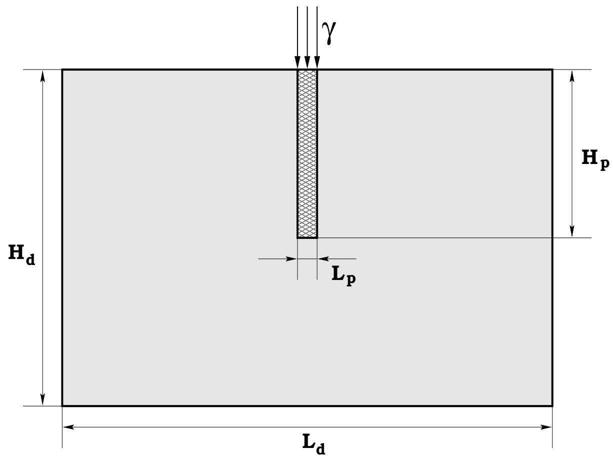

Figure 3: Single–pile problem, vertical cross–section of the domain

while R and S are defined as

$$R = \begin{bmatrix} 1 & -1 & . & . & . & . \\ -1 & 1 & . & . & . & . \\ . & . & 1 & -1 & . & . \\ . & . & -1 & 1 & . & . \\ . & . & . & . & 1 & 1 \\ . & . & . & . & -1 & -1 \end{bmatrix}, \quad S = \begin{bmatrix} 1/2 & . & 1/2 & . & . & . \\ . & 1/2 & . & 1/2 & . & . \\ . & . & 1/2 & . & 1/2 & . \\ . & . & . & 1/2 & . & 1/2 \\ 1/2 & . & . & . & 1/2 & . \\ . & 1/2 & . & . & . & 1/2 \end{bmatrix}.$$

The problem (19) can be solved analytically. The eigenvalues are:

$$\lambda_z^p \in \{1 - \frac{\sqrt{3}}{2},\ \frac{23}{32} - \frac{\sqrt{273}}{32},\ 1,\ \frac{23}{32} + \frac{\sqrt{273}}{32},\ 1 + \frac{\sqrt{3}}{2}\}.$$

Thus,

$$\lambda_z^{min} = 1 - \frac{\sqrt{3}}{2}, \quad \lambda_z^{max} = 1 + \frac{\sqrt{3}}{2}.$$

5 Numerical results

In this section we illustrate the performance of the developed solver on a number of model problems arising from bridge foundation modeling. Namely, we consider a single–pile problem and a square footing problem. We show scalability of both the I–AMLI method alone and the whole linear elasticity solver with the I–AMLI algorithm applied for the blocks Q_{ii} (see section 3).

First, consider a single concrete pile placed in a homogeneous soil layer and subjected by external surface forces $\underline{\gamma}$ and volume forces $\mathbf{g}$. In [9] this problem is referred to as the MOST problem. We assume that the construction technology guarantees the ideal contact between the pile and the soil, i.e. there is no internal friction. Next, consider a homogeneous soil layer (without the pile) subjected by the same surface and volume forces as in the previous example. We refer to this problem as to the square footing problem.

In our computations the volume force $\mathbf{g}$ is taken in the form $(0, -g_v)$. It represents the gravity force acting on the continuous media. A vertical external load $\underline{\gamma} = (0, -\gamma_v)$ is applied to the part of the top surface as shown in Figure 3. It is the same for both the MOST and the square footing problems. We assume that the load is uniformly distributed over the loading surface. The following boundary conditions are chosen in the experiments: homogeneous Dirichlet b.c. on the bottom side of the domain (displacements are equal zero), inhomogeneous Neumann b.c. on the loading surface and homogeneous Neumann b.c. on the rest of the boundary (stresses are equal zero). The Young modulus E and the Poisson ratio ν are defined as

$$E = \begin{cases} E_p & \text{for the pile (concrete),} \\ E_s & \text{for the soil (sandy clay),} \end{cases}$$

$$\nu = \begin{cases} \nu_p & \text{for the pile (concrete),} \\ \nu_s & \text{for the soil (sandy clay).} \end{cases}$$

The values of $L_d, H_d, L_p, H_p, E_p, E_s, \nu_p$ and ν_s are presented in Table 1.

The single–pile problem is discretized using a sequence of meshes, locally pre–refined in the area where the solution is known to be strongly varying, see Figure 1. A non–uniform grid $\hat{T}_z$ with 35 nodes, refined around the bottom of the pile, is used to construct a sequence of meshes $\hat{T}^{(i)}$. The square footing problem is discretized on a product of a uniform grid $\hat{T}_z$ with 32 nodes and a sequence of regular Cartesian meshes $\hat{T}_{xy}$. A number of meshes $\hat{T}_{xy}$ (with 17, 33, 65 and 129 points in each direction) is tested to show the performance of the algorithm with respect to the size of the algebraic system. The discretization is done by means of conforming piecewise–linear basis finite elements associated with the vertices of $\hat{T}^{(l)}$.

Computing time, speedup and a number of outer iterations for the square footing and MOST problems are given in Tables 2 and 3. The following stopping criterion is used for the outer iterative procedure with A:

$$\frac{\|\mathbf{r}_k\|_2}{\|\mathbf{r}_0\|_2} < 10^{-4}.$$

Here $\mathbf{r}_0$ denotes the initial residual and $\mathbf{r}_k$ is the residual after the k-th iteration. The blocks Q_{ii} are solved using the I–AMLI preconditioning. The "inner" stopping criterion is

$$\frac{\|\mathbf{r}_k\|_2}{\|\mathbf{r}_0\|_2} < 10^{-1}.$$

The block–Jacobi preconditioner (with the blocks grouped along the z–direction) is applied on the coarsest mesh $\hat{T}^{(0)}$.

Since most of the computations is devoted to solving the inner systems with Q_{ii}, we illustrate the performance of the inner multilevel solver alone. The time to solve the block Q_{33} with accuracy

$$\frac{\|\mathbf{r}_k\|_2}{\|\mathbf{r}_0\|_2} < 10^{-6}$$

is presented in Tables 4 and 5. The I–AMLI method is also compared with the block–Jacobi–preconditioned CG method applied to the same algebraic system. As we see from the above tables, the developed preconditioner achieves very good scalability with respect to both the number of degrees of freedom in the algebraic system and the number of processing elements.

L_d	H_d	L_p	H_p	E_p	E_s	ν_p	ν_s
37.2 m	31.0 m	1.2 m	15.0 m	$3.15 \cdot 10^{10}$ Pa	$1.0 \cdot 10^{7}$ Pa	0.2	0.3

Table 1: Problem parameters

Problem size	Outer iterations	Processing elements	Sun ES/4000		Cray T3E-600	
			Time	Speedup	Time	Speedup
65382	10	1	106.17	1.0	118.16	1.0
		2	63.96	1.7	66.70	1.8
		4	41.57	2.6	38.89	3.0
		8	—	—	25.19	4.7
257958	10	1	709.38	1.0	—	—
		2	380.18	1.9	—	—
		4	209.60	3.4	214.92	*3.8*
		8	—	—	122.73	*6.7*

Table 2: Single–pile problem, performance of the linear elasticity solver

Problem size	Outer iterations	Processing elements	Sun ES/4000		Cray T3E-600	
			Time	Speedup	Time	Speedup
101277	8	1	74.19	1.0	—	—
		2	40.22	1.8	42.42	*1.9*
		4	22.83	3.2	22.30	*3.6*
		8	—	—	12.77	*6.3*
		16	—	—	7.37	*10.9*
		32	—	—	4.45	*18.1*
		64	—	—	2.87	*28.1*
392925	8	1	368.83	1.0	—	—
		2	190.27	1.9	—	—
		4	99.78	3.7	—	—
		8	—	—	52.72	*6.9*
		16	—	—	27.74	*13.1*
		32	—	—	15.15	*24.0*
		64	—	—	8.48	*42.9*
1547613	8	1	—	—	—	—
		2	—	—	—	—
		4	—	—	—	—
		8	—	—	—	—
		16	—	—	—	—
		32	—	—	60.85	*27.2*
		64	—	—	31.05	*53.3*

Table 3: Square footing problem, performance of the linear elasticity solver

		I–AMLI/GCG				Block–Jacobi/PCG			
Problem size	Processing elements	Sun ES/4000		Cray T3E-600		Sun ES/4000		Cray T3E-600	
		Time	Speedup	Time	Speedup	Time	Speedup	Time	Speedup
21794	1	14.46	1.0	16.92	1.0	25.63	1.0	31.14	1.0
	2	9.15	1.6	9.87	1.7	15.36	1.7	17.19	1.8
	4	6.11	2.4	5.82	2.9	9.44	2.7	9.81	3.2
	8	—	—	3.91	4.3	—	—	6.16	5.1
85986	1	96.33	1.0	105.34	1.0	226.07	1.0	246.40	1.0
	2	52.97	1.8	56.68	1.9	120.92	1.9	129.29	1.9
	4	29.56	3.3	31.19	3.4	65.17	3.5	70.72	3.5
	8	—	—	18.20	5.8	—	—	40.01	6.2
341666	1	477.01	1.0	—	—	1786.94	1.0	—	—
	2	249.19	1.9	272.18	*1.9*	939.49	1.9	—	—
	4	137.21	3.5	137.56	*3.8*	506.64	3.5	—	—
	8	—	—	74.78	*7.1*	—	—	—	—

Table 4: Single–pile problem, performance of the inner I–AMLI solver

		I–AMLI/GCG				Block–Jacobi/PCG			
Problem size	Processing elements	Sun ES/4000		Cray T3E-600		Sun ES/4000		Cray T3E-600	
		Time	Speedup	Time	Speedup	Time	Speedup	Time	Speedup
33759	1	10.64	1.0	12.03	1.0	17.11	1.0	20.09	1.0
	2	5.96	1.8	6.44	1.9	9.56	1.8	10.71	1.9
	4	3.32	3.2	3.49	3.4	5.14	3.3	5.67	3.5
	8	—	—	2.03	5.9	—	—	3.20	6.3
	16	—	—	1.23	9.8	—	—	1.86	10.8
	32	—	—	0.79	15.2	—	—	—	—
	64	—	—	0.56	21.5	—	—	—	—
130975	1	60.02	1.0	64.78	1.0	147.04	1.0	159.65	1.0
	2	31.31	1.9	33.01	2.0	74.18	2.0	80.78	2.0
	4	16.66	3.6	17.14	3.8	38.56	3.8	41.66	3.8
	8	—	—	9.36	6.9	—	—	22.47	7.1
	16	—	—	5.02	12.9	—	—	11.85	13.5
	32	—	—	2.81	23.1	—	—	6.40	24.9
	64	—	—	1.63	39.7	—	—	3.57	44.7
515871	1	305.39	1.0	—	—	1067.92	1.0	—	—
	2	159.09	1.9	167.89	*2.0*	567.43	1.9	—	—
	4	85.40	3.6	83.96	*3.9*	296.98	3.6	—	—
	8	—	—	43.61	*7.6*	—	—	—	—
	16	—	—	22.86	*14.5*	—	—	—	—
	32	—	—	12.15	*27.2*	—	—	—	—
	64	—	—	6.44	*51.4*	—	—	—	—

Table 5: Square footing problem, performance of the inner I–AMLI solver

Even for relatively small–sized problems it outperform the block–diagonal preconditioner, which is known to be serious competitor in the parallel environment.

The computations are performed on Cray-T3E (Alpha 21164, 300 Mhz) and on Sun ES/4000 system (UltraSparc, 167 Mhz) platforms using portable message passing library (MPI). No special tuning for a particular computer platform has been performed. The timings are given in seconds of the wall-clock time.

6 Concluding remarks

An efficient multilevel preconditioner has been developed for the algebraic systems arising from finite–element discretization of linear elasticity problems. The algorithm exhibits good parallelization properties and shows high level of scalability on modern multiprocessor systems. The rate of convergence of the preconditioned system is independent of the size of the algebraic system and is not sensitive to the jumps in the problem coefficients. The method can work on non–uniform discretization meshes, which are needed to achieve acceptable discretization error while using only a few successive mesh refinement steps. This allows to discretize the problem efficiently and achieve the needed accuracy of the solution using a moderate number of degrees of freedom in the algebraic system.

References

[1] O. Axelsson, *Stabilization of AMLI methods; additive methods*, in Proc. Conf. on Algebraic Multilevel Iteration Methods with Applications, O. Axelsson and B. Polman, eds., University of Nijmegen, The Netherlands, 1996, pp. 49–62.

[2] O. Axelsson and I. Gustafson, *Preconditioning and two–level multigrid methods of arbitrary degree of approximation*, Math. Comp., 40 (1983), pp. 219–242.

[3] O. Axelsson and A. Padiy, *On the additive version of the algebraic multilevel iteration method for anisotropic elliptic problems*, Technical report No 9711, University of Nijmegen, 1997, to appear in SIAM J. Sci. Comput.

[4] O. Axelsson and A. Padiy, *On a robust and scalable linear elasticity solver based on a saddle point formulation*, Technical report No 9723, University of Nijmegen, 1997, submitted.

[5] O. Axelsson and P. Vassilevski, *A black–box generalized conjugate gradient solver with inner iterations and variable–step preconditioning*, SIAM J. Matrix Anal. Appl., 12 (1991), pp. 625–644.

[6] O. Axelsson and P. Vassilevski, *Algebraic multilevel preconditioning methods, part I*, Numer. Math., 56 (1989), pp. 157–177.

[7] O. Axelsson and P. Vassilevski, *Algebraic multilevel preconditioning methods, part II*, SIAM J. Numer. Anal., 27 (1990), pp. 1569–1590.

[8] R. Bank, T. Dupont and H. Yserentant, *The hierarchical basis multigrid method*, Numer. Math., 52 (1988), pp. 427–458.

[9] A. Georgiev, A. Baltov and S. Margenov, HIPERGEOS benchmark problems related to bridge engineering applications, project report COP-94-0820-MOST-4.

[10] J. F. Maitre and F. Musy, *The contraction number of a class of two–level methods; an exact evaluation for some finite element subspaces and model problems*, Lecture Notes in Math. 960, Springer, Berlin, 1982, pp. 535–544.

[11] S. Mikhlin, *The problem of the minimum of a quadratic functional*, Holden–Day Inc., San Francisco, 1965.

[12] P. Vassilevski, *On two ways of stabilizing the hierarchical basis multilevel methods*, SIAM Rev., 39 (1997), pp. 18–53.

[13] H. Yserentant, *On the multilevel splitting of finite element spaces*, Numer. Math., 49 (1986), pp. 379–412.