DOI: 10.15439/2022F42
ISSN 2300-5963 ACSIS, Vol. 30

Block Subspace Iteration Method for Structural Analysis on Multicore Computers

Sergiy Fialko
Tadeusz Kosciuszko Cracow University of Technology
ul. Warszawska 24 St., 31 155 Kraków, Poland
Email: sergiy.fialko@gmail.com

***Abstract*—The block subspace iteration method for problems of structural dynamics oriented on multi-core computers is presented to extract the natural vibration frequencies and modes. The investigation is focused on multithreaded parallelization of all principal stages of the method allowing to determine up to several thousand eigenpairs even for design models with a lot of very close or multiple eigenfrequencies.**

I. Introduction

MODERN design finite-element models of buildings and structures usually have a dense spectrum of natural vibration frequencies. Mathematically, the problem is reduced to solving a partial generalized algebraic eigenvalue problem of a large dimension

$$\mathbf{Kv}_i - \lambda_i \mathbf{Mv}_i = 0, \ \ i \in [1, n] \ . \qquad (1)$$

Here **K** is a sparse symmetric positive definite stiffness matrix, **M** – diagonal or sparse symmetric semi-definite mass matrix, $\{\lambda_i, \mathbf{v}_i\}$ – eigenpair, N – dimension of the problem (1), n – number of required eigenpairs.

Requirements of seismic norms to determinate such a number of eigenpairs, which will provide a sufficiently high percentage of modal masses [1], [2] for each of the seismic input directions, leads to the fact that for many design models, with dimensionality, N = 2,000,000 ÷ 6,000,000 often has to obtain several thousands of eigenpairs [3].

The vast majority of such calculations are carried out by small and medium-sized design bureaus, so the main tool for numerical solutions of such problems are multi-core computers with shared memory, on which the solution of such problems by conventional methods can take from several hours to several days. Thus, increasing the performance of numerical methods for solving problem (1) for large design models with a large number of required eigenpairs is of great importance.

Both the Lanczos method and the subspace iteration method can be seen as varieties of the Arnoldi method [4]. However, within the framework of this article, we will confine ourselves to treating these methods as a kind of inverse matrix iteration method.

In the existing FEA software for the problems of structural dynamics, various versions of the block Lanczos method with spectral transformations [5], [6], etc. are very popular. The block version of the Lanczos method allows us to confidently solve problems for which there are a large number of multiple or almost multiple eigenfrequencies, in other words, the spectrum of eigenfrequencies has the area of condenses. Spectral transformations of type

$$\mathbf{Mv}_i - \frac{1}{\lambda_i - \sigma}(K - \sigma M)v_i = 0, \ \ i \in [1, n] \ , \qquad (2)$$

where σ - shift, first, allow us to better separate the close eigenfrequencies, which speeds up the convergence of the method, and secondly, divide the desired frequency interval to relatively small sub-intervals, limiting the dimension of the reduced problem on the Krylov subspace even in the case when we need to determine several thousand eigenpairs. At the same time, the maximum dimension of the reduced problem on the Krylov subspace does not depend on the number of required eigenpairs, which provides a quasi-linear computational complexity of the method instead of the quadratic computational complexity typical for versions of the method operating on the single frequency interval.

With the development of multi-core computers with shared memory, it turned out that existing implementations of the Lanczos method face difficulties in multi-threaded parallelization. One of the reasons that reduce the effectiveness of parallelization is the dimensionality of Krylov's subspace changing from step to step. Our observations have shown that while the dimension of the Krylov subspace is approximately within the first third of the maximum dimension in the current frequency interval, it is not possible to effectively use all the cores of the processor.

The disadvantage of a simple version of the subspace iteration method [1] (section 14-6) is the quadratic increase in the time of solving the problem with an increase in the number of required eigenpairs, which makes it practically unacceptable for the class of problems presented here.

The block version of the shifted subspace iteration method [7] corrects many shortcomings of the previous version of the algorithm. The essence of the method is that the iterations are performed in a block of a fixed dimension, which is much smaller than the number of required eigenpairs. As soon as converged eigenpairs appear in the block, the corresponding eigenvectors are placed in the special storage, excluded from the block, and in their place are created new start vectors that

This work was supported by IT Company SCAD Soft (www.scadsoft.com)

are linearly independent of each other and orthogonalized both to the remaining vectors in the block and to the previously converged eigenvectors. At each step of the method, all the vectors in the block are orthogonalized to the previously converged eigenvectors in order to avoid duplication of eigenpairs. At each step of the method, all the vectors in the block are orthogonalized to the previously converged eigenvectors in order to avoid duplication of eigenpairs. These orthogonalization procedures are a bottleneck that limits the use of this method for problems in which several thousand eigenpairs need to be extracted.

A similar drawback is the version of the method of conjugated gradients with preconditioning and spectral transformations [3].

In this article, we present a block version of the subspace iteration method with spectral transformations as an alternative to the block Lanczos method. The dimensionality of Krylov's subspace remains constant all the time, which makes it possible to effectively use the capabilities of modern multi-core computers. Spectral transformations accelerate the convergence of the method by dividing the close natural frequencies, and most importantly, they make it possible to divide the frequency interval into computationally independent sub-intervals and provide a quasi-linear dependence of the solving time of the problem on the number of required eigenpairs.

II. Parallel block subspace iteration method with a spectral transformations

A. Foundation

Before proceeding to the presentation of the proposed method, let us first give a simple algorithm of the subspace iteration method, corresponding to the one described in [1], but at the same time, we will take into account the impact of the shift σ. Let the approximation of the eigenvectors $\mathbf{v}_i^k$, forming a rectangular matrix be known at step k

$$\mathbf{Q}^k = \left\{\mathbf{v}_1^k, \mathbf{v}_2^k, \ldots, \mathbf{v}_m^k\right\}, \tag{3}$$

where

$$\left(\mathbf{v}_i^k\right)^T \mathbf{M}\mathbf{v}_j^k = \delta_{ij},$$
$$\left(\mathbf{v}_i^k\right)^T (\mathbf{K}-\sigma\mathbf{M})\mathbf{v}_j^k = \begin{cases} 0, i \neq j \\ \neq 0, i = j \end{cases}, \quad i, j \in [1, m], \tag{4}$$

δ_{ij} is a Kronecker symbol, m – block dimension (Krylov subspaces dimension). Perform an iteration step with the inverse matrix for the expression (2):

$$(\mathbf{K}-\sigma\mathbf{M})\overline{\mathbf{Q}}^{k+1} = \mathbf{M}\mathbf{Q}^k. \tag{5}$$

Matrix $\overline{\mathbf{Q}}^{k+1}$ contains improved approximations of eigenvectors compared to the matrix $\mathbf{Q}^k$, but $\mathbf{K} - \sigma\mathbf{M}$ and $\mathbf{M}$ orthogonality of vectors $\overline{\mathbf{v}}_i^{k+1}$, $i \in [1, m]$ is lost. Therefore, the next step is to orthogonalize the column vectors of the matrix $\overline{\mathbf{Q}}^{k+1}$:

$$\mathbf{v}_i^{k+1} = \sum_{j=1}^{m} q_{i,j}^{k+1} \overline{\mathbf{v}}_j^{k+1}, \quad i \in [1, m]. \tag{6}$$

Substituting (6) in (2), we get an algebraic generalized eigenvalue problem in the subspace S_m formed by the vectors $\overline{\mathbf{Q}}^{k+1}$:

$$\frac{1}{\lambda_i - \sigma}\left(\{\mathbf{k}\} - \sigma\{\mathbf{m}\}\right)\mathbf{q}_i - \{\mathbf{m}\}\mathbf{q}_i = 0, \quad i \in [1, m], \tag{7}$$

where $\{\mathbf{k}\} = \mathbf{Q}^T\mathbf{K}\mathbf{Q}$, $\{\mathbf{m}\} = \mathbf{Q}^T\mathbf{M}\mathbf{Q}$, $\mathbf{q}_i = \{q_{i,1}, q_{i,2}, \ldots, q_{i,m}\}^T$ is a vector of dimension m containing the expansion coefficients (6). Here, the upper subscript $k + 1$ is omitted for the sake of brevity. Problem (7) is equivalent to a simpler problem:

$$\{\mathbf{k}\}\mathbf{q}_i - \lambda_i\{\mathbf{m}\}\mathbf{q}_i = 0, \quad i \in [1, m], \tag{8}$$

which is solved by using the LAPACK procedures implemented in the Intel Math Kernel Library [8]. Having determined at step $k + 1$ the approximations of eigenvalues and eigenvectors in subspace S_m, we obtain approximations of eigenvectors (Ritz vectors) in the source space of dimension N:

$$\mathbf{Q}^{k+1} = \overline{\mathbf{Q}}^{k+1}\left\{\mathbf{q}^{k+1}\right\}. \tag{9}$$

where $\{\mathbf{q}\} = \{\mathbf{q}_1, \mathbf{q}_2, \ldots, \mathbf{q}_m\}$. This ensures the $\mathbf{K} - \sigma\mathbf{M}$ and $\mathbf{M}$ orthogonality of the vectors $\mathbf{v}_i^{k+1}, i \in [1, m]$, since

$$\begin{aligned}\left(\mathbf{v}_i^{k+1}\right)^T \mathbf{M}\mathbf{v}_j^{k+1} &= \left(\mathbf{q}_i^{k+1}\right)^T \left(\mathbf{Q}^{k+1}\right)^T \mathbf{M}\mathbf{Q}^{k+1}\mathbf{q}_i^{k+1} = \\ &= \left(\mathbf{q}_i^{k+1}\right)^T \left\{\mathbf{m}^{k+1}\right\}\mathbf{q}_i^{k+1} = \delta_{ij}\end{aligned}, \tag{10}$$

and

$$\begin{aligned}\left(\mathbf{v}_i^{k+1}\right)^T \mathbf{K}_\sigma\mathbf{v}_j^{k+1} &= \left(\mathbf{q}_i^{k+1}\right)^T \left(\mathbf{Q}^{k+1}\right)^T \mathbf{K}_\sigma\mathbf{Q}^{k+1}\mathbf{q}_i^{k+1} = \\ &= \left(\mathbf{q}_i^{k+1}\right)^T \left\{\mathbf{K}_\sigma^{k+1}\right\}\mathbf{q}_i^{k+1} = \begin{cases} 0, i \neq j \\ \neq 0, i = j \end{cases}\end{aligned}, \tag{11}$$

where $\mathbf{K}_\sigma = \mathbf{K} - \sigma\mathbf{M}$, $\{\mathbf{k}_\sigma\} = \{\mathbf{k}\} - \sigma\{\mathbf{m}\}$.

Here, the central moment that ensures the convergence of the method is the expression (5). Stages (6) to (9) provide the orthogonality of the improved Ritz vectors and prevent duplication of eigenpairs. Let's expand the expression (5) according to the eigenvectors of the problem (2) and multiply left by $\mathbf{v}_j^T$:

$$\sum_{i=1}^{m} \alpha_i^{k+1}(\mathbf{K}-\sigma\mathbf{M})\mathbf{v}_i = \sum_{i=1}^{m} \alpha_i^{k}\mathbf{M}\mathbf{v}_i \quad \cdot\Big|\mathbf{v}_j^T, \quad j \in [1, m]. \tag{12}$$

Here $\mathbf{v}_i^k = \sum_{i=1}^{m} \alpha_i^k \mathbf{v}_i$, $\mathbf{v}_i^k$ is an approximation of eigenvector $\mathbf{v}_i$ in the iteration step k. Taking into account the orthogonality of eigenvectors and the norming conditions with respect to the matrix $\mathbf{M}$, and also applying the expression for Rayleigh's quantity $\lambda_i = \mathbf{v}_i^T\mathbf{K}\mathbf{v}_i$ for the problem (1), we get:

$$\alpha_i^k = \left(\frac{1}{\lambda_i - \sigma}\right)^k \alpha_i^0, \quad i \in [1, m], \tag{13}$$

from which it follows that the closer the shift σ to the eigenvalue λ_i , the faster will be the convergence of the iterative process k = 1, 2, … to eigenpair {λ_i, $\mathbf{v}_i$}. Figure 1 presents a typical pattern of convergence of eigenvalues around the shift σ: the closer the eigenvalue λ_i is to the shift σ, the faster "on average" the Ritz pair {λ^k_i, $\mathbf{v}_i$} converges to the corresponding eigenpair {λ_i, $\mathbf{v}_i$} with the required precision.

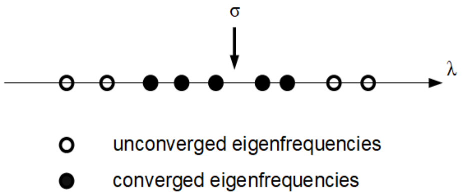

Fig. 1 Typical convergence of eigenfrequencies for methods based on shifted inverse iterations

The phrase "on average" should be understood in the statistical sense, since the rate of convergence depends not only on the difference $\lambda_i - \sigma$, but also on the selection of the initial approximation determined by α_i^0 coefficients. Thus, with the help of a proper selection of the shift value σ, we can control the position of the "center of convergence" on the axis λ. The information provided is well known and is presented here to facilitate understanding of the further presentation.

B. Block subspace iteration method

The disadvantage of the algorithm outlined above is that, with an increase in the number of required eigenpairs, the dimension m of the subspace increases, which leads to a rapid increase in the duration of such an analysis. To overcome this drawback, we divide the frequency interval into the subintervals, keeping a relatively small value of the parameter m, and also use multi-threaded parallelization. Unfortunately, the amount of RAM of modern multi-core computers does not allow us to concurrently solve the plurality of problems (2) for different shift values σ, as is done for distributed memory systems, so we parallelize the separate stages of the method.

The following Algorithm 1 presents the proposed approach. Step 1 performs the starting initialization of the method – it creates m orthogonal and **M** – orthonormal vectors forming the rectangular matrix $\mathbf{Q}^0$. *LeftMark* and *Rightmark* correspond to the left and right borders of the subinterval on the axis λ (Fig. 2), and *no_conv_modes* means the number of converged eigenpairs.

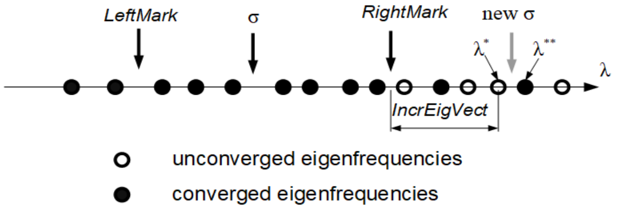

Fig. 2 Next frequency subinterval preparation

Algorithm 1. General algorithm of the block subspace iteration method with spectral transformation.

```
1.  Creation of m linearly independent vectors $\mathbf{Q}^0$
    according (3). Set $k$ = 1; no_conv_modes = 0;
    LeftMark = RightMark = 0; σ = 0.
2.  while no_conv_modes < nModes do
3.      Check status of vectors $\mathbf{Q}^k$ in block.
4.      Inverse iteration step (5).
5.      Create subspace matrices $\{\mathbf{k}\}^k$ and $\{\mathbf{m}\}^k$.
6.      Solve reduced eigenvalue problem (8).
7.      Calculate new Ritz vectors (9).
8.      k++.
9.  end while
```

Algorithm 2. Check status of vectors $\mathbf{Q}^k$ in block

```
1.  parallel for i = 1; i <= m; ++i do
2.      $\mathbf{r}_i^k = \mathbf{K}\mathbf{v}_i^k - \lambda_i^k\mathbf{M}\mathbf{v}_i^k$
3.      if $\|\mathbf{r}_i^k\|_2 / \|\lambda_i^k\mathbf{M}\mathbf{v}_i^k\|_2 < tol$
4.          $\mathrm{conv}_i$ = true;
5.      else
6.          $\mathrm{conv}_i$ = false;
7.  end of parallel for
8.  SetShiftProc();
9.  if $\sigma_{new} == \sigma$ then
10.     return;
11. else
12.     $\sigma = \sigma_{new}$;
13. end if
14. $\forall\lambda_i \in$ [LeftMark, RightMark] store {$\lambda_i$, $\mathbf{v}_i$} as a final
    results. Put: list_new_vect ← i.
15. LeftMark = RightMark;
16. Parallel for i ∈ list_new_vect do
17.     Generate new start vectors instead of stored
        eigenvectors, orthogonalize them against all
        remaining vectors in the block and normalize
        $\mathbf{v}_i^T\mathbf{M}\mathbf{v}_i = 1$.
18. end of parallel for
```

Loop **while** (steps 2 – 9) works until the number of required eigenpairs *nModes* are defined.

In step 3, the status of vectors in the block is checked - Algorithm 2. In a parallel loop **for** (steps 1 – 7), the residual vector $\mathbf{r}_i^k$ is determined. If the condition of step 3 is satisfied, this means that this Ritz pair has been converged with the required precision set by the *tol* parameter, and the i^{th} element of the *conv* array is assigned to *true*.

The *SetShiftProc*() procedure (step 8) calculates the number of converged eigenpairs, starting with *LeftMark* until the first non-converged Ritz pair meets, and the position *RightMark* (Fig. 2) is determined. Thus, segment [*LeftMark*, *RightMark*] contains only eigenvalues for converged eigenpairs. The transition to a new subinterval (change in the value of the shift σ) is carried out when the following two conditions are satisfied:

- The number of converged eigenpairs on the segment [*LeftMark*, σ] will be equal to *NoNegSignes* – *no_conv_modes*. Here, *NoNegSignes* is the number of sign changes on the main diagonal of the factorized matrix **K** – σ**M** (the number of eigenvalues enclosed in the interval from zero to σ), and *no_conv_modes* is the number of converging eigenpairs, which should be equal to the number of eigenvalues enclosed in the interval from zero to *LeftMark* (Sylvester's theorem of inertia).
- The number of converged eigenpairs on the [*LeftMark*, *RightMark*] segment is not less than the specified *IncrEigVect* value.

The first condition ensures that there are no skipped eigenfrequencies during the transition from one subinterval to another, and the second is necessary to ensure a sufficient number of iterations of the method for reliable prediction of the *RightMark* value.

As soon as the above conditions are satisfied, a transition to a new subinterval is carried out: a new shift value is calculated as $\sigma_{new} = \left(\lambda^* + \lambda^{**}\right)/2$, where λ^* is an approximation of eigenvalue locating on a value *IncrEigVect* from rightmost converged eigenvalue belonging to segment [*LeftMark*, *RightMark*], and λ^{**} is the next eigenvalue – see Fig. 2. To ensure the computational stability of the proposed algorithm, the prediction λ^{**} must be made with sufficient accuracy, which depends on the dimension of the block *m* and the value of the *IncrEigVect* parameter. The recommended values of these parameters are discussed in section III, *E*.

If at least one of the above conditions is not fulfilled, the shift value does not change, there is no transition to a new subinterval, the exit from Algorithm 2 (step 9) is performed and the iterations in this $\mathbf{Q}^k$ block continue. Otherwise, the σ shift value is changed and the transition to a new frequency interval is made (step 12). All converged eigenpairs in the [*LeftMark*, *RightMark*] segment are the final result and are placed in special storage on disk (step 14) and the *LeftMark* value is reset (step 15). Then, in a parallel region (steps 16 – 18), new starting vectors are generated in the addresses of converged eigenvectors, orthogonalized against themselves, as well as against the remaining vectors in the block and normalized, after which we proceed to step 4 of Algorithm 1.

If at this step *k* there is a change in the magnitude of the shift σ, the factorization of the matrix **K** – σ**M** is performed. Otherwise, the lower triangular matrix with the previous factorization is used. Forward-back substitutions are then performed. Multithreaded parallelization is used when calculating the rectangular matrix of the right parts (5) is performed. To factorize the matrix, the PARFES solver [9] designed specifically for multi-core computers with shared memory is used. Also, a parallel method [10] is applied to perform forward-back substitutions.

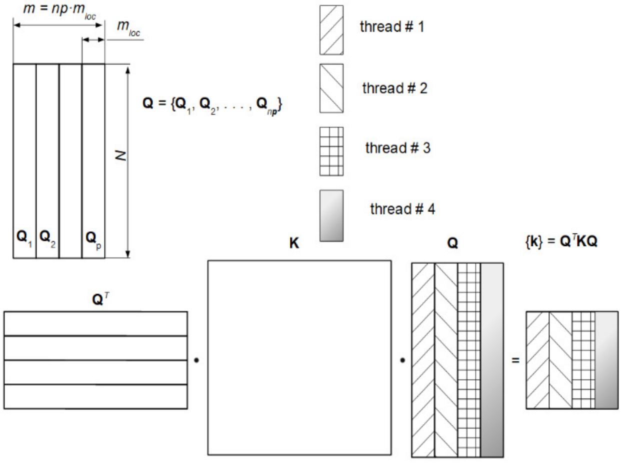

Fig. 3 Subdivision the matrix **Q** between threads

Calculation of projection matrices $\{\mathbf{k}\}^k = (\mathbf{Q}^k)^T\,\mathbf{K}\,\mathbf{Q}^k$ and $\{\mathbf{m}\}^k = (\mathbf{Q}^k)^T\,\mathbf{M}\,\mathbf{Q}^k$ in subspace $S_m = \text{span}\{\mathbf{v}_i^k \in \mathbf{Q}^k\}$ (Algorithm 1, step 5), is produced using multithreading.
To do this, the matrix **Q** (the iteration number *k* is omitted) is divided into blocks $\mathbf{Q}_1, \mathbf{Q}_2, \ldots, \mathbf{Q}_{np}$, where *np* is the number of threads and $m_{loc} = m\,/np$ is the number of vectors in each block $\mathbf{Q}_p$, $p \in [1, np]$ (Fig. 3). If the dimension *m* of the iterated block **Q** is not multiple to the number of threads, then for the last block is taken $m_{loc} = m - (np - 1)\cdot m/np$, where $(np - 1)\cdot m/np$ is the integer part of this expression. The mapping of blocks $\mathbf{Q}_1, \mathbf{Q}_2, \ldots, \mathbf{Q}_{np}$ onto threads is shown in Fig. 3. Each thread performs a task:

$$\{\mathbf{k}_{ip}\} = \mathbf{Q}^T\mathbf{K}\mathbf{Q}_{ip}, \quad \{\mathbf{m}_{ip}\} = \mathbf{Q}^T\mathbf{M}\mathbf{Q}_{ip}, \quad ip \in [1, np]. \qquad (14)$$

First, the sparse matrix **K** is multiplied by a dense rectangular matrix $\mathbf{B}_{ip} = \mathbf{K}\cdot\mathbf{Q}_{ip}$, and then $\{\mathbf{k}_{ip}\} = \mathbf{Q}^T\mathbf{B}_{ip}$ is calculated. Similarly, the dense matrix $\{\mathbf{m}_{ip}\}$ is evaluated.

The generalized eigenvalue problem (8) in the S_m subspace (Algorithm 1, step 6) is solved using the LAPACK procedure of the Intel MKL library [8], after which the vectors $\mathbf{v}_i^{k+1}$, $i \in [1, m]$ are determined (9), using a parallel version of the *dgemm* procedure from the Intel MKL library (Algorithm 1, step 7).

III. Numeric Results

Numerical results have been obtained on computer with 12-core Intel® Core™ i9–9920X CPU 3.50 GHz processor, 128 GB RAM, 64-bit Windows 10 Pro OS. This processor supports SIMD instructions AVX512F and FMA, has 512-bit registers that allow loading eight double words and simultaneously perform 8 multiplications and 8 additions.

The main attention is paid to the analysis of the time of solving the problem and the acceleration of the main stages of the proposed approach with an increase in the number of threads. The comparison is made with the block Lanczos method with spectral transformations [6], developed for the SCAD FEA software [11] and using the same PARFES solver to solve systems of linear algebraic equations with sparse symmetric matrices as the proposed approach. In addition, a multi-threaded parallelization of all the main stages of the Lanzos method was performed.

Design models are taken from the collection of problems of SCAD Soft – IT company, the developer of SCAD, one of the most widespread FEA software for the analysis and design of building structures in CIS countries, which has a certificate of compliance with regional building codes and a license for use in the design of nuclear power facilities.

A. Example 1

Figure 4 presents a design model of multistorey building comprising 4,262,958 equations. 500 eigenpairs are extracted. Such number of eigenpairs ensures the sum of modal masses 90% for each seismic input direction and satisfies requirements of the seismic building codes.

Dimension of block is accepted as $m = 96$ and parameter – $IncrEigVect = 15$. Table I depicts the total duration for the entire time of solving the problem of each basis stage of the proposed parallel block subspace iteration method (PBSI) depending on the number of threads. The number of threads does not exceed the number of physical core. This used thread binding to logical processors, where only one thread runs on each physical core. With the exclusive use of a computer for only one computing task, this strategy allows us to achieve the greatest performance and speed up the method while increasing the number of threads.

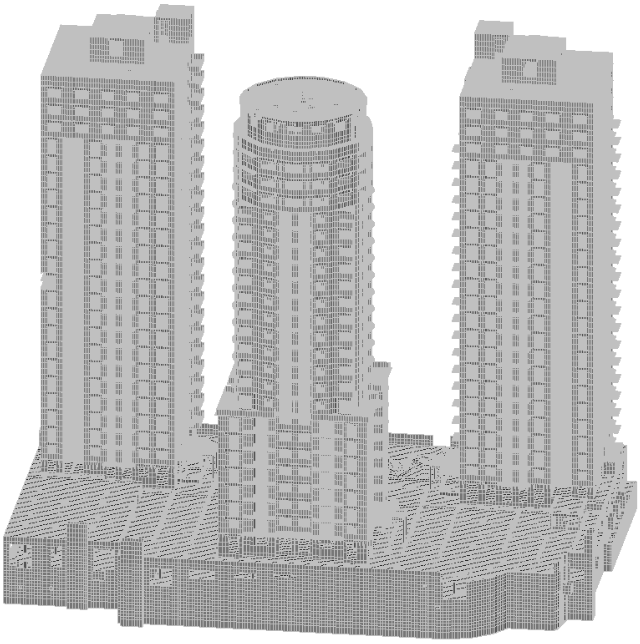

Fig. 4 Multistorey building 4,262,958 equations.

TABLE I.
DURATION OF THE BASIS STAGES OF THE PARALLEL BLOCK SUBSPACE ITERATION METHOD, S

# threads	Check status of $\mathbf{Q}^k$ vect.	Factorization	Resolution	Subspace matrices	Improved Ritz vectors	Rest	Total
1	1674	4744	5358	1719	284	1568	15347
2	1028	2386	3014	902	191	802	8323
4	638	1259	1770	491	123	409	4689
6	542	878	1342	365	105	298	3531
8	501	711	1159	310	101	242	3024
10	488	631	1024	275	92	203	2713
12	461	544	1013	262	98	206	2584
S_{12}	3.63	8.73	5.23	6.56	2.89	7.61	5.94

TABLE II.
DURATION OF THE BASIC STAGES OF THE BLOCK LANCZOS METHOD WITH SPECTRAL TRANSFORMATIONS, S

# threads	Generation of Lanczos vectors	Reorthogonalization	Subspace eigenvalue problem	Check of precision	Total
1	7516	3769	1230	319	13264
2	4445	2225	1238	235	8196
4	2741	1310	1227	214	5472
6	2655	1374	1218	189	5029
8	2222	945	1229	192	4480
10	2208	936	1224	180	4360
12	2194	937	1256	198	4343
S_{12}	3.43	4.02	0.98	1.61	3.05

The column «Check status of $\mathbf{Q}^k$ vect.» shows the duration of step 3 of Algorithm 1, the column «Factorization» – shows the duration of factorization of the sparse matrix $\mathbf{K} - \sigma\mathbf{M}$ by the PARFES solver [9], and the column «Resolution» depicts the time spent on performing multiplication of the mass matrix $\mathbf{M}$ by a rectangular matrix $\mathbf{Q}^k$ and on forward-back substitutions (5). At least for the diagonal mass matrix used in this problem, the main time of this step is forward-back substitutions, since the calculation of $\mathbf{MQ}^k$ is performed in about 5 seconds. The column «Subspace matrices» presents the time of evaluation of matrices $\{\mathbf{k}\}$ and $\{\mathbf{m}\}$ (step 5, Algorithm 1). The column «Improved Ritz vectors» depicts the time of the improved Ritz vectors evaluation $\mathbf{Q}^{k+1}$ (step 7, Algorithm 1), the column «Rest» demonstrates the time of all remaining operations which are parallelized too, and column «Total» – total time for solution of problem. Reduced generalized eigenvalue problem on subspace S_m (step 6, Algorithm 1) is solved in sequential mode, and the total duration of this stage for the given problem is about 0.2 s.

The last row shows the speedup for 12 threads: $S_{12} = T_1/T_{12}$, where T_1 is the total time of this stage on one thread, and T_{12} is on 12 threads.

The total speedup of the PBSI method is 5.94 on 12 threads, with the highest speedup achieved at the matrix factorization stage and the lowest – at the improved Ritz vector calculation stage. Taking into account that the duration of this stage is about an order of magnitude less than the duration of forward and back substitutions, we conclude that the buttlenecks of the proposed method are the stages "Check status of $\mathbf{Q}^k$ vect." and forward-back substitutions, having a speedup 3.63 and 5.23, respectively.

The forward and back substitution algorithms are presented in details in [10]. It also shows that the greater the number of right-hand sides, the higher the speedup of this algorithm with increasing the thread number. In the block Lanczos method [6], we usually have 7 right-hand sides and speedup does not exceed 2.5. The proposed approach has much more than 7 right-hand sides, so the speedup of forward-back substitutions is higher compared with the block Lanczos method.

Table II shows the results for the block Lanczos method with spectral transformations. The basis stages here are "Generation of Lanczos vectors", "Reorthogonalization", "Subspace eigenvalue problem", "Check of precision". Factorization of the $\mathbf{K} - \sigma\mathbf{M}$ matrix is performed with each change in the shift σ. The factorization time of the matrix and the time of forward and back substitutions are included in the total time of the "Generation of Lanczos vectors" stage. The "Subspace eigenvalue problem" stage is executed on a single thread. The total speedup of the block Lanczos method was almost two times worse than the PBSI method, and the total duration of the solution on 12 threads was 1.7 times longer.

B. Example 2

Figure 5 presents a design model of a shopping and entertainment center (TRK), containing 2,442,846 equations. Here we use an abbreviator TRK in the original (Ukrainian) language.

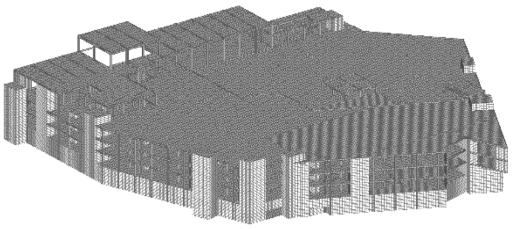

Fig. 5 Design model of a shopping and entertainment center (TRK), 2,442,846 equations.

Unlike the previous one, this design model, due to its low height, has much greater rigidity in the horizontal direction, which leads to slow convergence of the corresponding sums of modal masses. To achieve 90% of the sums of modal masses for each of the directions of seismic excitation, it was necessary to extract 2270 eigen pairs. At many parts, the natural vibration frequency spectrum undergoes condensation – Table III. In red color, especially close eigenfrequencies are highlighted.

For the PBSI method, the following parameter values are accepted: $m = 96$, *IncrEigVect* = 15. The duration of solving the problem on 12 threads is 6,729 seconds.

The time to solve this problem by the block Lanczos method is 11,883 s. Thus, the proposed PBSI method was 1.77 times faster than the block Lanczos method.

TABLE III.
FRAGMENT OF EIGENFREQUENCIES SPECTRUM

# mode	λ	ω = √λ 1/s	f = ω / (2π) Hz
391	1418.3925	37.6616	5.9940
392	1421.5990	37.7041	6.0008
393	1422.8605	37.7208	6.0035
394	1426.9157	37.7745	6.0120
395	1427.4229	37.7813	6.0131
396	1428.8853	37.8006	6.0162
397	1429.2057	37.8048	6.0168
398	1430.2163	37.8182	6.0190
399	1432.9059	37.8537	6.0246
400	1433.9266	37.8672	6.0268
401	1437.4244	37.9134	6.0341
402	1439.1556	37.9362	6.0377
403	1439.8368	37.9452	6.0392
404	1441.4548	37.9665	6.0426
405	1444.9049	38.0119	6.0498
406	1445.5251	38.0201	6.0511
407	1450.2098	38.0816	6.0609
408	1450.8574	38.0901	6.0622
409	1451.5816	38.0996	6.0637
410	1452.5786	38.1127	6.0658
411	1454.6621	38.1400	6.0702
412	1456.4738	38.1638	6.0740
413	1459.8107	38.2075	6.0809
414	1462.4419	38.2419	6.0864
415	1464.1232	38.2639	6.0899

C. Example 3

Figure 6 shows a design model of an industrial building comprising 1,807,218 equations. The bearing system of such a structure is the cross walls and floors (Fig. 7), which generates a huge number of local natural vibration modes, which practically do not contribute anything to the sums of modal masses both in horizontal directions and in vertical. To achieve 90% of the sum of modal masses in all directions, 20,352 eigenpairs had to be determined for this task.

For the PBSI method, the following parameter values are accepted: $m = 96$, *IncrEigVect* = 15. The duration of solving the problem on 12 threads is 45,819 s. The duration of solving the same problem by the block Lanczos method is 61,910 s.

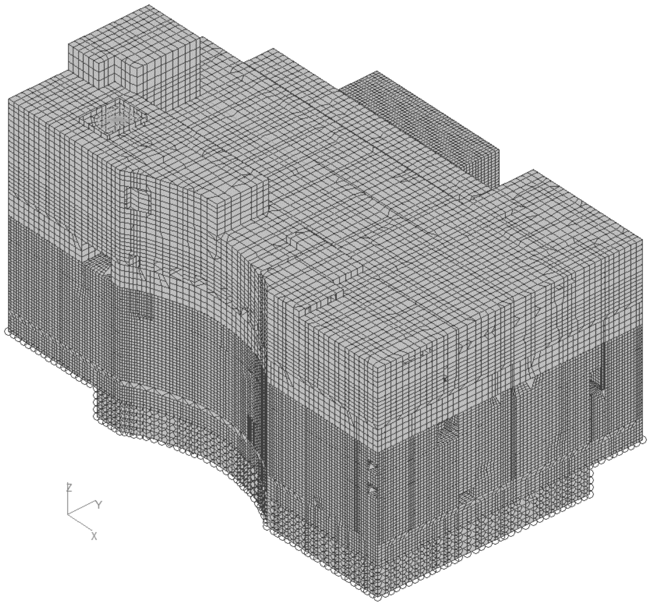

Fig. 6 Design model of industrial building – 1,807,218 equations.

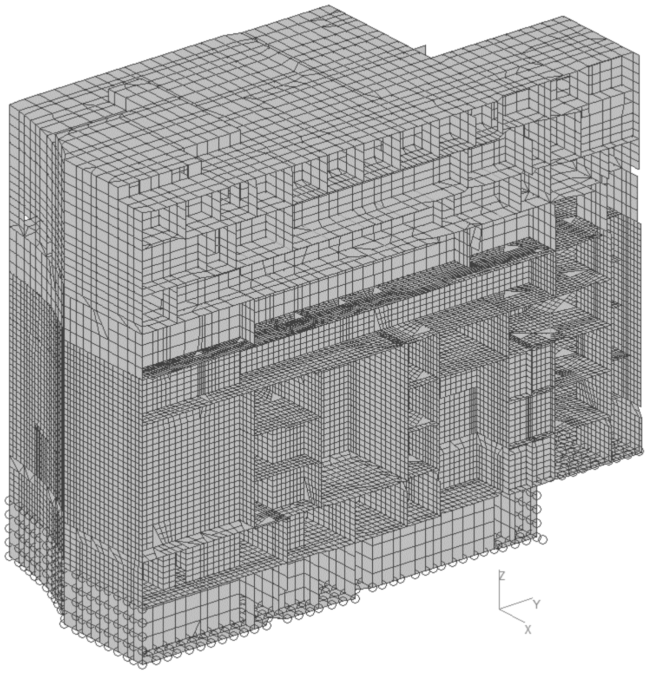

Fig. 7 Fragment of industrial building.

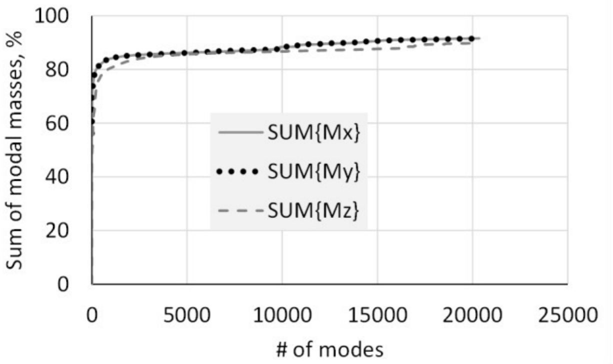

Fig. 8 Sum of modal masses in each seismic input direction

Fig. 8 shows the process of increasing the sums of modal masses with an increase in the number of eigenpairs taken into account in the modal analysis. Here SUM{Mx}, SUM{My}, SUM{Mz} are the sums of modal masses in directions OX, OY, and OZ correspondingly. This problem is an excellent test for checking the computational stability and reliability of methods for determining the frequencies and modes of natural oscillations since to achieve at least 90% of the sums of modal masses in the horizontal and vertical directions, it was necessary to extract more than 20,000 eigenvalue pairs.

D. *Example 4.*

Design model of multistorey building is shown in Fig. 9 and Fig. 10. This model comprises 2,002,428 equations and has a complex shape of the spatial configuration.

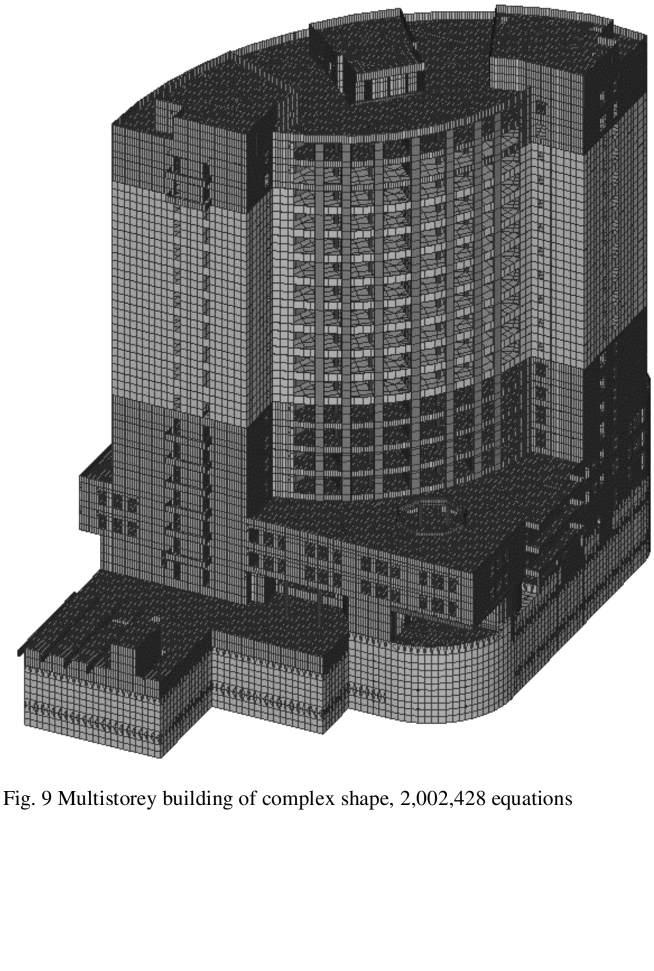

Fig. 9 Multistorey building of complex shape, 2,002,428 equations

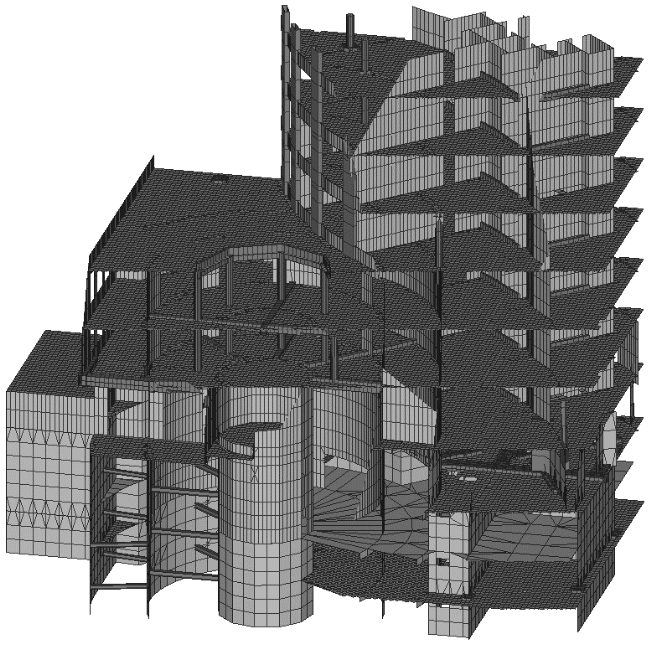

Fig. 10 Fragment of multistorey building of complex shape.

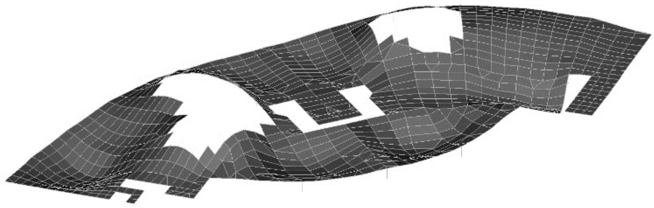

Fig. 11 Typical natural vibration mode for one from floor slabs.

A lot of local vibration modes (Fig. 11) making a very small contribution to the sum of modal masses turn out determination a large number of eigenpairs for seismic analysis – 1048 eigenpairs are required to obtain 90 % of the modal masses sum in horizontal directions and 75% in vertical (Fig. 12). This problem contains the condensation parts of the eigenvalue spectrum, one of which is presented in table IV. In red color, especially close eigenfrequencies are highlighted.

Fig. 11 demonstrates typical vertical vibrations of floor slabs. There are many such floors in the design model, so there are a large number of very close natural frequencies. In addition, the corresponding local forms of oscillation give a small contribution to the seismic response of the system, so it is necessary to determine a large number of eigenpairs to obtain reliable seismic response.

Fig. 12 demonstrates a slow increase of the sums of modal masses.

For the PBSI method, the following parameter values are accepted: m = 96, *IncrEigVect* = 15. The duration of solving the problem on 12 threads is 2,687 s. The duration of solving the same problem by the block Lanczos method is 4,200 s.

TABLE IV.
FRAGMENT OF EIGENFREQUENCIES SPECTRUM

# mode	λ	$\omega = \sqrt{\lambda}$ 1/s	$f = \omega / (2\pi)$ Hz
31	2530.432	50.3034	8.006
32	2536.1223	50.3599	8.015
33	2542.0002	50.4183	8.0243
34	2547.2958	50.4707	8.0327
35	2556.1409	50.5583	8.0466
36	2560.5829	50.6022	8.0536
37	2567.1267	50.6668	8.0639
38	2567.9224	50.6747	8.0651
39	2568.2671	50.6781	8.0657
40	2575.9373	50.7537	8.0777
41	2580.8602	50.8022	8.0854
42	2582.9832	50.8231	8.0887
43	2611.8944	51.1067	8.1339
44	2634.0931	51.3234	8.1684
45	2673.8288	51.7091	8.2298
46	2700.4928	51.9663	8.2707
47	2713.0356	52.0868	8.2899
48	2721.6674	52.1696	8.3031
49	2732.2792	52.2712	8.3192
50	2752.7392	52.4666	8.3503
51	2764.7959	52.5813	8.3686
52	2784.2313	52.7658	8.3979
53	2789.7365	52.818	8.4062
54	2791.1561	52.8314	8.4084
55	2808.2844	52.9932	8.4341
56	2809.4438	53.0042	8.4359
57	2836.3514	53.2574	8.4762
58	2844.4709	53.3336	8.4883

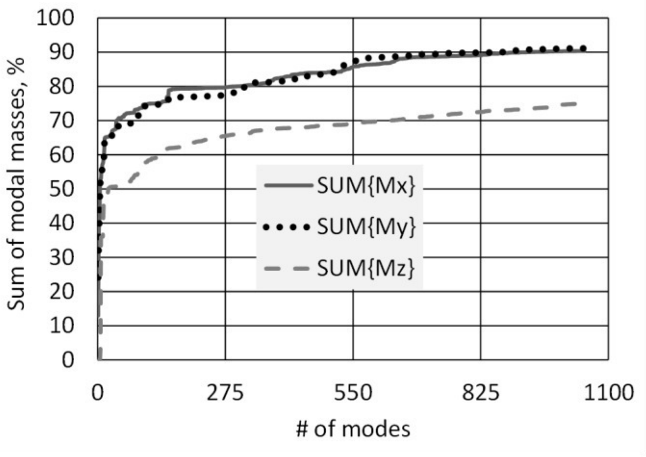

Fig. 12 Sum of modal masses in each seismic input direction

E. Reasoning about the parameters that control the convergence and numerical stability of the PBSI method.

The convergence rate and computational stability of the proposed method are determined by the values of the parameters m and *IncrEigVect*. It is desirable that the value of m be a multiple of the number of threads np. Then the number of vectors in each subblock $\mathbf{Q}_p$, $p \in [1, np]$ (Fig. 3) will be the same, which will favorably affect the balance of the computational load between threads. If the values of the parameter *m* are too small, the Ritz pairs located to the right of *RightMark* are determined with insufficient accuracy, which often leads to an unreliable forecast when choosing a new value of the shift of the new σ – Fig. 2. In this case, it may turn out that at the new frequency interval, the number of natural frequencies of *NoNegSignes – no_conv_modes* enclosed between the shift value σ and *LeftMark* – significantly exceeds the *IncrEigVect*, which often leads to a loss of convergence of the iterative process. Another danger of not having enough iterations in a given frequency interval is that there will be skipped some of the natural frequencies in the [*LeftMark*, *RightMark*] segment, and then the condition that number of converged eigenpairs on the segment [*LeftMark*, σ] must be equal to *NoNegSignes – no_conv_modes* (see the description of the SetShiftProc() procedure) will never be fulfilled – there will be an emergency interruption of the calculations after the control number of iterations exceeds.

Too small values of the *IncrEigVect* parameter lead to insufficient accuracy in determining Ritz pairs (too few iterations in a given frequency interval), and too large – to shift the new σ to the right boundary of the interval.

Testing of a large number of different tasks from the SCAD Soft collection showed that the values of the parameters m and *IncrEigVect*, close to optimal, are as follows: $m \in [96, 192]$, *IncrEigVect* $\in [10, 30]$. In this case, lower *m* values correspond to smaller *IncrEigVect* values.

It should be noted that when debugging the block Lanczos method with spectral transformations [6], we encountered similar problems when choosing the values of the parameters that ensure the computational stability and convergence of the method.

IV. Conclusion

The considered class of problems often requires determining a large number of natural vibration frequencies and modes to satisfy to requirements of seismic codes. In addition, in many cases, due to the presence of a large number of local oscillation modes, there are areas of condensation of the natural frequency spectrum. These features lead to the fact that the considered class of problems requires the development of effective numerical methods for their solution.

The parallel block method of subspace iteration proposed in this paper, designed to solve large-scale problems of determining the natural vibration modes and frequencies of buildings, structures, and deformable solids on multi-core computers with shared memory, demonstrates a shorter analysis time and greater speedup with an increase in the number of threads than the block Lanczos method with spectral transformations [6], which has been used in many industrial software over the years.

Acknowledgment

The author is deeply grateful to IT company SCAD Soft for providing a collection of real-life problems with the design models created by SCAD Office users.

References

[1] R. W. Clough, J. Penzien, *Dynamics of Structures*. Computers and Structures Inc., Berkeley, CA, USA, 2003, pp. 623 – 638.

[2] E.L. Wilson, *Three Dimensional Static and Dynamic Analysis of Structures*, Computers & Structures Inc., Berkeley, CA, USA, 2000, pp. pp. 13-1 – 13-16, 17-1 – 17-8. URL: http://www.edwilson.org/BOOK-Wilson/13-MODES.pdf. (Accessed 20 January 2022).

[3] S. Fialko, V. Karpilovskyi, “Block subspace projection preconditioned conjugate gradient method in modal structural analysis”, *Computers and Mathematics with Applications*, vol. 79, June 2020, pp. 3410–3428. https://doi.org/10.1016/j.camwa.2020.02.003.

[4] Y. Saad, *Numerical methods for large eigenvalue problems*, in: Classics in Applied Mathematics, Revised ed., SIAM, Philadelphia, 2011, http://dx.doi.org/10.1137/1.9781611970739.

[5] R. G. Grimes, J. G. Levis, H. D. Simon, “A shifted block Lanczos algorithm for solving sparse symmetric generalized eigenproblems”, *SIAM J. Matrix Anal. Appl*., Vol. 15, No. 1, January 1994, pp. 228-272. Doi: DOI:10.1137/S0895479888151111.

[6] S. Yu. Fialko, E. Z. Kriksunov, V. S. Karpilovskyy, “A block Lanczos method with spectral transformations for natural vibrations and seismic analysis of large structures in SCAD software”, in: Proceedings of the CMM-2003 – Computer Methods in Mechanics June 3–6, Gliwice, Poland, 2003, pp. 129–130. URL: https://scadsoft.com/download/049P.pdf (Accessed 20 January 2022).

[7] E. L. Wilson, T. Itoh, “An eigensolution strategy for large systems”, *Comput. Struct.*, vol. 16, Issues 1 – 4, 1983, pp. 259–265. https://doi.org/10.1016/0045-7949(83)90166-9.

[8] Intel MKL. URL: https://www.intel.com/content/www/us/en/develop/documentation/onemkl-developer-reference-c/top.html (Last access: 17.10.2021).

[9] S. Fialko. “Parallel finite element solver for multi-core computers with shared memory”, *Computers and Mathematics with Applications*, vol. 94, 2021, pp. 1 – 14. https://doi.org/10.1016/j.camwa.2021.04.013.

[10] S. Fialko. “Parallel algorithm for forward and back substitution in linear algebraic equations of finite element method”, Journal of telecommunications and information technology, vol. 4, 2019, pp. 20 – 29. https://doi.org/10.26636/jtit.2019.134919.

[11] V. S. Karpilovskyi, E. Z. Kryksunov, A. A. Maliarenko, S. Y. Fialko, A. V. Perelmuter, M. A. Perelmuter. *SCAD Office.V.21. System SCAD++*. Moscow: Publishing House SCAD SOFT, 2018. URL: https://scadsoft.com/download/SCAD1033.pdf.