Primljen / Received: 10.3.2017.
Ispravljen / Corrected: 25.5.2017.
Prihvaćen / Accepted: 20.6.2017.
Dostupno online / Available online: 10.8.2017.

Iterated Ritz Method for solving systems of linear algebraic equations

Authors:

Prof. Emerituss Josip Dvornik, PhD. CE
University of Zagreb
Faculty of Civil Engineering
Department of Engineering Mechanics
dvornik@grad.hr

Prof. Damir Lazarević, PhD. CE
University of Zagreb
Faculty of Civil Engineering
Department of Engineering Mechanics
damir@grad.hr

Original scientific paper

Josip Dvornik, Damir Lazarević

Iterated Ritz Method for solving systems of linear algebraic equations

The paper describes research state of a new iterative method for solving systems of linear algebraic equations. The method is suitable for extremely large systems with sparse matrices. In addition to its own characteristics, it also has a feature of generality, as many iterative methods are only special cases of this approach. The algorithm was developed independently, and then implemented into the open source code program FEAP. Also, various checks were conducted, especially on practical models. Although the method has been only partially studied, good results have already been obtained.

Key words:
iterative method, Jacobi method, Gauß – Seidel method, method of steepest descent, conjugate gradient method

Izvorni znanstveni rad

Josip Dvornik, Damir Lazarević

Iterirani Ritzov postupak za rješavanje sustava linearnih algebarskih jednadžbi

U radu je opisano stanje razvoja nove iteracijske metode za rješavanje sustava linearnih algebarskih jednadžbi. Metoda je pogodna za izrazito velike sustave slabo popunjenih matrica. Osim vlastitih obilježja, posjeduje i svojstvo općenitosti, jer su mnogi iteracijski postupci samo poseban slučaj ovoga pristupa. Algoritam je realiziran samostalno, a potom je pridružen programu otvorena koda FEAP. Provedene su i raznolike provjere, posebice na praktičnim modelima. Premda je postupak tek djelomice istražen, već pokazuje dobre rezultate.

Ključne riječi:
iteracijski postupak, Jacobijeva metoda, Gauß – Seidelova metoda, metoda najstrmijega silaska, metoda konjugiranih gradijenata

Wissenschaftlicher Originalbeitrag

Josip Dvornik, Damir Lazarević

Iteririertes Ritz-Verfahren zur Lösung von linearen Gleichungssystemen

In der Arbeit wird der aktuelle Entwicklungsstand des neuen Iterationsverfahrens für die Lösung von linearen Gleichungssystemen beschrieben. Das Verfahren eignet sich insbesondere für sehr große Systeme mit schwach gefüllten Matrizen. Es besitzt neben eigenen Merkmalen auch das Merkmal der Allgemeinheit, da viele Iterationsverfahren nur ein Spezialfall dieses Ansatzes sind. Der Algorithmus wurde selbständig ermittelt und danach dem FEAP Programm mit einem offenen Programmcode zugeordnet. Es wurden auch zahlreiche Prüfungen vorgenommen, insbesondere an praktischen Modellen. Obwohl das Verfahren erst teilweise untersucht wurde, zeigt es gute Ergebnisse.

Schlüsselwörter:
Iterationsverfahren, Jacobi-Verfahren, Gauß-Seidel-Verfahren, Methode des steilsten Abstiegs, Methode der konjugierten Gradienten

1. Brief theoretical introduction

In the analysis of engineering models using numerical methods, usually system of algebraic equations

$$\mathbf{Ku=f} \tag{1}$$

(often very large) has to be efficiently solved. If displacement method is considered, the system of equations is related to equilibrium conditions. In this case, **K** is the stiffness matrix, **u** is the vector of unknown displacements, and **f** is the external load vector. If system matrix is symmetric and positive definite, the solution is equivalent to the minimization of the quadratic form that represents potential energy of a static system:

$$\Pi(\mathbf{u})=\frac{1}{2}\mathbf{u}^{\mathrm{T}}\mathbf{Ku}-\mathbf{u}^{\mathrm{T}}\mathbf{f} \tag{2}$$

The first term is the strain energy, and the second one is the work (potential) of external load. This equation is a discretized approximation of the Lagrange energy functional of the continuous (mathematical) model of a linearly elastic body. The surface of the constant energy level $\Pi(\mathbf{u}) = c$ is the ellipsoid (hyperellipsoid) that can be represented by the

$$(\mathbf{u}-\mathbf{u}_0)^{\mathrm{T}}\mathbf{K}(\mathbf{u}-\mathbf{u}_0)=1 \tag{3}$$

It is a body of n – dimensional space, where n denotes the number of unknown degrees of freedom. From the geometric viewpoint, components of vector **u** are point space coordinates, while components of vector $\mathbf{u}_0$ are ellipsoid centres. The matrix **K** (of the order n) is given by the product $\mathbf{Q}^{\mathrm{T}}\mathbf{DQ}$. The orthogonal matrix **Q** consists of the columns that define ellipsoid axes, i.e. eigenvectors. The elements of diagonal matrix **D** are $(2c/\lambda_i)^2$, where c is the energy level, and λ_i is corresponding eigenvalue. The lengths of ellipsoid semi-axes are $2c/\lambda_i$. Centre is defined at $c = 0$. The possibilities in plane ($n = 2$) and in space ($n = 3$) are shown on Figures 1.a and 1.b. According to the principle of minimum potential energy of a stable body, the point in space with the lowest energy level is the solution to the problem, and lies at the centre of all ellipsoids. It can be considered as degenerate ellipsoid, with the lengths of all main axes equal to zero. In numerical realization, depending on the accepted accuracy, it is however a very small ellipsoid with the energy level just a little higher than the minimum one.

2. Iteration idea

An approximate solution, while keeping the symbol **u**, can be found by successive application of the discretized Ritz method. The idea is based on the selection of linearly independent coordinate vectors $\boldsymbol{\phi}_i$ in the direction of which, with appropriate scalars a_i, solution increment can be expanded [1]:

$$\Delta\mathbf{u} = \sum_{i=1}^{m} a_i\boldsymbol{\phi}_i \tag{4}$$

In the traditional realization of the procedure, the unknowns a_i and vectors $\boldsymbol{\phi}_i$ are known as Ritz coefficients and Ritz vectors. The number of vectors must satisfy the inequality $1 \le m \le n$. If a rectangular matrix with m columns and n rows is defined in such a way that the columns are coordinate vectors, i.e.

$$\boldsymbol{\Phi} = [\boldsymbol{\phi}_1\ \boldsymbol{\phi}_2 \ldots \boldsymbol{\phi}_m] \tag{5}$$

and if vector **a** comprises scalars

$$\mathbf{a} = [a_1\ a_2 \ldots a_n]^{\mathrm{T}} \tag{6}$$

the solution increment can be written as

$$\Delta\mathbf{u} = \boldsymbol{\Phi}\,\mathbf{a} \tag{7}$$

Now, in the sense of iterative process:

$$\mathbf{u}_{i+1} = \mathbf{u}_i + \Delta\mathbf{u}_i \tag{8}$$

where indices denote two consecutive iteration steps. If the energy in the i – th step is

$$\Pi_i = \frac{1}{2}\,\mathbf{u}_i^{\mathrm{T}}\,\mathbf{K}\,\mathbf{u}_i - \mathbf{u}_i^{\mathrm{T}}\mathbf{f} \tag{9}$$

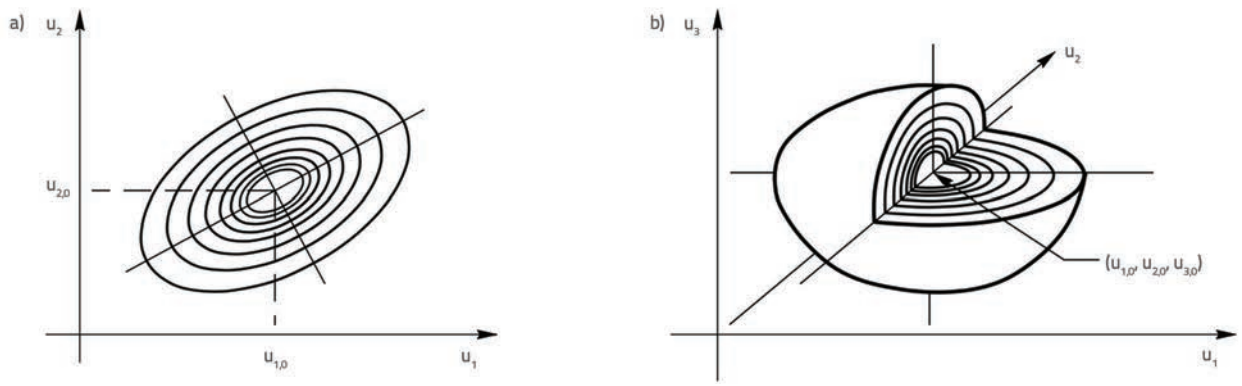

Figure 1. Energy ellipse and ellipsoid

then, considering (7) and (8), in the next step it is

$$\Pi_{i+1} = \frac{1}{2}(\mathbf{u}_i^T + \mathbf{a}_i^T\boldsymbol{\Phi}_i^T)\,\mathbf{K}(\mathbf{u}_i + \boldsymbol{\Phi}_i\mathbf{a}_i) - (\mathbf{u}_i^T + \mathbf{a}_i^T\boldsymbol{\Phi}_i^T)\,\mathbf{f} \tag{10}$$

By multiplying sub-expressions in parentheses and arranging the newly created terms, we obtain:

$$\Pi_{i+1} = \Pi_i + \frac{1}{2}\,\mathbf{a}_i^T\boldsymbol{\Phi}_i^T\,\mathbf{K}\,\boldsymbol{\Phi}_i\mathbf{a}_i - \mathbf{a}_i^T\boldsymbol{\Phi}_i^T\mathbf{r}_i \tag{11}$$

where the unbalanced load, or the residual is given by

$$\mathbf{r}_i = \mathbf{f} - \mathbf{K}\mathbf{u}_i \tag{12}$$

Since the energy Π does not depend on the displacement increment $\Delta\mathbf{u}_i$, and hence also not on variables $\mathbf{a}_i$, it can be omitted from the minimization procedure, i.e. it is sufficient to differentiate the energy increase

$$\Delta\Pi_i = \frac{1}{2}\,\mathbf{a}_i^T\boldsymbol{\Phi}_i^T\,\mathbf{K}\,\boldsymbol{\Phi}_i\mathbf{a}_i - \mathbf{a}_i^T\boldsymbol{\Phi}_i^T\mathbf{r}_i \tag{13}$$

If the generalized Ritz stiffness matrix is introduced

$$\mathbf{K}_i = \boldsymbol{\Phi}_i^T\,\mathbf{K}\,\boldsymbol{\Phi}_i \tag{14}$$

which is also symmetric, and in the case of linearly independent coordinate vectors also positive definite, and if the generalized Ritz load vector is defined as

$$\bar{\mathbf{r}}_i = \boldsymbol{\Phi}_i^T\mathbf{r}_i \tag{15}$$

the energy increase can be written in an abbreviated form

$$\Delta\Pi_i = \frac{1}{2}\,\mathbf{a}_i^T\,\mathbf{K}_i\,\mathbf{a}_i - \mathbf{a}_i^T\bar{\mathbf{r}}_i \tag{16}$$

to which after minimization (differentiation) according to , the following equation system can be related

$$\mathbf{K}_i\mathbf{a}_i = \bar{\mathbf{r}}_i \tag{17}$$

In the sense of the Ritz method, this system is used to approximate the initial one (1). As the approximation is usually unsatisfactory, an iterative improvement should be made. By solving the system (17), symbolically written as

$$\mathbf{a}_i = \mathbf{K}_i^{-1}\bar{\mathbf{r}}_i, \tag{18}$$

the coefficients a_i are obtained and then an approximate displacement increment $\Delta\mathbf{u}_i$ is calculated according to (7) $\mathbf{u}_{i+1}$. The new displacement is determined according to (8). The procedure terminates after the stopping criterion is satisfied. Usually, the Euclidean norm of the residual is adopted, i.e. iterative process is finished if

$$\|\boldsymbol{r}_i\|_2 \le \varepsilon\|\boldsymbol{r}_0\|_2 \tag{19}$$

where $\mathbf{r}_0$ is the initial residual ($\mathbf{r}_0 = \mathbf{f}$ if $\mathbf{u}_0 = \mathbf{0}$), while is a very small positive number. It should be noted that the current residual can also be determined from the recursive formula

$$\mathbf{r}_i = \mathbf{r}_{i-1} - \mathbf{K}\Delta\mathbf{u}_{i-1} \tag{20}$$

obtained using expression (12) in which, according to (8), should be inserted $\mathbf{u}_i = \mathbf{u}_{i-1} + \Delta\mathbf{u}_{i-1}$ and recognized that $\mathbf{r}_{i-1} = \mathbf{f} - \mathbf{K}\mathbf{u}_{i-1}$. The method is accelerated by expression (20) as, unlike expression (12), the total displacement does not need to be calculated in each step. In fact, it is determined only at the very end, when stopping criterion is reached. However, if (20) is applied, based on equilibrium conditions (12) the residual must be updated occasionally, because of accumulation of rounding errors. Equilibrium condition is also needed in the very beginning of the method (for), to calculate the initial residual if $\mathbf{u}_0$ is not a null vector. In case of convergence problems, the maximum number of steps must be limited, in order to avoid time consuming calculation.

3. Overrelaxation (underrelaxation) of displacement

Motivated by the known method of successive overrelaxation, the procedure can be accelerated using the relaxation factor Ω_i, by which the solution increment is multiplied. Then instead of using (8), the new displacement is defined as

$$\mathbf{u}_{i+1} = \mathbf{u}_i + \Omega_i\Delta\mathbf{u}_i \tag{21}$$

In fact, in most cases the local energy minimum (within the subspace used) is not globally optimal. The convergence can often be accelerated by overrelaxation (using $\Omega_i > 1$), or by underrelaxation ($\Omega_i < 1$). In both cases system (17) will not be satisfied, and the energy will not achieve a minimum within the given subspace. However, although such a step is not locally optimal, it can speed up global convergence of the method. Unfortunately, an appropriate value of the relaxation factor is not easily selected from one step to another. It is only known that the relation $0 < \Omega_i < 2$ is valid. Although a mathematical proof of these limits exists [2], it can be given a clear interpretation. As Π is a quadratic function, search for a minimum along a certain direction $\mathbf{p}_i$ (Figure 2), which usually coincides with $\Delta\mathbf{u}_i$, can be described by the following function:

$$\Pi(\Omega_i) = \Pi_{\min} + (\Pi_0 - \Pi_{\min})(\Omega_i - 1)^2$$

where Π_0 represents the energy at the starting point of the search direction (at the beginning of the step), Π_{min} is the local minimum along this direction, and $\Pi(\Omega_i)$ is the value at some point of the quadratic function. Based on the condition of monotone convergence of the method $\Pi(\Omega_i) < \Pi_0$, the following can be written: $(\Omega_i - 1)^2 < 1$ or $0 < \Omega_i < 2$. At interval boundaries the energy remains the same as the initial one, i.e. $\Pi(0) = \Pi(2)$, and so the method does not converge. For the values Ω_i outside of boundaries, the energy increases and the procedure diverges.

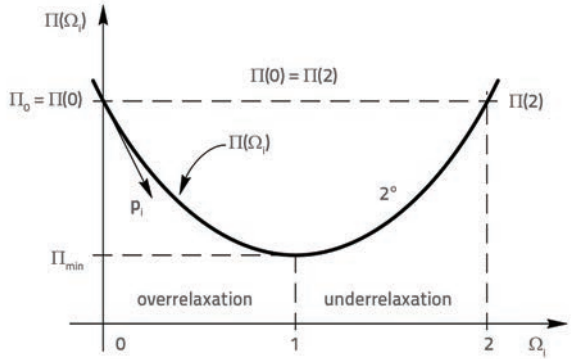

Figure 2. Dependence of energy on relaxation factor

4. Iteration by means of a small system

The procedure is started by selecting the initial approximation $\mathbf{u}_0$. Then, at each step, the set of coordinate vectors of the matrix is defined $\mathbf{\Phi}_i$ and the vector $\Delta\mathbf{u}_i$ is determined. This increase (which is in the subspace spanned by coordinate vectors determined by the vector $\mathbf{a}_i$), provides the largest reduction of energy $\Delta\Pi_i$ within that subspace. This is why a system of linear algebraic equations (17) has to be solved, but with a small number of unknowns, equal to the number of coordinate vectors. In this way, solution of the initial system containing n equations is reduced to multiple solving of the system with m equations. In some of our examples, n was about 10^7, while m was no more than ten. Known schemes of matrix transformations that cause formation of a small system are shown on Figure 3.

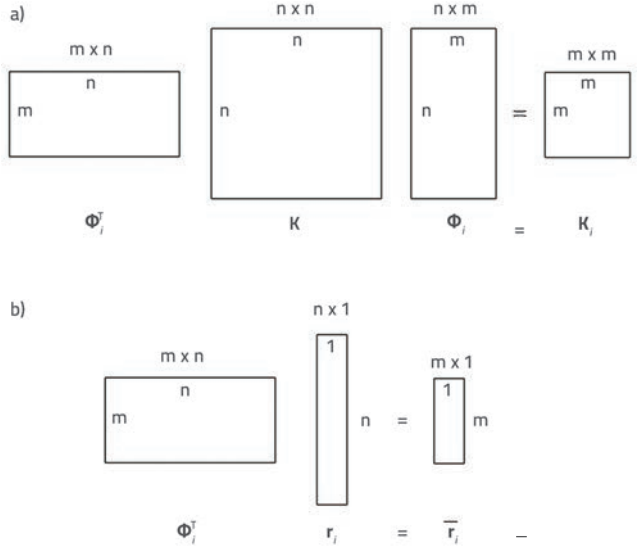

Figure 3. Formation of a small system: a) matrix , b) vector

The solution of this system (matrix $\mathbf{K}_i$ is usually full), can be obtained using any direct solution method. Cholesky decomposition was used in our case. If the iterative process is convergent, the sum of small-system solutions approaches large-system solution and the sum of the small system energies increases monotonically and approaches minimum energy of a large system. The convergence is not guaranteed in the case of linearly dependent coordinate vectors (when the subspace degenerates), orthogonality of the residual on the current subspace, and the effect of rounding error at the end of iterative process (if the stopping criterion is too strict). These problems are additionally discussed in Section 7.

5. Basic pseudocode

A highly natural and not very complex idea behind this algorithm is easily noticed. Despite such advantages, this approach and interpretation have not been widely accepted by researchers in the field of efficient iterative methods for solving large systems [3]. Theoretically, this algorithm, just like many other iterative algorithms, can be considered as a Krylov approach [4, 5], and some similarities with ideas given in this paper can be found in [6]. It is interesting to note that the subspace iteration, which is in essence this procedure, has quite a wide application in solving eigenvalue problems. Main elements of the described method are briefly presented by a simple pseudocode.

Table 1. Iterated Ritz procedure

	Necessary: K, **f**, ε stiffness matrix, load vector, stopping criterion
1.	**Result: u** displacement vector
2.	$i \leftarrow 0$ step counter
3.	$\mathbf{u}_i \leftarrow \mathbf{0}$ initial solution null – vector
4.	$\mathbf{r}_i \leftarrow \mathbf{f}$ residual equal to load
5.	**repeat**
6.	$\mathbf{\Phi}_i \leftarrow [\boldsymbol{\varphi}_{1,i}\ \boldsymbol{\varphi}_{2,i}\ldots\ \boldsymbol{\varphi}_{m,i}]$ definition of coordinate vectors
7.	$\mathbf{K}_i \leftarrow \mathbf{\Phi}_i^T\ \mathbf{K}\ \mathbf{\Phi}_i$ formation of a "small" system matrix
8.	$\bar{\mathbf{r}}_i \leftarrow \mathbf{\Phi}_i^T\mathbf{r}_i$ formation of a "small" right hand side vector
9.	$\mathbf{a}_i \leftarrow \mathbf{K}_i^{-1}\bar{\mathbf{r}}_i$ solution of a "small" system
10.	$\Delta\mathbf{u}_i \leftarrow \mathbf{\Phi}_i\mathbf{a}_i$ determination of an solution increment
11.	$\mathbf{u}_{i+1} \leftarrow \mathbf{u}_i + \Delta\mathbf{u}_i$ calculation of a new displacement
12.	$\mathbf{r}_{i+1} \leftarrow \mathbf{f} - \mathbf{K}\mathbf{u}_{i+1}$ new residual
13.	$i \leftarrow i + 1$ increase of the step counter
14.	**until** $\|\mathbf{r}_i\|_2 \le \varepsilon\|\mathbf{r}_0\|_2$

The efficiency of the procedure is subject of compromise. If a large number of well-chosen coordinate vectors are used (spanning the subspace in which good approximate solution lies), the energy reduction per step will be greater, less steps will be needed for finding the solution, but individual steps will last longer. A smaller number of vectors imply shorter step time, but also a less efficient subspace, and more steps to obtain the solution. An optimal approach would be to find a good-quality small sized subspace, so that even a stronger stopping criterion can be reached after a smaller number of fast steps.

6. Special cases of the method

The rectangular matrix $\mathbf{\Phi}_i$ is called the subspace matrix. As already pointed out, matrix columns are coordinate vectors $\boldsymbol{\phi}_{1,i}$ to $\boldsymbol{\phi}_{m,i}$ that form this subspace. Depending on the choice of these vectors, many known iterative methods can be distinguished. They will however not be explained in this text [7-11], i.e. only the well-known ones will be presented as special cases of this approach. It is only necessary to use appropriate coordinate vectors. Brief descriptions given below are not intended for faster realization of these methods, compared to traditional strategies [12], i.e. they only contribute to proper understanding and emphasize the generality of this iterative algorithm. It should also be mentioned that very popular single or multiple preconditioning [13, 14] can be interpreted using (one or several) coordinate vectors. For this procedure, the preconditioning does not imply any significant change of the algorithm (cf. Section 7.2.3).

6.1. The Jacobi method

The Jacobi method is obtained if only one coordinate vector $\boldsymbol{\phi}_{m,i} = \mathbf{D}^{-1}\mathbf{r}_i$ is defined at each step. Here, $\mathbf{D}$ is the diagonal matrix with diagonal elements of $\mathbf{K}$. Thus the matrix $\mathbf{\Phi}_i$ becomes the column matrix, i.e. $\mathbf{\Phi}_i = [\boldsymbol{\phi}_i]$, and $\mathbf{K}_i$ and $\bar{r}_i$ degenerate into scalars. Therefore, a small system (17) is reduced to a single equation. Solution in every step defines the displacement increment.

6.2. The methods of Gauß – Seidel and Successive overrelaxation

The procedure can be described by a sequence of cycles or "crosses" through all degrees of freedom. Then the cycle consists of steps. Only one degree of freedom is solved at each step (using node numbering sequence). The process is often called relaxation, and contains the matrix $\mathbf{\Phi}_i$ which in every step has one coordinate vector equal to the orth, i.e. $\boldsymbol{\phi}_i = \mathbf{e}_i$. The component corresponding to the degree of freedom that is currently relaxed is equal to one, while all others are equal to zero. Matrix $\mathbf{K}_i$ and vector $\mathbf{r}_i$ degenerate into scalar, therefore, only one equation needs to be solved. The equilibrium of only the corresponding degree of freedom is satisfied in this way, as the residual of the previous equation is disturbed at the same time. During the convergent iterative process, the disturbance gradually decreases. The cycle ends when all equations (steps) are solved, and only the last one is in equilibrium. This is followed by the start of a new cycle, with the same coordinate vectors. With the steps progress, the component equal to one moves from the first to the last vector component again.

If the matrix approach is applied (by cycles that correspond to steps in other methods), and if the solution is sought using the node numbering order, each cycle has one coordinate vector $\boldsymbol{\phi}_i = \mathbf{L}^{-1}\mathbf{r}_i$, where $\mathbf{L}$ is the lower triangular matrix (contains the lower triangular part and diagonal elements of $\mathbf{K}$). In this interpretation, the method of successive overrelaxation is described by the same matrix, but the diagonal elements are multiplied by the local relaxation factor ω, which should be distinguished from the global . Then, even the equilibrium of the degree of freedom just solved is no longer valid (In the Gauß – Seidel method $\Omega = \omega = 1$). If the solution is sought opposite to the node numbering, then the lower triangular matrix should be replaced with the upper one, i.e. $\boldsymbol{\phi}_i = \mathbf{U}^{-1}\mathbf{r}_i$.

There are also other ways of relaxation, for instance, there is the known "chessboard scheme" on the rectangular mesh. In the first half of the cycle only "black" nodes are solved, while "white" ones are solved in the second half of the cycle. Relaxation is also possible by numbering from right to left (instead of using the traditional left to right). If diagonal connections between nodes are used, the number of possible relaxation paths will increase. New access possibilities are obtained if, in addition, the columns and rows interchange places. Further possibilities exist in the three dimensional rectangular meshes, especially with diagonal connections between nodes, both, in coordinate planes and along space diagonals. Irregular meshes from the finite element method are even more complex, and have a huge number of meaningful connections between nodes, and hence numerous ways of load (residual) propagation.

An interesting idea is to go from the highest (by absolute value) to the lowest residual of the current cycle. However, reordering is needed after solution of each equation (after every step), due to the change of residual in the current node and its topological neighbours. Of course, continuous correction (sorting) of residual requires additional time, which affects the numerical efficiency.

In principle, the algorithm redefined in this way needs a smaller number of steps to reach the solution. In fact, known methods of Cross and Werner-Csonka, converge better if started from the node with the largest residual established. Then, residual vector has to be updated and the node with the largest residual has to be equilibrated again. The procedure no longer depends on nodes numbering, but rather on the distribution of residual (load). That is why the notion of cycle loses its real meaning as, before some nodes are equilibrated for the first time, the other ones have already been solved several times.

6.3. Method of steepest descent

In this method also, only one coordinate vector equal to residual vector is used in each step. Thus, $\boldsymbol{\phi}_i = \mathbf{r}_i$. It is also a gradient of the energy functional with a negative sign, usually used for traditional realization of the procedure, i.e. $\mathbf{r}_i = -\nabla\Pi(\mathbf{u})$. If matrix $\mathbf{K}$ is symmetric and positive definite, the method is convergent, but the convergence is usually slow, especially in the case of a poorly conditioned matrix. Although this method is not efficient, it is used as a pedagogical introduction and motivation for improvement of other iterative methods.

6.4. Conjugate gradient method

In the first step there is only one coordinate vector, the residual, as in the method of steepest descent. After that, the vector of previous solution increment is added, which accelerates the convergence. That is why this method is much faster than the previous one. It can be written: $\boldsymbol{\phi}_{1,i} = \mathbf{r}_i$ and $\boldsymbol{\phi}_{2,i} = \Delta\mathbf{u}_{i-1}$. Thus, the small matrix is of the order two, i.e. $\mathbf{\Phi}_i = [\boldsymbol{\phi}_{1,i}\boldsymbol{\phi}_{2,i}]$, and a system with

two unknowns should be solved at each step. This description differs from the usual ones where the **K**- orthogonalization of vectors is used. However, this orthogonalization equal to solving of the abovementioned system with two unknowns.

The orthogonality property of these two vectors is gradually lost due to accumulating of round-off errors. Nevertheless, even the approximate orthogonality from adjacent steps greatly speeds up the convergence, compared to the steepest descent method. This theory has been confirmed by numerical tests on relatively small systems. The residual norm suddenly drops to zero (or more precisely to the numerical approximation of zero) exactlyin the n – th step. On the other hand, the tests conducted on large systems show smooth behaviour in the n – th step, when drop could theoretically be expected. That is why the method needs stopping criterion. In other words, it behaves like usual iterative method.

Although the method is relatively fast, and in the case of symmetric and positive definite matrix has a practical use, the convergence in not always satisfying, especially if the matrix is ill conditioned. That is why many preconditioning techniques are often used, and the preconditioned conjugated gradient method is frequently adopted. Then the procedure can terminate (satisfy the stopping criterion) after a much smaller number of steps compared to the theoretically required . Let's close this section with a somewhat optimistic citation [4]: "The choice of the [iterative] method is a delicate problem. If the [system] matrix A is symmetric and positive definite, then the choice is easy: Conjugate Gradients." In accordance with our method, that means that there is actually no better subspace of coordinate vectors than the plane? It seems that there should be enough room for improvement.

7. On the selection of an efficient subspace of coordinate vectors

We would like to define the matrix $\mathbf{\Phi}_i$ so that the method converges much faster compared to traditional procedures. Although we wish to keep the number of coordinate vectors small, we think that only one vector (such as in Jacobi, Gauß – Seidel, successive overrelaxation and steepest descent), or two (in the case of conjugate gradients), are less than optimum. Obviously, as the number of vectors is much smaller than , the solution increment $\Delta\mathbf{u}_i$ can only accidentally hit the solution **u** in the early phase of the calculation.

We can imagine that, in addition to (two) coordinate vectors of the conjugate gradient method, a third vector is introduced. In such a case, the subspace is extended to three vectors. Compared to the subspace that has two vectors only, a greater energy reduction per step can reasonably be expected. The contribution of this additional vector can, in the worst case, be equal to zero. The following conclusion is also possible: the minimum of the energy functional in a larger subspace cannot be greater (at a higher level) than the minimum in a smaller subspace. In that sense, it is possible to add the fourth, fifth and additional vectors and even greater reduction of energy per step can be expected. In this way, most of the energy from the static system should be exhausted in several steps and the system can be "damped out" to the lowest point – the solution of the problem. The idea of expanding the subspace is obviously quite attractive, but only up to a certain point. On the one hand, formation of vectors must not be time consuming process. On the other hand, if number of vectors is excessively increased, small system (to be solved in each step) could be unacceptably large. Ultimately, if the subspace dimensions are equal to the number of unknowns, the total energy can be minimized and the solution obtained in the first step (or maybe in the second step due to rounding errors), but at a "price" equal or greater than needed for finding direct solution of the problem.

7.1. Necessary conditions of the selection

A small matrix can be singular (or almost singular – ill conditioned) if some coordinate vectors are exactly (or almost exactly) linearly dependent. It should be recalled that vectors $\boldsymbol{\phi}_i$ are linearly independent if the expression

$$\sum_{i=1}^{m} a_i \boldsymbol{\phi}_i = \mathbf{0} \tag{22}$$

is valid only in the case of all $a_i = 0$. In numerical realization, this condition should be even stronger: vectors should not be even "almost" linearly dependent. Then, the Euclidian norm of linear combination of vectors can be smaller than a small positive number δ, i.e.

$$\left\| \sum_{i=1}^{m} a_i \boldsymbol{\phi}_i \right\|_2 < \delta \tag{23}$$

only if all coefficients are $|a_i| < \delta$. The violation of condition (22), and especially (23) is not automatically prevented by the various strategies used to generate coordinate vectors. (as discussed below). The vectors that do not fulfil these conditions should be discarded. Although this reduces subspace dimension, the small matrix becomes regular and better conditioned. If more than two vectors are linearly dependent, it is not clearly defined which of them should be rejected. Then the exact or approximate orthogonalization of such vectors should be considered. In any case, the situation in which vectors are generated and then rejected, orthogonalized, or maybe even replaced by others, makes step more "expensive", and therefore should only occasionally happen. Obviously, procedure in which it is not possible (or it is rarely possible) to generate dependent vectors should be considered. For easier realization of such procedure the generation method can even be changed during the calculation. If the dependence nevertheless happens, there is a "last moment solution" as during decomposition some pivots of the matrix $\mathbf{K}_i$ become equal (close) to zero. This can be recognized and used for discarding the corresponding equations from calculation of the small system. This subspace reduction has proven to be a fast and simple solution of linear dependence problems.

However, formation of the subspace is one thing, but its quality is something completely different. For instance, if coordinate vectors

are perpendicular to the residual vector, the subspace is not useful. Then the right hand side of the small system (Ritz load vector) is $\bar{\mathbf{r}}_i = \mathbf{\Phi}_i^{\mathrm{T}}\mathbf{r}_i = \mathbf{0}$. This results in $\mathbf{a}_i = \mathbf{0}$ and $\Delta\Pi_i = 0$, and so the energy is not reduced. In other words, the procedure does not converge. To avoid this problem, the norm $\|\boldsymbol{\phi}_i^T\mathbf{r}_i\|_2 / \|\mathbf{r}_i\|_2$ should be greater than a small constant. To achieve this, it is sufficient to have one coordinate vector $\boldsymbol{\phi}_i$ that is not orthogonal to residual $\mathbf{r}_i$. For example, it can be determined by multiplying the selected positive definite matrix $\mathbf{P}$ (of order n) by vector of the residual: $\boldsymbol{\phi}_i = \mathbf{P}\mathbf{r}_i$. Then, according to the definition of positive definiteness $\boldsymbol{\phi}_i^{\mathrm{T}}\mathbf{r}_i = \mathbf{r}_i^{\mathrm{T}}\mathbf{P}\mathbf{r}_i > 0$, unless $\mathbf{r}_i$ is a null – vector, but this means that the solution has been achieved. To further explain this situation, something else should also be noted: If old coordinate vectors $\mathbf{\Phi}_i$ (from the previous step) are kept during the calculation of new residual $\mathbf{r}_{i+1}$, then the unfavourable case (mentioned above) would be obtained, as the new vector of the residual is always orthogonal to the old subspace, i.e. the following is valid: $\Omega = 1$ The statement is not valid in case of overrelaxation or underrelaxation of displacement, but only if $\mathbf{\Phi}_i^{\mathrm{T}}\mathbf{r}_{i+1} = \mathbf{0}$. Using symbols and relations introduced during description of the method, a simple proof can be as follows:

$$\begin{aligned}
\mathbf{\Phi}_i^{\mathrm{T}}\mathbf{r}_{i+1} &= \mathbf{\Phi}_i^{\mathrm{T}}(\mathbf{f} - \mathbf{K}\mathbf{u}_{i+1}) \\
&= \mathbf{\Phi}_i^{\mathrm{T}}\mathbf{f} - \mathbf{\Phi}_i^{\mathrm{T}}\mathbf{K}(\mathbf{u}_i + \Delta\mathbf{u}_i) \\
&= \mathbf{\Phi}_i^{\mathrm{T}}\mathbf{f} - \mathbf{\Phi}_i^{\mathrm{T}}\mathbf{K}\mathbf{u}_i - \mathbf{\Phi}_i^{\mathrm{T}}\mathbf{K}\Delta\mathbf{u}_i \\
&= \mathbf{\Phi}_i^{\mathrm{T}}(\mathbf{f} - \mathbf{K}\mathbf{u}_i) - \mathbf{\Phi}_i^{\mathrm{T}}\ \mathbf{K}\ \mathbf{\Phi}_i\mathbf{a}_i \\
&= \mathbf{\Phi}_i^{\mathrm{T}}\mathbf{r}_i - \mathbf{K}_i\mathbf{a}_i = \mathbf{\Phi}_i^{\mathrm{T}}\mathbf{r}_i - \bar{\mathbf{r}}_i \\
&= \mathbf{\Phi}_i^{\mathrm{T}}\mathbf{r}_i - \mathbf{\Phi}_i^{\mathrm{T}}\mathbf{r}_i = \mathbf{0}
\end{aligned} \tag{24}$$

Compared to the Ritz coordinate functions on the continuum, compatibility conditions and geometrical boundary conditions in this discrete alternative are not sought. If only the system of equations is known, rather than the static system from which it was generated, such properties are not even defined. It is only known that they are contained in the system matrix. However, a coordinate vector can intuitively be considered "smoother" if it contains a smaller relative contribution of "high modes" – eigenvectors of the matrix $\mathbf{K}$ with high eigenvalues. Such a vector forms more realistic coefficients of the Ritz matrix, because corresponding residual also has a small portion of high modes. The diagonal element of a small matrix related to such coordinate vector and the corresponding "generalized stiffness" are smaller. On the contrary, an excessively "rough" vector generates large stiffness in the Ritz matrix and causes locking of residual. This can easily be proven by expanding the coordinate vector $\boldsymbol{\phi}_i$ in the base of matrix eigenvectors:

$$\boldsymbol{\phi}_i = \sum_{j=1}^{n} h_j\mathbf{v}_j \tag{25}$$

In case of normalized vectors ($\mathbf{v}_j^{\mathrm{T}}\mathbf{v}_j = 1$), the corresponding diagonal element of the small matrix (briefly marked as system matrix) can be written as

$$k_{i,i} = \boldsymbol{\phi}_i^{\mathrm{T}}\mathbf{K}\boldsymbol{\phi}_i = \sum_{j=1}^{n} h_j^2\lambda_j \tag{26}$$

It can be observed that coefficients h_j^2 (to which a greater is attributed) are multiplied by a larger eigenvalue and thus contribute more to (the increase of) stiffness $k_{i,i}$. Let's mention, very rough vectors also have a great squared norm of residual $\|\mathbf{r}\|_2^2$. With the progress of the steps, smooth coordinate vectors effectively reduce contribution of low eigenvectors. After that, the procedure behaves like influence of these vectors does not exist. Thus, the condition number of the system matrix becomes smaller, and the speed of convergence increases. If additional similar vectors are added, the energy reduction should be greater. From the theoretical viewpoint, the "smoothness" property is not necessary, but it is included in this section, because it is important for an efficient realization of the method. Without it, the method is neither efficient nor competitive.
Although the above limitations reduce the possible choices, the set from which appropriate coordinate vectors can be selected still remains very large. Unfortunately, we are not (and as far as we know nobody is) familiar with good criteria for the selection of generally efficient vectors. In addition, the background theory that would make selection easier is also insufficiently known. The problem is that many possibilities arise and we can be only satisfied with the implementation and comparison of numerical tests on numerous examples. As a rule, a specific set of vectors works fine for some models, while it is quite bad for the others. In such circumstances, we would be satisfied with the selection of appropriate vectors, and finding the best set (several fast vectors not dependent on models) would be a great research result.

7.2. Several proposals for generation

There are two basic approaches to the generation of coordinate vectors: general and special. In the first approach, no additional data about the model are needed. The system matrix and the right hand side vector are sufficient. If they are properly defined, convergence of the method is satisfying in most practical cases. In the second approach some special features, valid only for the model that is currently considered, are exclusively used. Then, an excellent convergence is expected but only for this specific model (or perhaps for a small group of similar models). Some strategies for coordinate vectors generation are shown below, primarily according to the first and then to the second approach.

7.2.1. Selection of constant vectors

The simplest approach is to select a group of linearly independent vectors in advance. Such is the case in the methods of Gauß – Seidel or successive overrelaxation. The number of vectors coincides with the number of unknowns, and their sequence is cyclically repeated until convergence criterion is reached. This was considered in Section 6.

7.2.2. Selection based on current residual

An appropriate set of vectors can be defined using a current residual $\mathbf{r}_i$. In fact, as the "expensive" calculation of an initial error (we borrowed the symbol for the orth) immediately leads to the

correct solution, we can try to find the matrix $\mathbf{e}_0 = \mathbf{K}^{-1}\mathbf{r}_0$, which correctly approximates **P** with the smallest possible amount of calculation $\mathbf{K}^{-1}$. A good choice is to use positive definite matrix, but it is not absolutely necessary. It can be nonsymmetric and ill conditioned, and even singular, as all this does not imply singularity of the matrix $\mathbf{K}_i$. Of course, we must not generate all coordinate vectors with matrices of the same singularity.

As already pointed out, the coordinate vector is generated as $\mathbf{P}\,\mathbf{r}_i$. There is also an additional advantage in the vector of current residual, as it ensures non-orthogonality of coordinate subspace to this vector, which is important for convergence of the method. For instance, if **K** is approximated by identity matrix (**P** = **I**) we obtain $\boldsymbol{\phi}_i = \mathbf{r}_i$, which is really the steepest descent method. The Jacobi iteration is based on the slightly better approximation by diagonal matrix ($\mathbf{P} = \mathbf{D}^{-1}$). As already pointed out, then $\boldsymbol{\phi}_i = \mathbf{D}^{-1}\mathbf{r}_i$. These procedures are not efficient, as the ("cheap") replacement matrix **P** contains insufficient data about inverse of **K**. The following should be emphasized: if several coordinate vectors are used, it is not necessary that an individual vector approximates $\mathbf{K}^{-1}\mathbf{r}_i$ well, but rather that the subspace spanned by all vectors contains the best possible approximation of this product.

More appropriate coordinate vectors can be generated using one (or several) cycles of the Gauß – Seidel method or the method of successive overrelaxation. In such cases it would be useful to try various ways of "visiting" the nodes. However, unlike the classical realization, in order to save a computer time, an incomplete procedure should be used: return to the already relaxed nodes should be prohibited. The idea is good, but increases "the price" of the cycle. Perhaps, the nodes could be sorted in the first cycle (according to absolute values of residual components), while keeping or rarely correcting the order afterwards. Between cycles (and maybe between steps) it is desirable to introduce the local relaxation factor , but in this case an optimum value should be researched. Let us mark with $\mathbf{L}_\omega$ and $\mathbf{U}_\omega$ the lower and upper triangular matrix of **K**, whose diagonal elements are multiplied by ω. These matrices can be simply and quickly inverted and multiplied by vector. The coordinate vector determined by one cycle of the successive overrelaxation procedure, using the order of unknowns numbering (from first to last), can be presented as $\boldsymbol{\phi}_i = \mathbf{L}_\omega\mathbf{r}_i$. Using the same procedure, but in the opposite order, $\boldsymbol{\phi}_i = \mathbf{U}_\omega^{-1}\mathbf{r}_i$ is obtained. In both cases, the cycle starts with the null-vector. The coordinate vector can be generated by multiple use of the sum or product of these two approaches. Thus we have:

$$\boldsymbol{\phi}_i = (\mathbf{L}_\omega^{-1} + \mathbf{U}_\omega^{-1})(\mathbf{L}_\omega^{-1} + \mathbf{U}_\omega^{-1})\ldots\mathbf{r}_i \tag{27}$$

ili:

$$\boldsymbol{\phi}_i = \mathbf{L}_\omega^{-1}\mathbf{U}_\omega^{-1}\mathbf{L}_\omega^{-1}\mathbf{U}_\omega^{-1}\ldots\mathbf{r}_i \tag{28}$$

In the first approach, the matrix **K** can also be placed between the parentheses. The matrix **P** can easily be recognized in equations above. If such vectors are independently used, a smaller relaxation factor (close to one) makes the convergence slower, but guarantees faster process of "smoothing". It would therefore be efficient to use them as an addition to other coordinate vectors. The increase of the cycles number (successive changes of matrices $\mathbf{L}_\omega^{-1}$ and $\mathbf{U}_\omega^{-1}$), results in the greater smoothness of the vector (the contribution of higher eigenvectors decreases, and contribution to convergence in the region of lower eigenvalues increases). In this way, it is also possible to generate coordinate vectors using other iterative methods. It is even possible to use algorithms that are neither convergent nor numerically stable. Thus, the local relaxation factor does not need to lie within theoretically determined limits of the global factor (which has to be between 0 and 2).

Interesting coordinate vectors can be generated using the smoothening of the residual vector. In this case, we are talking about "filtering". A component of such vector is equal to the sum of residual components in the neighbouring nodes, multiplied by the weighting factors. This approach was found to be efficient in some examples with large jumps of residual function. These jumps occur due to bad prediction of displacement in some steps, which often appears in the region of supports and free boundaries of the model. Generally, if a residual is decomposed into eigenvectors of the matrix **K**, then two or three coordinate vectors that smoothen the lower part of the residual spectrum (low eigenvalues) should be formed, and additional two specialized for the upper part of the spectrum should be added. The previous displacement increment $\Delta\mathbf{u}_{i-1}$ should also be added, which is the origin of success of the conjugate gradient method. In this paper, coordinate vectors were generated using the symmetric successive overrelaxation procedure. Thus, the first vector was defined as

$$\boldsymbol{\phi}_1 = \mathbf{L}_\omega^{-1}\mathbf{D}\mathbf{U}_\omega^{-1}\mathbf{r}_i \tag{29}$$

and others were generated by recursive formula

$$\boldsymbol{\phi}_j = \mathbf{L}_\omega^{-1}\mathbf{D}\mathbf{U}_\omega^{-1}(\mathbf{K}\boldsymbol{\phi}_{j-1}),\ j = 2,\ldots,m \tag{30}$$

The previous displacement increment was also added. Calculations were made using different number of coordinate vectors (Section 9). Somewhat greater local coefficient of relaxation ω = 1,65 was selected. The global one was kept to Ω = 1. Let us now explain the product in parentheses. As one step in the direction of vector $\boldsymbol{\phi}_1$ gives current displacement increment $\alpha\boldsymbol{\phi}_1$, where α is a number, the residual $\mathbf{r}_i - \alpha\mathbf{K}\boldsymbol{\phi}_1$ is obtained according to (20). If the symmetric overrelaxation is applied to this residual, because of (29), the second vector becomes $\boldsymbol{\phi}_1 = \boldsymbol{\phi}_1 - \alpha\,\mathbf{L}_\omega^{-1}\mathbf{D}\mathbf{U}_\omega^{-1}(\mathbf{K}\,\boldsymbol{\phi}_1)$. Vector $\boldsymbol{\phi}_1$ already participates in the formation of the subspace, $\boldsymbol{\phi}_1$ and α influences only the length of the new vector (it does not change the subspace that this vector expands), therefore and can be dropped. Thus, the form (30) is obtained. An interesting, faster realization of this procedure, could be if **K** is used instead of **D**. These considerations are also of general significance. In this way, coordinate vectors can be generated using any iterative method, i.e. not only the symmetric relaxation. Different methods can also be used for every vector as well. For instance, the incomplete Cholesky factorization can be used for the first vector. Then, forward and backward successive overrelaxations are used for the second and third vectors. The

symmetric version of these methods can be used for the fourth and fifth vector, etc. Similarly to the described effect of one vector, here the subspace is smoothened by all vectors obtained by relaxation. Residual is smoother with an increase of the vectors used (or cycles in the formation of one vector), which contributes to faster convergence. A similar effect is obtained by the vector from the Jacobi method, with components r_i/k_{ii}.

7.2.3. Generation according to preconditioning

Various ideas used for traditional preconditioning of equation systems can be applied for generation of coordinate vectors. In addition to the matrices from the iterative procedures given above, the incomplete Cholesky factorization or the matrix polynomial of **K** are also used for the preconditioning matrix, let's mark it as **M** It is used for reduction of the system matrix condition number. Symbolically, instead of solving $\mathbf{K}\,\mathbf{u}=\mathbf{f}$ we indirectly solve

$$\mathbf{M}^{-1}\,\mathbf{K}\,\mathbf{u}=\mathbf{M}^{-1}\,\mathbf{f} \tag{31}$$

but the matrix **M** should be quite rapidly inverted. From our standpoint, $\mathbf{M}^{-1}$ is nothing else but **P**, and the coordinate vector is once again $\mathbf{Pr}_i$. Interestingly, some of our examples converged better by coordinate vectors generated using backward node numbering. However, in literature it is quite usual to use preconditioning procedure by forward numbering only. It would therefore be worthwhile to implement preconditioning technique using backward procedure. In order to generate several coordinate vectors, two or more preconditioning methods (of matrices **P**) can be used at the same time. This is analogous to multiple preconditioning. Then the efficiency of the step can increase. Consequently, the matrix transformations used for preconditioning are not necessary, i.e. in the sense of our method, the preconditioning is just the way of forming coordinate vectors. During generation process, they can become (either exactly or approximately) linearly dependent and one or several vectors must be excluded.

7.2.4. Selection based on previous displacement increment

An appriopriate set of coordinate vectors is based on history recycling, i.e. on the use of previous displacement increment $\Delta\mathbf{u}_{i-1}$. It is known that this vector significantly improves convergence of conjugate gradients with respect to the steepest descent, and it also enables use of relaxation factor. Therefore, it can increase efficiency of the method. In our interpretation, one of subspace vectors is $\boldsymbol{\phi}_i = \Delta\mathbf{u}_{i-1}$ and it has a similar effect, although (compared to the conjugate gradient method) the recursive orthogonality of the previous increments is lost. In our procedure (unlike many other procedures), there is only orthogonality between successive (not distant) solution increments, residuals or subspaces, as the orthogonalization makes the step "more expensive". If , successive orthogonality is also lost. That is why, during calculation, a small system can become ill conditioned and the convergence slower. The coordinate vectors could possibly be orthogonalized after a certain number of steps, but we are not inclined to do it. Additional vectors can be generated by multiplying the last displacement increment by some matrix, i.e. $\boldsymbol{\phi}_i = \mathbf{S}\Delta\mathbf{u}_{i-1}$. The adding of earlier increments $\Delta\mathbf{u}_{i-2}$, $\Delta\mathbf{u}_{i-3}$ and so on, was not sufficiently effective. Besides the last displacement increment, the current solution vector $\mathbf{u}_i$ can similarly be used as one of coordinate vectors. The "recycling" of this vector makes sense due to the loss of orthogonality, and it could be efficiently applied to nonlinear systems without such a property. Finally, let's mention that this group of vectors is not used independently.

7.2.5. Selection based on data about the model

Another strategy for generating coordinate vectors is to use some specific data about the model that is being solved. The approximate geometry and simplified model properties are frequently used. Often, it is sufficient to use only a problem that is in some way similar. For example, less important degrees of freedom can be excluded; the same geometry and element mesh can be used, but with simpler distribution of stiffness; hierarchical behaviour of a complex model can be used (as in manual calculation); coarse finite element mesh can be applied, etc. Generally, the displacements of these models under residual load can be used as appropriate coordinate vectors. This is most often associated with crude vectors that are most effective in the beginning of calculation. Later on, the solution needs to be smoothened, and it is better to use some of the more accurate approaches, described earlier. Nevertheless, such vectors can ensure very fast solution of many concrete problems, but the generation process has additional difficulty: lack of generality. Each type of equation requires separate approach.

For example, the substitute model for a thick beam can be a traditional thin beam, while a membrane could be used instead of a shell. If the solution of the substitute model $\mathbf{u}_z$ is known, then the coordinate vector is $\boldsymbol{\phi} = \mathbf{N}\mathbf{u}_z$, where **N** is the matrix of interpolation functions that connect degrees of freedom of default and substitute models. In the case of a beam, the substitute model is based on line elements (and can be simply supported), and the default model is defined by the mesh of planar finite elements (Figure 4).

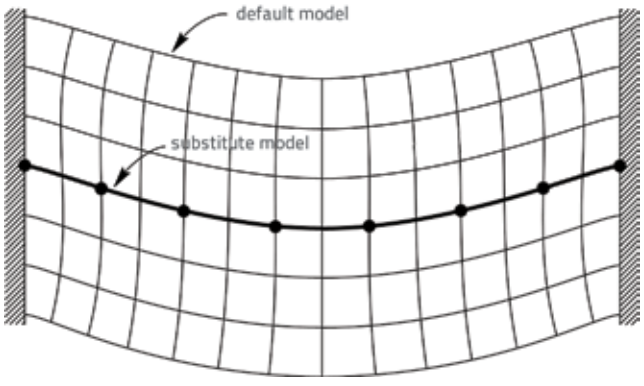

Figure 4. Model of a clamped beam

The nodes of both models that lie on the axis are connected by third-degree polynomials, and those of the default model (lying outside of the axis) are tied to the substitute model using cross section hypothesis of thin beam element. Columns of the matrix **N** are formed via polynomials and the hypothesis. The matrix defined in this way is singular, as displacements between models are linearly dependent, but the coordinate vector is correct and with the progress of the procedure leads toward solution for a thin beam. Of course, this solution is not good enough for original model. That is why a residual vector (or some other "correction") must be used as an additional coordinate vector to correct the assumption of straight cross sections, and provide a good solution of a high beam.

The iterative solution procedure based on the model stiffness hierarchy can also be used for formation of a coordinate vector. The coordinate vector can be defined as a solution after only several iterations between parts of the model with different stiffness. Obviously, this would only be a very crude approximation of the final result (which would be obtained after a larger number of steps), but it is nevertheless quite acceptable for fast definition of a coordinate vector.

Vectors can be generated with crude finite element mesh, using several iterations of a multigrid method [15, 16], but also through direct use of the analytically defined (continuous) coordinate functions over approximate domain of the model, which meet geometric boundary conditions. Notable examples of these functions are the Ritz functions and, as an interesting extension, R-functions [17]. Both can be multiplied by polynomials. Even more freely selected smooth functions that do not satisfy natural and geometric boundary conditions (here polynomials can once again be mentioned) can be used as coordinate vectors. Openings and similar irregularities do not need to be taken into account. In these cases, components of coordinate vectors are equal to the values of functions in the nodes. Obviously, such vectors are "rough", and the elements of the corresponding Ritz matrix are "excessively stiff", but can be used in the first steps of the procedure. However, if they are kept, the solution will be smoothened at later stages of the procedure, but the convergence will not be impressive. It would be better to somehow smooth these vectors in advance, possibly, for each one to use two cycles of overrelaxation with small , first with increasing and second with decreasing node numbering sequence.

Coordinate vectors can also be generated by means of analogy. For instance, if we are solving a slab problem, then we can use the solution to the problem of magnetic or electric field, torsion, membrane, grid, etc. The solution for the same slab with different load can also be applied, or even a roughly sketched displacement field. Then, measuring from the picture, we can determine displacements as vector components. All such solutions can be columns of the matrix $\boldsymbol{\phi}_i$.

Coordinate vectors can be used for an efficient definition of kinematic constraints. The hypothesis of straight cross sections of beam elements is one of such constraints. If we write them in the form of $\mathbf{Tu} = \mathbf{f}$ where $\mathbf{T}$ is the constraint matrix, then the initial approximation of the solution must satisfy nonhomogeneous constraints, i.e. $\mathbf{Tu}_0 = \mathbf{f}$, while coordinate vectors must satisfy homogeneous constraints only, i.e. $\mathbf{T\Phi}_i = \mathbf{0}$.

Let us finally mention the mixed approach to the generation of coordinate vectors. Thus the displacement increment $\Delta\mathbf{u}_i$ or the current approximation $\mathbf{u}_i$ is multiplied by some functions of coordinates. They can be Ritz functions or R-functions, and even polynomials, as mentioned above.

7.2.6. Coordinate vectors as input data

Finally, if someone finds a better set of (one or several) coordinate vectors, the matrix $\boldsymbol{\phi}_i$ could be formed without difficulties, and described procedure easily used. In this way, the proposed algorithm can be considered as a general approach to the iterative solution methods, where coordinate vectors are set as input data (in addition to the necessary ones required by all iterative methods).

8. Briefly about realization

The proposed pseudocode was realised using the programming language gfortran [18]. 64 bit Ubuntu version 5.3.1 and OS X version 6.1.0 were used. Once the program was verified on small equation systems, stiffness matrices and loads were generated for a considerable number of planar and space trusses. The formation of model was carried out in two ways. On the one hand, elements were placed in a traditional way, so that the trusses can form a good static system. On the other hand, to make condition number of the corresponding system much larger, we irregularly connected distant nodes by truss elements with large differences in stiffness values. In this way, we formed illogical trusses that cannot be regarded as structures. Thus, from the numerical viewpoint, we tested the program on both

Table 2. Basic data about numerical models

Figure No.		Number of nodes	Number of elements	Number of unknowns	Number of elements saved	Matrix fill rate
5.	Left	1.030.301	1.000.000	3.060.300	123.026.091	$1{,}31 \cdot 10^{-5}$
	Right	276.244	1.461.134	820.446	17.723.235	$2{,}63 \cdot 10^{-5}$
6.	Left	71.307	278.499	206.527	3.826.156	$8{,}97 \cdot 10^{-5}$
	Right	11.844	11.664	69.984	1.635.876	$3{,}34 \cdot 10^{-4}$
7.	Left	3.018.960	2.918.728	8.955.164	358.190.300	$4{,}47 \cdot 10^{-6}$
	Right	486	1.782	2.754	54.594	$7{,}20 \cdot 10^{-3}$

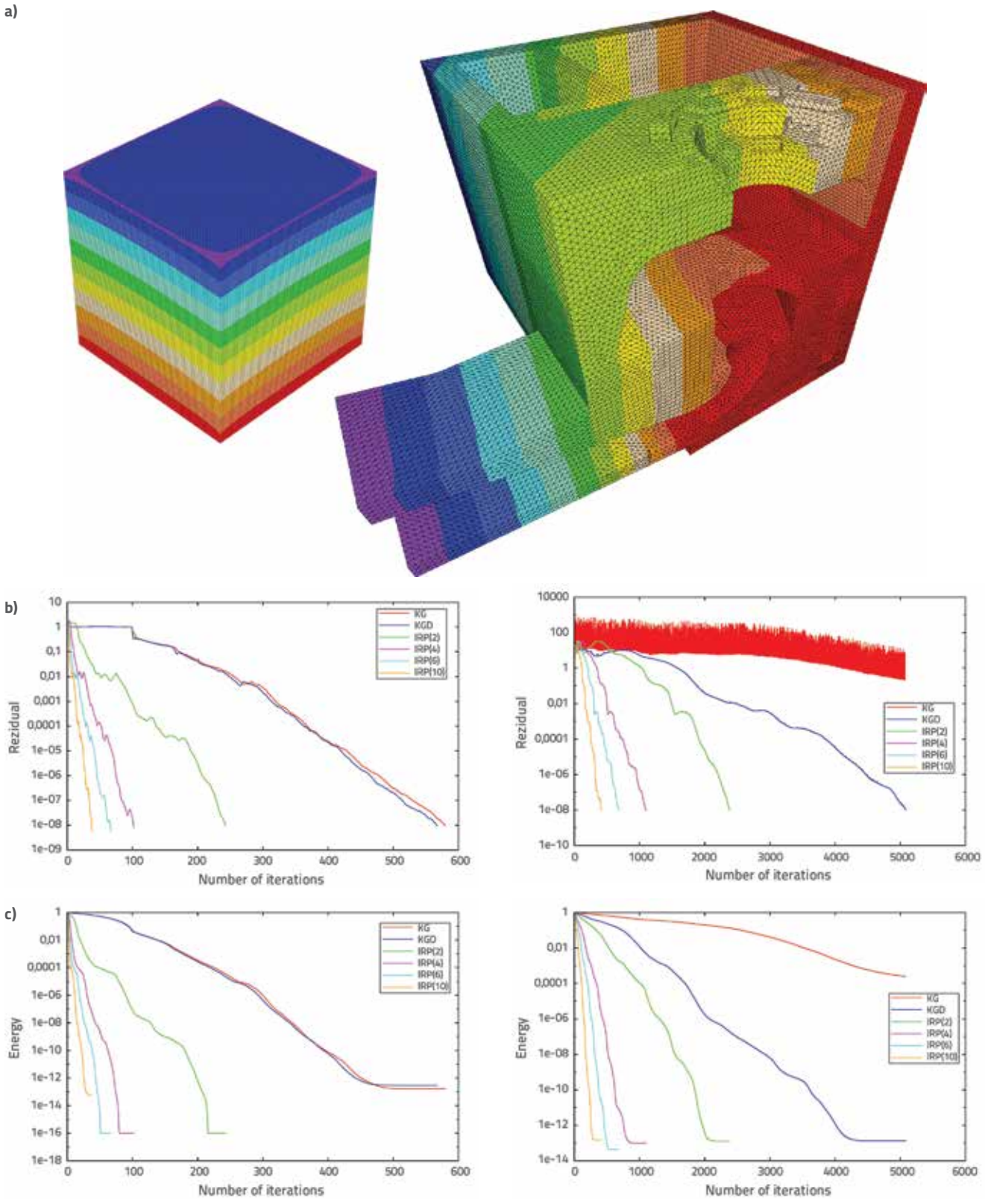

Figure 5. Cube model (left) and HPP Rama powerhouse model (right; only a half of the model is shown) [22]: a) distribution of vertical displacements; b) decrease of residual; c) decrease of energy

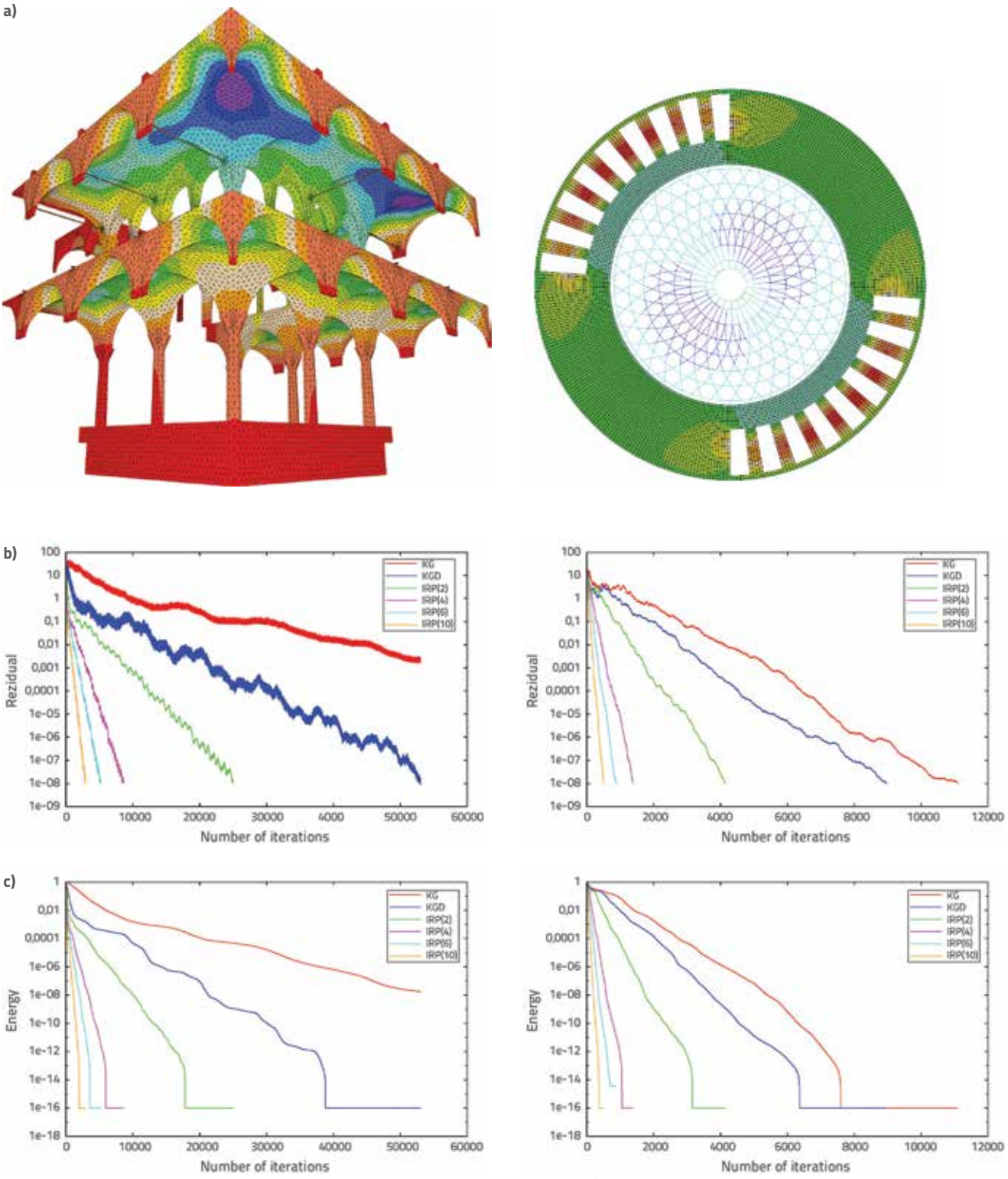

Figure 6. Models of the Rector's Palace atrium in Dubrovnik (left) [23] and Krešimir Ćosić Sports Hall dome in Zadar (right) [24]: a) distribution of vertical displacements; b) decrease of residual; c) decrease of energy

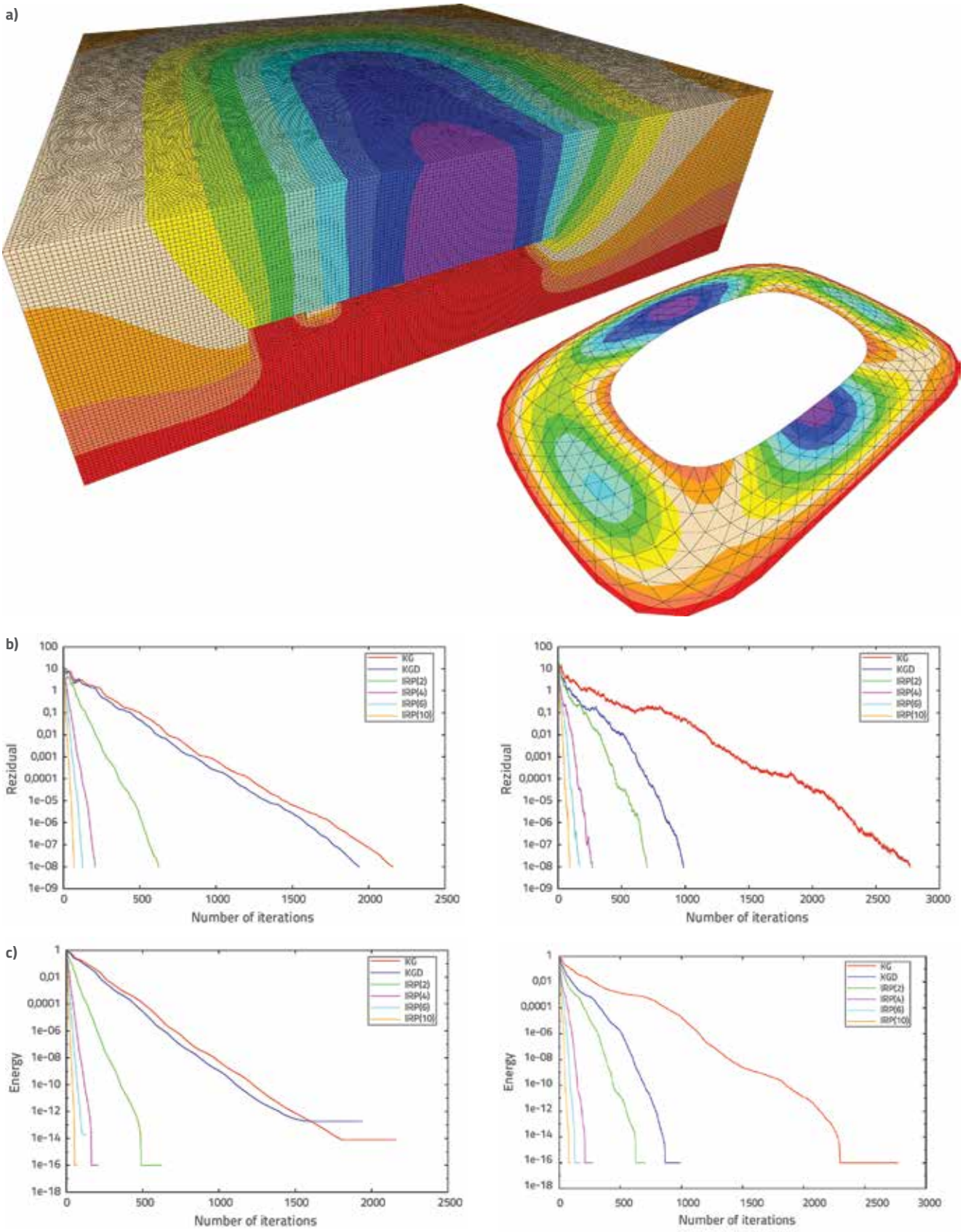

Figure 7. Models of the underground quarry in Kanfanar with the surrounding area (left; internal rooms and columns are not visible) [25] and roof structure of the future Kantrida Stadium in Rijeka (right) [26]: a) distribution of vertical displacements; b) decrease of residual; c) decrease of energy

Table 3. Comparison of methods according to the number of steps

Figure No.		Number of steps until convergence is achieved					
		KG	KGD	IRP(2)	IRP(4)	IRP(6)	IRP(10)
5.	Left	580	567	243	102	67	38
	Right	$>10^4$	5083	2381	1097	682	410
6.	Left	$>10^5$	53.002	24.995	8608	5166	2871
	Right	11 091	8966	4142	1381	888	498
7.	Left	2 157	1936	623	208	126	71
	Right	2 769	987	703	269	169	94

good and bad models. System matrix is saved sparsely, using full bookkeeping method by columns, and we also tried to do it by rows [19]. A very similar strategy also exists in the open source code program for the finite element method FEAP [20]. So, it was possible to easily connect our code with this package. The version 8.4.1 was used [21]. The compilation with the FEAP was also realized using the gfortran language.

9. Results of practical models

After fundamental checks, we analysed several models from structural engineering practice, on which we worked in previous years (Figures 5 to 7). For clarity reasons, the loads and supports are omitted from the figures. Models contain various types and shapes of finite elements. The unknowns are primarily displacements, although somewhere rotations are also present. Basic data about the models are given in Table 3. The fill rate of matrices is defined as the ratio between the number of elements saved and the number of all elements of the matrix.

Various checks of iterative algorithms are necessary as their dependence on the type of problem is fairly known. In other words, they can be adjusted for very fast calculation of typical examples, with the data known in advance, used to set key parameters of the method. However, the efficiency decreases as soon as the problem deviates from the expected. Figures show distribution of vertical displacements of the models [under a)], and dependence of the logarithm of the residual $\|\mathbf{r}_i\|_2/\|\mathbf{r}_0\|_2$ and energy ratio $(\Pi_0 - \sum_i \Delta\Pi_i)/\Pi_0$ on the number of iterations [under b) and c)]. Red and purple colours denote the area of the smallest and the greatest displacement, respectively. Calculations were made based on the iterated Ritz procedure (IRP) with two, four, six and ten coordinate vectors (argument of IRP), and using the method of conjugate gradients without preconditioning (KG) and with diagonal preconditioning (KGD). The following can be observed in figures b) and c): the number of iterations decrease with an increase in subspace, i.e. the reduction of residual and energy norm per step is larger. Even for two coordinate vectors, the method converges faster than KG and KGD (which can also be interpreted with two vectors). However, it has to be mentioned that there is a better preconditioning technique for the conjugate gradient method, e.g. by incomplete Cholesky factorization, although the results show that there is an adequate reserve for greater number of coordinate vectors (which has to be additionally checked). This is also confirmed by Table 3, with the number of steps needed for reducing the residual ratio to 10^{-8}. It should also be emphasized that the results from all examples are in good agreement with solutions obtained by direct and iterative methods used inside FEAP.

10. Conclusion

According to the described properties and our experience, the iterated Ritz method can be advantageous when solving large linear systems with sparse matrices. In some of our examples, as many as 10^{-7} unknowns were used, with the fill rate of approximately 10^{-6}. The explanation of the procedure is close to the engineering way of thinking, i.e. to the Ritz energy interpretation, unlike for instance the method of conjugate gradient that is normally explained geometrically, in the abstract -dimensional space. This procedure should not be worse than the conjugate gradient method (with and without preconditioning). In fact, a good subspace extension does not result in worse convergence. Additionally, several strategies for generating coordinate vectors can be used (as if several iterative methods are simultaneously applied). In case of a selection of appropriate vectors, the convergence is much faster compared to convergence based on an individual procedure. The preconditioning that transforms original system is not necessary here, but the algorithms developed for preconditioning can successfully be used in the generation of vectors. The restart known from the traditional method of conjugate gradients, by which the procedure is restarted due to loss of orthogonality, cannot be justified in this case. Only the displacement increment from the previous step is used, which improves the solution in the current step. As the orthogonality between iteration values is not required, additional advantages could be expected in the nonlinear problems. In such problems orthogonality properties, favoured by numerous iterative algorithms, are lost by definition. Therefore, the method can successfully be applied in the field of optimization as well.

It is clear that it would not be efficient to solve extremely large system using one processor only. In that case parallel approach is necessary. It would be interesting to use one processor for each coordinate vector. Because the proposed algorithm is (due to numerous possibilities) still in the intensive research phase (already a mere change in the amount of Ω and ω greatly influence the rate of convergence), currently there is no need to measure overall performance of the method and compare it to other direct and iterative solvers. This could only be done after careful programming and compilation of the final version of the program, including optimization options that contribute to the code efficiency. Therefore, this paper is more of methodological (algorithmic) than practical nature.

Acknowledgements

This work has been fully supported by Croatian Science Foundation under the project IP-2014-09-2899. We wish to thank Assistant Professor Mario Uroš PhD, who helped us to prepare numerical models.

REFERENCES

[1] Dvornik, J.: Generalization of the CG Method Applied to Linear and Nonlinear Problems, Computers & Structures, 10 (1979) 1/2, pp. 217-223.

[2] Varga, R.S.: Matrix Iterative Analysis, Second Edition, Springer Series in Computational Mechanics, Springer - Verlag, Berlin, 2000, https://doi.org/10.1007/978-3-642-05156-2

[3] Saad, Y., van der Vorst, H.: Iterative solution of linear systems in the 20th century, Journal of Computational and Applied Mathematics, 123 (2000), pp. 1-33.

[4] van der Vorst, H.A.: Iterative Krylov Methods for Large Linear Systems, Cambridge University Press, Cambridge, 2003, https://doi.org/10.1017/CBO9780511615115

[5] Liesen, J., Strakoš, Z.: Krylov Subspace Methods. Principles and Analysis, Oxford University Press, Oxford, 2013.

[6] Brezinski, C.: Multiparameter descent methods, Linear Algebra and its Applications, 296 (1999) 1-3, pp. 113.-141.

[7] Axelsson, O.: Iterative Solution Methods, Cambridge University Press, New York, 1994, https://doi.org/10.1017/CBO9780511624100

[8] Greenbaum, A.: Iterative Methods for Solving Linear Systems, Siam, Society for Industrial and Applied Mathematics (SIAM), Philadelphia, 1997, https://doi.org/10.1137/1.9781611970937

[9] Young, D.M.: Iterative Solution of Large Linear Systems, Dover Publications, Inc., Mineola, New York, 2003.

[10] Saad, Y.: Iterative Methods for Sparse Linear Systems, Society for Induppial and Applied Mathematics (SIAM), Philadelphia, 2003, https://doi.org/10.1137/1.9780898718003

[11] Olshanskii, M.A., Tyrtyshnikov, E.E.: Iterative Methods for Linear Systems. Theory and Application, Society for Industrial and Applied Mathematics (SIAM), Philadelphia, 2014, https://doi.org/10.1137/1.9781611973464

[12] Barrett, R., Berry, M., Chan, T.F., Demmel, J., Donato, J., Dongarra, J., Eijkhout, V., Pozo, R., Romine, C., van der Vorst, H.: Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods, Society for Industrial and Applied Mathematics (SIAM), Philadelphia, 1987.

[13] Benzi, M.: Preconditioning techniques for large linear systems: A survey, Journal of Computational Physics, 182 (2002), pp. 418-477.

[14] Korneev, V.G., Langer, U.: Domain Decomposition Methods and Preconditioning, u: Encyclopedia of Computational Mechanics (Editors: Erwin Stein, René de Borst, Thomas J. R. Hughes), Chapter 22, Volume 1: Fundamentals, John Wiley & Sons, Ltd, Chichester, 2004., pp. 617-649.

[15] Trottenberg, U., Oosterlee, C. W., Schüller, A.: Multigrid, Academic Press, London, 2001.

[16] Hackbusch, W.: Multigrid Methods for FEM and BEM Applications, Encyclopedia of Computational Mechanics (Editors: Erwin Stein, René de Borst, Thomas J. R. Hughes), Fundamentals, John Wiley & Sons, Ltd, Chichester, 1 (2004) 20, pp. 577-596.

[17] Dvornik, J.: Generiranje glatkih funkcija na složenom području pomoću R - funkcija, KoG, 1 (1996), pp. 27-30.

[18] The gfortran team: Using GNU Fortran, For GCC version 7.0.0 (pre - release), (GCC), Free Software Foundation, Boston, 2016.

[19] Nour - Omid, B., Taylor, R. L.: An Algorithm for Assembly of Stiffness Matrices into a Compacted Data Structure, Report No. UCB/SESM - 84/06, Structural Engineering and Structural Mechanics, Department of Civil Engineering, University of California, Berkeley, 1984.

[20] Taylor, R.L.: FEAP - A Finite Element Analysis Program, Version 8.4, User Manual, Department of Civil and Environmental Engineering, University of California at Berkeley, Berkeley, 2013.

[21] Taylor, R.L.: FEAP - A Finite Element Analysis Program, Version 8.4, Programmer Manual, Department of Civil and Environmental Engineering, University of California at Berkeley, Berkeley, 2014.

[22] Statički i dinamički proračun postojećih temelja agregata na nova opterećenja i djelovanja, knjiga G3, glavni projekt (projektant konstrukcije za ovu projektnu obradu Milan Crnogorac), Javno poduzeće Elektroprivreda hrvatske zajednice Herceg bosne d. d. Mostar, Zagreb, 2012.

[23] Lazarević, D., Dvornik, J., Fresl, K.: Analiza oštećenja atrija Kneževa dvora u Dubrovniku, GRAĐEVINAR, 56 (2004) 10, pp. 601-612.

[24] Lazarević, D., Anđelić, M., Uroš, M.: Oblikovanje i proračun kupole dvorane "Krešimir Ćosić" u Zadru, Građevinar, 62 (2010) 10, pp. 875.-886.

[25] Hrženjak, P., Petzel, M., Vujec, S.: Kontrolna mjerenja i numeričke analize za dimenzioniranje stupova i komora pri podzemnom otkopavanju arhitektonsko-građevnog kamena, Znanstvenostručno savjetovanje s međunarodnim sudjelovanjem "Mehanika stijena i tuneli", (urednici Jašarević, I., Hudec, M., Vujec, S.), 1 (1999), pp. 127-133.

[26] Uroš, M., Gidak, P., Lazarević, D.: Optimization of stadium roof structure using force density method, Proceedings of the third international conference on structures and architecture (ICSA2016) - Structures and Architecture - Beyond their Limits (editor Paulo J.S. Cruz), CRC Press/Balkema, Taylor & Francis Group, Guimaraes, 2016. pp. 693-700, https://doi.org/10.1201/b20891-95