PROJECT REPORT, COPERNICUS CO94–00820

LOCAL REFINEMENT SOLUTION OF $3D$ ELASTICITY EQUATIONS

PANAYOT S. VASSILEVSKI AND LUDMIL T. ZIKATANOV

Abstract. In this report we describe the principal steps in the design of the finite element solution of elliptic problems on grids with regular patched local refinement. Also a basic iterative procedure for solving the composite grid problem, utilizing the regular structure of the matrix blocks corresponding to both the coarse and fine grids (i.e. the patch) is presented. Some preliminary numerical results for 3D elasticity equations results are also shown. A brief summary of the fortran modulus that implement both the discretization and solution methods is included.

1. Problem formulation and structure of the report

In this section we present the elasticity system of main interest to our project "Copernicus" COP–94 00820. Then the finite element discretization of the problem on two grids (or based on two finite element spaces) is presented. This induces respective block structure of DD (domain decomposition) type to the composite finite element stiffness matrix.

The finite element discretization is presented in Section 2. The handling of interface communication when introducing local refinement is given in subsection 2.2. This is of main interest when implementing the action of the composite grid stiffness matrix on vectors, a principal step of any iterative method method for solving the resulting system of linear equations. The preconditioner which utilizes solutions on regular grids (the global coarse grid and the fine grid in the subdomain where the local refinement is present) is outlined in Section 3. This is the so–called BEPS (or symmetrized FAC) preconditioner, cf. [1] and [6]. Section 4 contains some preliminary tests of our implementation of the finite element discretization and preconditioning solution strategies on typical examples of 3–D elasticity problem motivated by the benchmark problems as formulated in the report [4]. Finally a brief description of the developed software modulus is included in Section 5. A flow chart of a typical interconnection of various software modulus for solving our test problems is also given.

1.1. Problem formulation.

We here state the elasticity equations, the problem domain, respective boundary conditions and the finite element discretization in a variational form.

The stressed strained state is described as coupled system of partial differential equations as follows:

Date: March 18, 1995.

$$D\underline{\sigma} + \underline{p} = 0 \qquad \forall (x_1, x_2, x_3) \in \Omega \subset \mathbb{R}^3. \tag{1.1}$$

Considering the stresses σ as the derivatives of the displacements the above system of equations is a second order elliptic system for the displacements $\underline{u}$ in each direction. The boundary conditions which deal with practical implementation of such a mathematical model are of mixed type. For our typical applications, commonly homogeneous Dirichlet boundary condition is imposed on the bottom plane of the domain. On the other boundaries the normal components of the stresses are given (i.e., Neumann boundary conditions).

The strains are determined by $\underline{\varepsilon} = D^T \underline{u}$, where

$$D = \begin{pmatrix} \frac{\partial}{\partial x_1} & 0 & 0 & \frac{\partial}{\partial x_2} & 0 & \frac{\partial}{\partial x_3} \\ 0 & \frac{\partial}{\partial x_2} & 0 & \frac{\partial}{\partial x_1} & \frac{\partial}{\partial x_3} & 0 \\ 0 & 0 & \frac{\partial}{\partial x_3} & 0 & \frac{\partial}{\partial x_2} & \frac{\partial}{\partial x_1} \end{pmatrix}$$

Then the stresses σ_{ij} are obtained from the strains by the following identities:

$$\begin{pmatrix} \sigma_{11} \\ \sigma_{22} \\ \sigma_{33} \\ \sigma_{12} \\ \sigma_{23} \\ \sigma_{31} \end{pmatrix} = E^\star \begin{pmatrix} (1-\nu) & \nu & \nu & & & 0 \\ \nu & (1-\nu) & \nu & & & \\ \nu & \nu & (1-\nu) & & & \\ & & & \frac{1-2\nu}{2} & & \\ & & & & \frac{1-2\nu}{2} & \\ 0 & & & & & \frac{1-2\nu}{2} \end{pmatrix} \begin{pmatrix} \epsilon_{11} \\ \epsilon_{22} \\ \epsilon_{33} \\ 2\epsilon_{12} \\ 2\epsilon_{23} \\ 2\epsilon_{31} \end{pmatrix} \tag{1.2}$$

where $E^* = \frac{E}{(1+\nu)(1-2\nu)}$, ν is the Poisson ratio and E is the deformation module.

The bilinear form corresponding to the problem (1.1–1.2) takes the following form:

$$\begin{aligned} a(\underline{u},\underline{w}) &= \int_\Omega (1-\nu) E^\star \frac{\partial u_1}{\partial x}\frac{\partial w_1}{\partial x} + \int_\Omega \frac{1-2\nu}{2} E^\star \frac{\partial u_2}{\partial x}\frac{\partial w_2}{\partial x} + \int_\Omega \frac{1-2\nu}{2} E^\star \frac{\partial u_3}{\partial x}\frac{\partial w_3}{\partial x} \\ &= \int_\Omega \frac{1-2\nu}{2} E^\star \frac{\partial u_1}{\partial y}\frac{\partial w_1}{\partial y} + \int_\Omega (1-\nu) E^\star \frac{\partial u_2}{\partial y}\frac{\partial w_2}{\partial y} + \int_\Omega \frac{1-2\nu}{2} E^\star \frac{\partial u_3}{\partial y}\frac{\partial w_3}{\partial y} \\ &= \int_\Omega \frac{1-2\nu}{2} E^\star \frac{\partial u_1}{\partial z}\frac{\partial w_1}{\partial z} + \int_\Omega \frac{1-2\nu}{2} E^\star \frac{\partial u_2}{\partial z}\frac{\partial w_2}{\partial z} + \int_\Omega (1-\nu) E^\star \frac{\partial u_3}{\partial z}\frac{\partial w_3}{\partial z} \\ &+ \int_\Omega \nu E^\star \left[\frac{\partial u_1}{\partial x}\frac{\partial w_2}{\partial y} + \frac{\partial u_2}{\partial y}\frac{\partial w_1}{\partial x}\right] \\ &+ \int_\Omega \frac{1-2\nu}{2} E^\star \left[\frac{\partial u_1}{\partial y}\frac{\partial w_2}{\partial x} + \frac{\partial u_2}{\partial x}\frac{\partial w_1}{\partial y}\right] \\ &+ \int_\Omega \nu E^\star \left[\frac{\partial u_1}{\partial x}\frac{\partial w_3}{\partial z} + \frac{\partial u_3}{\partial z}\frac{\partial w_1}{\partial x}\right] \\ &+ \int_\Omega \frac{1-2\nu}{2} E^\star \left[\frac{\partial u_1}{\partial z}\frac{\partial w_3}{\partial x} + \frac{\partial u_3}{\partial z}\frac{\partial w_1}{\partial z}\right] \\ &+ \int_\Omega \nu E^\star \left[\frac{\partial u_2}{\partial y}\frac{\partial w_3}{\partial z} + \frac{\partial u_3}{\partial z}\frac{\partial w_2}{\partial y}\right] \\ &+ \int_\Omega \frac{1-2\nu}{2} E^\star \left[\frac{\partial u_2}{\partial z}\frac{\partial w_3}{\partial y} + \frac{\partial u_2}{\partial y}\frac{\partial w_3}{\partial z}\right], \end{aligned} \tag{1.3}$$

where $\underline{u} = \begin{bmatrix} u_1 \\ u_2 \\ u_2 \end{bmatrix}$ and $\underline{w} = \begin{bmatrix} w_1 \\ w_2 \\ w_2 \end{bmatrix}$ satisfy homogeneous Dirichlet boundary conditions at corresponding Dirichlet boundary piece of $\partial\Omega$.

2. Local refinement finite element discretization

In this section we describe the principal steps of discretizing problem corresponding to the bilinear form (1.3) on grids with local refinement.

Let V_0 be an initial coarse finite element space associated with a regular coarse triangulation $\mathcal{T}_H$ of the domain Ω. Let V_1^0 be a local fine finite element space, that is, if $\varphi \in V_1^0$ then supp $\varphi \subset \overline{\Omega}_1$. Here, Ω_1 is the subdomain of the original domain where we have introduced local refinement. Then one can define the composite finite element discretization space as $V = V_1^0 + V_0$. Then the discrete problem is formulated in the same way as in the case of standard finite element space using now V as a discretization space. There are, however, some details of how one can actually compute the corresponding stiffness matrix and this reflects also the data structure of the matrix entries and the unknown function (or vectors). This is described now in more details.

2.1. **Element stiffness matrix computation and assembly of the matrix blocks.** We first compute the stiffness matrix $\widetilde{A}$ corresponding to the global coarse space V_0 is the usual way. The element stiffness matrices corresponding to each fine element in $\mathcal{T}_h^1$, the fine–grid triangulation introduced in $\overline{\Omega}_1$, are also computed in the usual way. Each such element stiffness matrix is stored in our implementation based on tri–linear elements on parallelepiped elements in $(3, 3, 8, 8)$ arrays using Fortran convention. The first $(3, 3)$ block stands for the number of degrees of freedom at every vertex of the given element and it equals three since we have three components of the unknown function and because we have three equations for that associated with each component. The block $(8, 8)$ comes from the number of vertices of the parallelepiped element which gives the number of basis functions associated with the element. in total we have 24×24 matrix which equals to the total degrees of freedom associated with a single element.

In $\overline{\Omega}_1$ we assemble the subdomain stiffness matrix, including the degrees of freedom associated with $\partial\Omega_1$. This corresponds to Neumann type stiffness matrix. Note that the degrees of freedom associated with the interface boundary $\Gamma \equiv \partial\Omega_1 \setminus \Gamma_D$ are also included in that subdomain stiffness matrix. This stiffness matrix will be used while implementing the action of the composite stiffness matrix on vectors.

2.2. **Computing actions composite–matrix times vectors.** To implement the action of the global composite–grid stiffness matrix on certain vectors we utilize the following block structure of the vectors. Each composite–grid vector $\mathbf{v}$ is represented as a pair of vectors $\begin{bmatrix} \mathbf{v}_1 \\ \mathbf{v}_0 \end{bmatrix}$ where $\mathbf{v}_1$ correspons to the degrees of freedom in $\overline{\Omega}_1$ and $\mathbf{v}_0$ corresponds to the degrees of the freedom on the global coarse mesh, however, $\mathbf{v}_0$ is kept zero at the degrees of freedom associated with the coarse–grid nodes which are inside Ω_1. Also, on the interface $\Gamma(= \partial\Omega_1 \setminus \Gamma_D)$, $\mathbf{v}_1$ is defined for all degrees of freedom associated with both coarse and fine grid nodes. However, the fine grid nodes on Γ which are not present in the coarse grid (the so–called slave nodes), the values of $\mathbf{v}_1$ are obtained by interpolation. This interpolation comes from the natural imbedding V_H restricted to $\overline{\Omega}_1$ in V_1, the fine finite element space defined in $\overline{\Omega}_1$. In our particular application of trilinear functions the interpolation on Γ is one bilinear one; i.e., the value of each component of $\mathbf{v}_1$ (these are three) at a given "slave" node on Γ is obtainde by bilinear interpolation using four neighboring coarse–grid points on Γ which are the actual (true) degrees of freedom on Γ.

Then, the action of the composit grid matrix A on vectors $\mathbf{v}$ is defined in Ω in the following ways:

$$A\mathbf{v}_1|_{\Omega\setminus\overline{\Omega}_1} = (\widetilde{A}\mathbf{v}_0)\Big|_{\Omega\setminus\overline{\Omega}_1},$$

$$A\mathbf{v}_1|_{\Omega_1} = (A_1^N\mathbf{v}_1)\big|_{\Omega_1},$$

and finally on Γ,

$$A\mathbf{v}_1|_{\Gamma} = (A_1^N\mathbf{v}_1)\big|_{\Gamma} + (\widetilde{A}_0^N\mathbf{v}_0)\Big|_{\Gamma}.$$

We note, that we need the actions of the coarse–grid stiffness matrix $\widetilde{A}^N$ corresponding to the Neumann problem in $\Omega_2 \equiv \Omega \setminus \overline{\Omega}_1$, which we do not have to assemble explicitly. This is implemented in a separate subroutine and is based on element by element matrix vector products in Ω_2. Note that those are coarse–grid elements contained in $\overline{\Omega}_2$.

For further purposes (to define the preconditioner) we now define the following special matrix vector products:

$$A_{11}\mathbf{v}_1^0 = \left(A\begin{bmatrix}\mathbf{v}_1^0\\ 0\end{bmatrix}\right)\Bigg|_{\Omega_1}. \tag{2.4}$$

This is actually the action of the subdomain stiffness matrix block A_{11} (zero Dirichlet boundary conditions on Γ are imposed).

$$A_{12}\mathbf{v}_2 = \left(A\begin{bmatrix}0\\ \mathbf{v}_2\end{bmatrix}\right)\Bigg|_{\Omega_1}. \tag{2.5}$$

Here we chose the composite vector be zero in the subdomain Ω_1 (but not on Γ). Finally we define:

$$A_{21}\mathbf{v}_1 = \left(A\begin{bmatrix}\mathbf{v}_1\\ 0\end{bmatrix}\right)\Bigg|_{\Omega_2\cup\Gamma}. \tag{2.6}$$

Here the composite vector is possibly non–zero only in Ω_1 and is zero on Γ and in Ω_2.

Having the above actions defined, we can now actually split the actual (true) degrees of freedom in a DD (domain decomposition) fashion. Given $\mathbf{v}$ we have $\mathbf{v}_1$ the corresponding to the degrees of freedom in Ω_1 and $\mathbf{v}_2$ corresponds to the degrees of freedom on Γ and Ω_2 which actually corresponds to the coarse–grid nodes.

Then the composite grid stiffness matrix admits the following two–by–two block–structure:

$$A = \begin{bmatrix}A_{11} & A_{12}\\ A_{21} & A_{22}\end{bmatrix} \tag{2.7}$$

Note that out of these four blocks we have direct access only to A_{11} which a subblock of A_1^N. The remaining three blocks are available only their actions defined by (2.6), (2.5) for A_{22} by

$$A_{22}\mathbf{v}_1 = \left(A \begin{bmatrix} 0 \\ \mathbf{v}_2 \end{bmatrix} \right)\Bigg|_{\Omega_2 \cup \Gamma} .$$

The latter action will not be needed in the construction of the preconditioner.

3. BEPS PRECONDITIONER

Consider the block structure of A in (2.7) We introduce the nodal degrees of freedom:

- $\mathcal{N}_H$ – corresponding to the coarse–grid nodes in Ω;
- $\mathcal{N}_h^1$ – corresponding to the fine grid nodes in Ω_1.
- $\mathcal{N}_H^2 = \mathcal{N}_H \cap (\Omega_2 \cup \Gamma)$ – corresponding to the set of nodes outside Ω_1 which are coarse–grid nodes only.

The preconditioner of main interest is defined then as follows:

$$B = \begin{bmatrix} A_{11} & 0 \\ A_{21} & \widetilde{S} \end{bmatrix} \begin{bmatrix} I & A_{11}^{-1}A_{12} \\ 0 & I \end{bmatrix} \begin{matrix} \}\mathcal{N}_h^1 \\ \}\mathcal{N}_H^2 = \mathcal{N}_H \cap (\Omega_2 \cup \Gamma) \end{matrix} . \tag{3.8}$$

Here we have used the following Schur complement of the (global) coarse–grid stiffness matrix partitioned in the corresponding DD block–form:

$$\widetilde{A} = \begin{bmatrix} \widetilde{A}_{11} & \widetilde{A}_{12} \\ \widetilde{A}_{21} & \widetilde{A}_{22} \end{bmatrix} \begin{matrix} \}\mathcal{N}_H \cap \Omega_1 \\ \}\mathcal{N}_H^2 = \mathcal{N}_H \cap (\Omega_2 \cup \Gamma) \end{matrix}$$

Then

$$\widetilde{S} = \widetilde{A}_{22} - \widetilde{A}_{21} \left(\widetilde{A}_{11} \right)^{-1} \widetilde{A}_{12} .$$

Note that the following representation holds for the inverse of $\widetilde{S}$; namely,

$$\widetilde{A}^{-1} = \begin{bmatrix} \star & \star \\ \star & \widetilde{S}^{-1} \end{bmatrix} . \tag{3.9}$$

This shows that to implement the preconditioner (3.8) one can use the following algorithm:

ALGORITHM 3.1. *(*BEPS ALGORITHM*) Given the problem*

$$A\mathbf{y} = \mathbf{b} = \begin{bmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \end{bmatrix} .$$

Then the solution $\mathbf{y} = B^{-1}\mathbf{b}$ *is computed in the following steps:*

- *solution in* Ω_1 *(fine-grid problem), i.e., solve:*

$$A_{11}\mathbf{y}_1^F = \mathbf{b}_1;$$

- *computation of the residual*

$$\mathbf{d} = \mathbf{b} - A \begin{bmatrix} \mathbf{y}_1^F \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ \mathbf{b}_2 - A_{21}\mathbf{y}_1^F \end{bmatrix} \begin{matrix} \}\Omega_1 \\ \}\Omega_2 \cup \Gamma \end{matrix} ;$$

- *coarse–grid correction (solution of the global coarse-grid problem)*

$$\widetilde{A}\widetilde{\mathbf{y}} = \begin{bmatrix} 0 \\ \mathbf{b}_2 - A_{21}\mathbf{y}_1^F \end{bmatrix} \begin{matrix} \}\mathcal{N}_H \cap \Omega_1 \subset \mathcal{N}_h^1 \\ \}\mathcal{N}_H \cap (\Omega_2 \cup \Gamma) \end{matrix} ;$$

- *finding the harmonic component* $\mathbf{y}_1^H$ *in* Ω_1*, i.e., compute* $\mathbf{y}_1^H$ *such that* $A_{11}\mathbf{y}_1^H + A_{12}\widetilde{\mathbf{y}}_2 = 0$*, i.e. compute the harmonic extension*

$$\mathbf{y}_1^H = -A_{11}^{-1}A_{12}\widetilde{\mathbf{y}}_2.$$

- *Then*

$$\mathbf{y} \equiv B^{-1}\mathbf{b} = \begin{bmatrix} \mathbf{y}_1^F + \mathbf{y}_1^H \\ \widetilde{\mathbf{y}}_2 \end{bmatrix}.$$

It is clear that Algorithm 3.1 requires the actions of the matrix blocks A_{21}, A_{12} which are available through the formulas (2.6) and (2.5). The other major steps involve solutions with A_{11} and $\widetilde{S}$. The latter is based on the representation (3.9) and hence requires solutions with the coarse–grid stiffness matrix $\widetilde{A}$. Therefore both solutions (with A_{11} and $\widetilde{A}$) are performed on regular (in our case, standard parallelepiped) grids. This was the main point in the development of the BEPS preconditioners in [1] and later its study in [6]. The case of inexact solutions with A_{11} on the subdomain Ω_1 was numerically studied in Ewing and Vassilevski [7] and later in Petrova [8] and [9].

4. Model test examples

We here formulate our preliminary test results on 3D–elasticity problems discretized on patched local refined grids. At the present we have only two–level patched grids.

A Cross–section of the domain Ω with the actual coarse grid is shown in Figure 1 (on the left). This means that the grid is (nx_c, ny_c, nz_c), where $nx_c = ny_c = 10$ and $nz_c = 15$ are the dimension of the grid nodes in the respective coordinate direction. On the right of Fig. 1, the locally refined grid in the actual domain Ω_1 is shown. That is, Ω_1 has left lower vertex with integer coordinates $(4, 4, 6)$ and the top vertex (on the main diagonal of the parallelepiped Ω_1) has integer coordinates $(7, 7, 15)$. In the test we have created several grids with local refinement by refining each coarse parallelepiped in $\overline{\Omega}_1$ into $l_{nx} \times l_{ny} \times l_{nz}$ finer elements. The triples $l_{nx} \times l_{ny} \times l_{nz}$ are shown in the Table below. The stiffness matrix used in this model example is obtained as in [5]. A single pile in homogeneous sandy clay soil is considered. The mechanical characteristics are as follows:

- (a) Pile: specific volume weight $\gamma = 25kN/m^3$; $E = 31500MPa$; $\nu = 0.2$;
- (b) $E = 10MPa$; $\nu = 0.3$;

It can be seen from the above description that the coefficients of the equations (1.1) have a large jumps across the pile interface.

For the solution we employed the preconditioned CG (conjugate gradient) method using the BEPS preconditioner as described in Section 3, and implemented as in Algorithm 3.1. The actions of the composite matrix A on vectors was implemented as described in Section 2.2. The data storage of composite vectors and matrix blocks is also described there, that is, it utilizes the data structure which has been implemented on regular (in our case parallelepiped) grids.

FIGURE 1. Coarse grid and pile position; and locally refined grid.

The table shows the number of PCG iterations for three different refinements.

Total number of unknowns	Refinement $l_{nx} \times l_{ny} \times l_{nz}$	Number of PCG iterations	Average reduction factors
8238	$2 \times 2 \times 2$	21	0.237
13845	$3 \times 3 \times 3$	39	0.444
33330	$4 \times 4 \times 6$	46	0.499

TABLE: Test of BEPS preconditioned CG method for solving 3D elasticity equations on two–level patched locally refined grids

5. Brief outline of the developed software modulus

In this section we briefly describe the main building blocks for the implementations of the finite element solution of our 3D elasticity equations in a parallelepiped domain Ω discretized on two–level patched local refinement grids.

- computing the element stiffness matrices using tri–linear basis functions for the 3D–elasticity system; quadrature used to compute the corresponding integrals, and the resulting element matrix block has the form (3,3,8,8) using Fortran convection. The routine which deals with the forming of the local element stiffness and mass matrices is *LOCSTM*. This routine can form the corresponding stiffness matrices for iso–parametric finite elements. The quadrature rules which are employed are $3, 4, 5$ Gaussian quadratures. In a typical situation the tensor product of $3 - point$ Gaussian quadrature rule is used, because the coefficients are piecewise constants.
- Stiffness matrix assembly on regular parallelepiped mesh; The FORTRAN routine is named *ASSEMB*. This routine uses the block form of the global stiffness matrix, and worked on general meshes which are topologically equivalent to the rectangular ones. The matrix blocks of

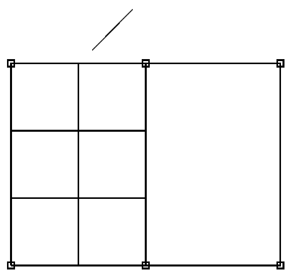

FIGURE 2. Slave and coarse nodes on the cross-section $y = const$ on the interface Γ

the corresponding stiffness matrix (here denoted by A) have the form:

$$AL(3, 3, ny, -1 : 1, nx, nz)$$
$$AD(3, 3, ny, -1 : 1, nx, nz)$$
$$AU(3, 3, ny, -1 : 1, nx, nz)$$

Here, nz is the number of plane blocks along the z–coordinate direction in the parallelepiped mesh used. nx is the number of line blocks along y–coordinate direction within each plane block and ny is the number of nodes in y–coordinate directions at each line block. AL stands for the lower triangular part of A, AU stands for the upper triangular part of A, and AD is the block–diagonal part of A. Note that each these blocks admit block tridiagonal structure within each z–plane using as blocks grid lines oriented along y–direction. This is because we have 27–point stencil of A for tri–linear basis functions on our regular (parallelepiped) grid. Finally we mention that we are exploiting band storage of these tridiagonal matrices where for example $(., ., j, k, ., .)$ gives the $(j + k, j)$ entry of the given tri–diagonal matrix; $k = -1, 0, 1$ and $1 \leq j, j + k \leq ny$.

This module is used to assemble both the global coarse matrix $\widetilde{A}$ and the stiffness matrix in Ω_1 (with Neumann boundary conditions on the interface boundary Γ).

- Subroutine *PRODAL* evaluates the composite grid stiffness matrix times composite vector. It uses tri–linear interpolation at the slave nodes based on four neighboring coarse grid nodes (see Figure 2) for a typical illustration. The interpolation is executed in the routine *EX-PND*. The action of the composite–grid matrix A on composite vectors which are stored as pair of vectors $\begin{bmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \end{bmatrix}$, where $\mathbf{v}_1$ is defined on the fine rectangular mesh in the sub–parallelepiped Ω_1 and $\mathbf{v}_2$ is defined on the global coarse mesh in Ω but is kept zero at the coarse nodes in

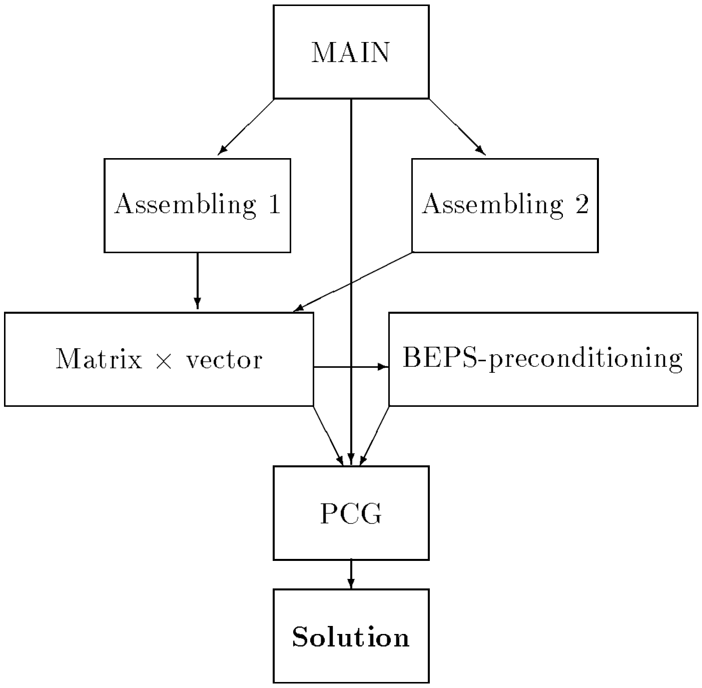

FIGURE 3. Program flow chart

Ω_1. These are represented as arrays of the form,

$$(3, ny_1, nx_1, nz_1)$$

for the vectors in Ω_1, i.e., of the form $\mathbf{v}_1$. here ny_1, nx_1 and nz_1 are the respective dimensions of the parallelepiped grid in Ω_1. And

$$(3, ny_c, nx_c, nz_c)$$

for the coarse–grid vectors in Ω, i.e., of the form $\mathbf{v}_2$ where ny_c, nx_c and nz_c are the respective dimensions of global coarse grid in Ω.

- The actions $A \begin{bmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \end{bmatrix}$ and $A_{12}\mathbf{v}_2$ and $A_{21}\mathbf{v}_1$ are implemented as shown in subsection 2.2. They are computed by the routines *A21NEW* and *A12NEW* respectively.
- Subroutine *PPCCGG* is a standard PCG routine which implements the preconditioned CG method. Here we exploit the actions of matrix A on vectors and the preconditioning solutions step.
- Subroutine *DLINVY* computes the inverse action of the BEPS preconditioner on given composite vector. The BEPS preconditioner as presented in Algorithm 3.1 utilizes solutions with the explicitly assembled matrix blocks A_{11} and $\widetilde{A}$, those are on regular (parallelepiped) grids. These solutions are based (at the present) on inner PCG methods using as preconditioners block–ILU factorization matrices that exploit block–size reduction as described in Chan and Vassilevski [3] (proposed

earlier in [2]). However, any other preconditioner available for 3D elasticity problem on parallelepiped mesh can be used as well. The group at CICT is now working on the construction of simple and inexpensive circulant matrix based preconditioners for 3D elasticity problems.

The program structure is shown in Figure 5. The block ASSEMBLING 1 stands for assembling on a standard parallelepiped mesh which we use to assemble the global coarse–grid stiffness matrix and the stiffness matrix A_{11} (corresponding to Dirichlet boundary conditions on the interface Γ). The block ASSEMBLING 2 is used to assemble at the same time the matrix blocks for the grid with local refinement (which corresponds to Neumann boundary conditions at the interface Γ) both outside and in the interior of Ω_1.

REFERENCES

[1] J.H. Bramble, R.E. Ewing, J.E. Pasciak and A.H. Schatz, *A preconditioning technique for the efficient solution of problems with local grid refinement*, Comput. Meth. Appl. Mech. Eng., 67(1988), 149-159.

[2] T.F. Chan and P.S. Vassilevski, *A framework for block–ILU factorizations using block–size reduction*, Math. Comp. 64(1995), 129–156.

[3] T.F. Chan and P.S. Vassilevski, *Convergence analysis of block–ILU factorization algorithms exploiting coarse–space block–size reduction for finite element elliptic equations*, CAM Report **95–46**, Math. Dept., UCLA, 1995.

[4] A. Georgiev, A. Baltov and S. Margenov, *On some practical requirements to HIPERGEOS applications in the bridge engineering*, Copernicus Technical Report # 1, 1995.

[5] A. Georgiev, S. Margenov, P.S. Vassilevski and L. Zikatanov, *Application of the block–size reduction BILU algorithm for 3D simulation of pile foundations*, Copernicus Technical Report, March, 1996.

[6] R.E. Ewing, R.D. Lazarov and P.S. Vassilevski, *Local refinement techniques for elliptic problems on cell-centered grids; II. Optimal order two-grid iterative methods*, Numer. Lin. Alg. Appl. 1(1993),

[7] R.E. Ewing and P.S. Vassilevski, *Two-level iterative refinement preconditioners*, Proceedings of Fifth Conference on DOMAIN DECOMPOSITION METHODS FOR PDEs, May 6-8, 1991, Norfolk, Virginia, SIAM, Philadelphia, 1992, pp. 262-270.

[8] S.I. Petrova, *Efficiency of Local Refinement Preconditioners with Inexact Subdomain Solvers*, Tech.Rep. Report IT/IF/14-21, Université Libre de Bruxelles, February 1995.

[9] S.I. Petrova, *Practical Implementation of Local Refinement Preconditioners*, Proceedings of the Second IMACS International Symposium on Iterative Methods in Linear Algebra, June 17-20, 1995, Blagoevgrad, Bulgaria.

CENTER OF INFORMATICS AND COMPUTER TECHNOLOGY, BULGARIAN ACADEMY OF SCIENCES, "ACAD. G. BONTCHEV" STREET, BLOCK 25 A, 1113 SOFIA, BULGARIA

E-mail address: `panayot@iscbg.acad.bg`

INSTITUTE OF MATHEMATICS, BULGARIAN ACADEMY OF SCIENCES, "ACAD. G. BONTCHEV" STREET, BLOCK 8, 1113 SOFIA, BULGARIA

E-mail address: `lyudmil@iscbg.acad.bg`