

On a multigrid method for solving partial eigenproblems*

Maxim Larin[†]

Abstract

Recently the direct application of a multigrid technique for computing the smallest eigenvalue and its corresponding eigenvector of a large symmetric positive definite matrix A has been investigated in [5]. This method solves the eigenvalue problems on the sequence of nested grids using an interpolant of the solution on each grid as the initial guess for the next one and improving it by the full approximation scheme applied as an inner nonlinear multigrid method.

In the present paper the generalization of the method for computing few smallest eigenvalues and their corresponding eigenvectors of the elliptic self-adjoint operator is presented. Moreover, the quality of the method is improved by using the nonlinear Gauss-Seidel iteration instead its linearized version as pre- and postsmoothing steps. Finally, we give some practical advices for a good choice of multigrid-related parameters.

KEY WORDS multigrid methods, eigenvalue problems, sparse matrices.

1 Introduction

A lot of physical laws and governing processes can be formulated by means of partial differential equations. In practice, most such problems concern the elliptic self-adjoint boundary value problem

$$\mathbf{div}(k(x)\mathbf{grad} u(x)) = \lambda u(x),$$

in a bounded domain of R^d with appropriate boundary conditions. Here $d = 2$ or 3 is a spatial dimension. The coefficient matrix $k(x)$ denotes a symmetric positive definite second or fourth order tensor, which is frequently piecewise constant and/or highly anisotropic. The displacement equation of linear elasticity problems have the similar form, where $u(x)$ now denotes the displacement vector $\mathbf{u}(x) = (u_1, \dots, u_d)^T$ and the coefficient matrix $k(x)$ denotes the elasticity tensor. Below we will stay with the scalar case of (1), using it as a model for the vector case [6].

Consider the elliptic boundary value problem with homogeneous boundary conditions

$$-\sum_{i,j=1}^2 \frac{\partial}{\partial x_i} \left(k_{ij}(x) \frac{\partial}{\partial x_j} u(x) \right) = \lambda u(x), \quad x \in \Omega, \quad u(x) = g(x), \quad x \in \partial\Omega, \quad (1)$$

*The work is supported by grant RFBR 01-07-90367.

[†]Institut für Geometrie und Praktische Mathematik, RWTH Aachen, 52056 Aachen, Germany. E-mail: larin@igpm.rwth-aachen.de .

where Ω is a polygonal domain in R^3 with boundary $\partial\Omega$, where the coefficient matrix $[k_{ij}(x)]$ is assumed to be uniformly symmetric positive definite for any $x \in \overline{\Omega}$, and $g(x)$ is a given function. Although we assume Dirichlet boundary conditions, the results of this paper are valid for general boundary conditions. Under these assumption all eigenvalues λ of (1) are real and positive.

The Galerkin variational formulation of the boundary value problem (1) is as follows.

Find $\lambda_k \in \mathbf{R}$ and $u_k \in H_0^1(\Omega) = \{v \in H^1(\Omega) : v = 0 \text{ on } \partial\Omega\}$ such that

$$\begin{aligned} a(u_k, v) &= \lambda_k b(u_k, v), & k &= 1, 2, \dots, p, \dots \\ b(u_k, u_l) &= \delta_{kl}, & l &= 1, 2, \dots, p, \dots \end{aligned} \quad (2)$$

for all $v \in H_0^1(\Omega)$, where $H^1(\Omega)$ is the usual Sobolev space, δ_{ij} is the Kronecker symbol and bilinear forms $a(u, v)$ and $b(u, v)$ are defined by

$$a(u, v) = \int_{\Omega} \left[\sum_{i,j=1}^2 k_{ij}(x) \frac{\partial u}{\partial x_i} \frac{\partial v}{\partial x_j} \right] d\Omega, \quad b(u, v) = \int_{\Omega} uv \, d\Omega.$$

Let us assume that the domain Ω is decomposed by a set of finite elements Υ with a characteristic meshsize h . Usally, $h = O(n^{-1/d})$. Introducing $V_h \subset H^1(\Omega)$ or $V_h \subset [H^1(\Omega)]^3$ the space of vector functions with local support, associated with the vertices of Υ , and applying to (2) the standard Galerkin procedure, we obtain the following eigenvalue problem

$$\begin{aligned} A \mathbf{u}_k &= \lambda_k \mathbf{u}_k, & A &= A^T \in \mathbf{R}^{n \times n}, & 0 < \lambda_1 \leq \lambda_2 \leq \dots \leq \lambda_p, \\ (\mathbf{u}_k, \mathbf{u}_l) &= \delta_{kl}, & k, l &= 1, \dots, p, \end{aligned} \quad (3)$$

where the matrix A corresponds to the bilinear form $a(\cdot, \cdot)$ or $\hat{a}(\cdot, \cdot)$ and $(\mathbf{u}, \mathbf{v}) = \mathbf{u}^T \mathbf{v}$. Typically, the matrix A is large, sparse and ill-conditioned, i.e., its condition number goes to infinity as $O(h^{-2})$ when h tends to zero.

The classical methods for solving eigenvalue problems such as the inverse iteration and the Rayleigh quotient iteration [7] can not be applied for the large-scale finite element problems, since the computer storage for holding the typically denser LU or Cholesky factor is not available. On the other hand, the convergence rate of the Lanczos method [7] slows down considerably for decreasing h and is also unacceptable for real-life problems.

The natural way to overcome these difficulties is to use an approximate inverse method instead of the exact method for solving the linear system with A whenever it is required in the algorithm. See [3, 4] for a combination of preconditioning and the eigensolvers.

However, all of these methods are treating this eigenvalue problem as a purely algebraic, and hence, we loss some valuable information about the desired eigenpair. For example, the smallest eigenvector of the Laplace operator is very smooth, i.e., it can be well approximated on coarser grids, and hence, one can use this information during the solution process. The nested iteration multigrid process, which take the advantage of this smoothness, was proposed in [1] and futher investigated for the smallest eigenpair in [5]. In the present work the full multigrid method for finding p smallest eigenvalues and their eigenvectors or, shortly, the FMG-EV(p) method is presented. To improve the robustness of the relaxation steps the nonlinear Gauss-Seidel method is used instead its linearized version. Finally, an experimental study of this methods to understand the main points of the method having a great influence as on the accuracy of computed eigenpairs and as on its computational complexity.

In the next section the method for constructing the sequence of matrices is discussed. The FMG-EV(p) method for finding few smallest eigenpairs will be presented in the Section 3. Finally, the corresponding numerical results showing its robustness and conclusions are given.

2 Definition of $\{A^{(k)}\}$

In order to define a multilevel process we have to construct a sequence of symmetric positive definite matrices $\{A^{(k)}\}$, $k = 0, 1, \dots, L-1, L$ of an decreasing order n_k such that $A^{(0)} = A$. For last two decade many scientists concerns to the question for solving the linear system of equations: how to construct the *best* (or *optimal*) sequence of matrices $A^{(k)}$ (in terms of computational costs of the corresponding iterative process)? The problem was successfully solved for sufficiently wide number of industrial problems (see [8] and references therein). There are two major types of the methods, which are based on either *problem*-dependent geometrical information or *matrix*-dependent numerical information. The choice of the method to compute $\{A^{(k)}\}$ usually depends on the problem to be solved, and hence, is up to the user priority. However, in the case of the eigenvalue problem the situation is quite different.

Let $N_k \in \{1, \dots, n_k\}$ be the set of indices of all unknowns (or nodes) on the level k , $k \geq 0$ and is partitioned into two non-intersection subsets, e.g., $N_k^f \subset \{1, \dots, n_k\}$ and $N_k^c \subset \{1, \dots, n_k\}$ such that $N_k = N_k^c \cup N_k^f$ and $N_k^c \cap N_k^f = \emptyset$. This partitioning yields the following block matrix form of $A^{(k)}$

$$A^{(k)} = \begin{bmatrix} A_{11}^{(k)} & A_{12}^{(k)} \\ A_{21}^{(k)} & A_{22}^{(k)} \end{bmatrix} \begin{matrix} \} n_k \setminus n_{k+1} \\ \} n_{k+1} \end{matrix} .$$

where the first group of unknowns correspond to the indices in N_k^f and the second one forms a new set of indices on the next level, i.e., $N_{k+1} = N_k^c$. For the given partitioning, we define a prolongation matrix $P_{k+1}^k \in \mathbf{R}^{n_k \times n_{k+1}}$ of the following form

$$P_{k+1}^k = \begin{bmatrix} J_k \\ I_{k+1} \end{bmatrix} \begin{matrix} \} n_k \setminus n_{k+1} \\ \} n_{k+1} \end{matrix} ,$$

where I_{k+1} is an identity matrix of order n_{k+1} and the choice of blocks J_k depend on the method used for construction of the sequence of matrices $A^{(k)}$.

For example, the matrix-dependent methods are based on the Galerkin coarse-grid matrix formulation:

$$A^{(k+1)} = R_k^{k+1} A^{(k)} P_{k+1}^k, \quad R_k^{k+1} = (P_{k+1}^k)^T,$$

where B^T is a transpone of B and the block J_k in the definition of an interpolation matrix P_{k+1}^k is a sparse approximation of $-(A_{11}^{(k)})^{-1} A_{12}^{(k)}$. Assume "ideal" situation for iterative methods, i.e.

$$J_k = -(A_{11}^{(k)})^{-1} A_{12}^{(k)},$$

then we get

$$A^{(k+1)} = A_{22}^{(k)} - A_{21}^{(k)} (A_{11}^{(k)})^{-1} A_{12}^{(k)} .$$

and hence, we solve the following eigenproblem on the coarse level

$$A^{(k+1)} \mathbf{u}^{(k+1)} = \lambda^{(k+1)} \mathbf{u}^{(k+1)} \Leftrightarrow (A_{22}^{(k)} - A_{21}^{(k)} (A_{11}^{(k)})^{-1} A_{12}^{(k)}) \mathbf{u}_2^{(k)} = \lambda^{(k+1)} \mathbf{u}_2^{(k)} . \quad (4)$$

On the other hand, from

$$\begin{bmatrix} A_{11}^{(k)} & A_{12}^{(k)} \\ A_{21}^{(k)} & A_{22}^{(k)} \end{bmatrix} \begin{bmatrix} \mathbf{u}_1^{(k)} \\ \mathbf{u}_2^{(k)} \end{bmatrix} = \lambda^{(k)} \begin{bmatrix} \mathbf{u}_1^{(k)} \\ \mathbf{u}_2^{(k)} \end{bmatrix},$$

we obtain the *equivalent* eigenproblem on the coarse level

$$(A_{22}^{(k)} - A_{21}^{(k)}(A_{11}^{(k)} - \lambda^{(k)}I)^{-1}A_{12}^{(k)})\mathbf{u}_2^{(k)} = \lambda^{(k)}\mathbf{u}_2^{(k)},$$

which is *different* from (4). Thus, the direct application of matrix-dependent multilevel technique developed for solving the linear system of equations is not a good idea, and hence, the method for computing a good sequence of matrices $A^{(k)}$ is required.

From the first point of view the problem-dependent approach have to work much better than matrix-dependent one, since the problem information about the behaviour of the smallest eigenvectors are used (at least implicitly). Nevertheless, the smallest eigenvalue on two adjoint levels are still different due to different characteristic mesh sizes on different grids.

In the present paper following [1] we use the standard multigrid method, although any kind of multilevel techniques specially developed for eigenproblems can be used to construct that sequence of matrices.

Let us assume that the initial (coarse) triangulation Υ_L of the domain Ω is such that the coefficient matrix $k(x)$ is constant in each element $T_i \in \Upsilon_L$ and can be discontinuous across the elements. To obtain a sufficiently accurate solution of the problem (2) we make a uniform refinement procedure to construct a sequence of nested meshes (triangulations) $\Upsilon_L \subset \Upsilon_{L-1} \subset \dots \subset \Upsilon_1 \subset \Upsilon_0 = \Upsilon$. Introduce $V_k \subset H^1(\Omega)$ the space of linear vector functions, associated with the vertices of Υ_k . Now the Galerkin procedure applied to (2) on each grid Υ_k leads to the sequence of eigenvalue problems with the corresponding matrices $A^{(k)}$ of the decreasing order:

$$\begin{aligned} A^{(k)}\mathbf{u}_i^{(k)} &= \lambda_i\mathbf{u}_i^{(k)}, & A^{(k)} &= A^{(k)T} \in \mathbf{R}^{n \times n}, & 0 < \lambda_1^{(k)} &\leq \lambda_2^{(k)} \leq \dots \leq \lambda_p^{(k)}, \\ (\mathbf{u}_i^{(k)}, \mathbf{u}_j^{(k)}) &= \delta_{ij}, & i, j &= 1, \dots, p, & k &= 0, 1, \dots, L, \end{aligned} \quad (5)$$

where the matrix $A^{(k)}$ corresponds P_{k+1}^k to the bilinear form $a(\cdot, \cdot)$ on the level k .

For the given partitioning, coefficients for the interpolation of the value $\mathbf{u}^{(k)}$ from $\mathbf{u}^{(k+1)}$ are computed using the standard linear interpolation [2, 9]. Note that by the definition of P_{k+1}^k we have for $k < l$

$$P_l^k = P_{k+1}^k P_l^{k+1} = \dots = P_{k+1}^k P_{k+2}^{k+1} \dots P_{l-1}^{l-2} P_l^{l-1},$$

and vica-versa

$$R_k^l = R_{l-1}^l R_k^{l-1} = \dots = R_{l-1}^l R_{l-2}^{l-1} \dots R_{k+1}^{k+2} P_k^{k+1}.$$

3 The FMG-EV(p) method

The idea of the FMG-EV method for the first eigenpair based on the idea of the well-known full multigrid approach, i.e., we find some approximated solution of the analogous problem on the coarse level and using its interpolant as the initial guess for some inner iterative

process on the next (finer) one, see [1, 5] for details. Now there are several possibilities to define a multilevel method for computing the first p eigenvalues and their eigenvectors of A . There are two different approaches how to proceed eigenvectors through the multilevel process simultaneously or sequentially. Both of them have their advantages and disadvantages in terms accuracy of the eigenpairs computed, handling the orthogonalization conditions, the computational complexity and the amount of storage requirements.

Following [1] we use a combination of those approaches. Indeed we compute all eigenvectors simultaneously through outer nested iteration process followed by the Ritz projection [7] to improve the eigenvector approximations on the currently finest level, but we proceed vector by vector through the inner nonlinear multilevel method.

$\{\mu_i^{(0)}, \mathbf{v}_i^{(0)}\}_{i=1}^p = \text{FMG-EV}(p, A^{(L)}, \dots, A^{(0)}):$ $i = 0, \quad k = L, \quad \mu_{i-1}^{(k)} = 0$ <ol style="list-style-type: none"> 1 $i = i + 1$ Compute the coarse level approximation $\{\mu_i^{(k)}, \mathbf{v}_i^{(k)}\}$, which is orthogonal to computed ones $\{\mu_j^{(k)}, \mathbf{v}_j^{(k)}\}_{j=1}^{i-1}$ if $((i < cn_k) \text{ and } (i < p))$ go to 1 $\{\mu_j^{(k)}, \mathbf{v}_j^{(k)}\}_{j=1}^i = \text{Ritz}(A^{(0)}, \{\mathbf{v}_j^{(k)}\}_{j=1}^i)$ 2 if $(k = 0)$ stop $k = k - 1$ for $s = 1, \dots, i$ $\mathbf{v}_{s,0}^{(k)} = P_{k+1}^k \mathbf{v}_s^{(k+1)}$ $\mu_{s,0}^{(k)} = \mu_s^{(k+1)}$ for $it = 1, \dots, q$ (<i>inner iteration</i>) $\{\mu_{s,it}^{(k)}, \mathbf{v}_{s,it}^{(k)}\} = \text{FAS-EV}(A^{(k)}, \mu_{s,it-1}^{(k)}, \mathbf{v}_{s,it-1}^{(k)}, \{\mathbf{v}_j^{(k)}\}_{j=1}^{s-1})$ $\{\mu_j^{(k)}, \mathbf{v}_j^{(k)}\}_{j=1}^i = \text{Ritz}(A^{(0)}, \{\mathbf{v}_{j,q}^{(k)}\}_{j=1}^i)$ if $(i < p)$ then go to 1 else go to 2 <p>Here FAS-EV $(A, \mu_i, \mathbf{v}_i, \{\mathbf{v}_j\}_{j=1}^{i-1})$ denotes the inner V-cycle multigrid method applied for solving the eigenproblem $A\mathbf{u}_i = \lambda_i \mathbf{u}_i$ with an orthogonalization condition to $\{\mathbf{v}_j\}_{j=1}^{i-1}$ using $\{\mu_i, \mathbf{v}_i\}$ as initial guess, and Ritz $(A, \{\mathbf{v}_i\}_{i=1}^n)$ computes new eigenvalue and eigenvector approximations in the subspace spanned onto $\mathbf{v}_1, \dots, \mathbf{v}_n$ by the Ritz method.</p>
--

There are few hard points in this approach. First of all, there is no exact correspondence between eigenvectors on different levels, i.e., the i th eigenvalue and its eigenvector on the coarse level can be a *good* approximation to j th eigenpair on the fine level with $i \neq j$. It is not a problem if j is also less than p , and hence, during the following Ritz projection we find both of desired eigenvectors on the fine level. However, if j is great than p , then the further efforts to improve this approximation in the direction of i th eigenvector is vain since an inner nonlinear method works good only in a small neighbourhood of the solution.

Thus, on the coarse level we can find a fixed number of approximation vectors, which are a good approximation to desired eigenvectors. This number depend on the order of the coarse level problem, and hence, can be defined as cn_L , where c is a coefficient less than 1. Now if $p > cn_L$, then we find the coarse level approximations only for first cn_L eigenvectors and

start our nested iteration process with this smaller number of vectors. After transferring and processing these approximated eigenvectors on the next finer level we compute the deficient coarse level approximations using the relaxation steps followed by orthogonalization with respect to previous ones, and if p is still great than cn_{L-1} , then we repeat this process on the next finer level and so on. Note that the coarsest level used in the inner multilevel method for i th eigenvector is one on each $\mathbf{v}_i^{(k)}$ first appeared. The rest components of this method is similar to the FMG-EV method for the first eigenvalue [5].

Since we apply the inner nonlinear multilevel method sequentially to each approximated eigenvector, then the main algorithm for the inner multilevel method presented in [5] does not changed. Nevertheless, in addition to it we have to orthogonalize the current eigenvector approximation to previously computed ones.

$$\begin{aligned}
\{\mu_i^{(l)}, \mathbf{v}_i^{(l)}\} = & \text{FAS-EV} (A^{(l)}, \mu_{i,0}^{(l)}, \mathbf{v}_{i,0}^{(l)}, \{\mathbf{v}_j^{(k)}\}_{j=1}^{i-1}): \\
& \mathbf{b}_i^{(l)} = \mathbf{0} \\
& \text{for } k = l, \dots, L-1 \\
& \quad \text{for } it = 1, \dots, \nu_1 \quad (\text{presmoothing}) \\
& \quad \quad \{\mu_{i,it}^{(k)}, \mathbf{v}_{i,it}^{(k)}\} = \text{Relax} (A^{(k)}, \mu_{i,it-1}^{(k)}, \mathbf{v}_{i,it-1}^{(k)}, \{\mathbf{v}_j^{(k)}\}_{j=1}^{i-1}, \mathbf{b}_i^{(k)}) \\
& \quad \quad \{\mu_i^{(k)}, \mathbf{v}_i^{(k)}\} = \{\mu_{i,\nu_1}^{(k)}, \mathbf{v}_{i,\nu_1}^{(k)}\} \\
& \quad \quad \mathbf{b}_i^{(k+1)} = R_k^{k+1} \mathbf{b}_i^{(k)} + (A^{(k+1)} R_k^{k+1} - R_k^{k+1} A^{(k)}) \mathbf{v}_i^{(k)} \\
& \quad \quad \mathbf{v}_{i,0}^{(k+1)} = R_k^{k+1} \mathbf{v}_i^{(k)} \\
& \quad \quad \mu_{i,0}^{(k+1)} = \mu_i^{(k)} \\
& \quad \text{Solve the coarse level problem } A^{(L)} \mathbf{v}_i^{(L)} = \mu_i^{(L)} \mathbf{v}_i^{(L)} + \mathbf{b}_i^{(L)} \\
& \quad \text{with } (\mathbf{v}_i^{(L)}, R_l^L \mathbf{v}_j^{(l)}) = (R_{L-1}^L \mathbf{v}_i^{(L)}, R_l^L \mathbf{v}_j^{(l)}), j = 1, \dots, i-1 \\
& \quad \text{for } k = L-1, \dots, l \\
& \quad \quad \mathbf{v}_{i,0}^{(k)} = \mathbf{v}_i^{(k)} + P_{k+1}^k (\mathbf{v}_i^{(k+1)} - R_k^{k+1} \mathbf{v}_i^{(k)}) \\
& \quad \quad \mu_{i,0}^{(k)} = \mu_i^{(k+1)} \\
& \quad \quad \text{for } it = 1, \dots, \nu_2 \quad (\text{postsmoothing}) \\
& \quad \quad \quad \{\mu_{i,it}^{(k)}, \mathbf{v}_{i,it}^{(k)}\} = \text{Relax} (A^{(k)}, \mu_{i,it-1}^{(k)}, \mathbf{v}_{i,it-1}^{(k)}, \{\mathbf{v}_j^{(k)}\}_{j=1}^{i-1}, \mathbf{b}_i^{(k)}) \\
& \quad \quad \quad \{\mu_i^{(k)}, \mathbf{v}_i^{(k)}\} = \{\mu_{i,\nu_2}^{(k)}, \mathbf{v}_{i,\nu_2}^{(k)}\}
\end{aligned}$$

Here $\text{Relax} (A, \mu_i, \mathbf{v}_i, \{\mathbf{v}_j\}_{j=1}^{i-1}, \mathbf{b})$ denotes a nonlinear iteration step applied for solving the nonlinear problem $(A - \hat{\mu}_i I) \mathbf{u}_i = \mathbf{b}$ with an orthogonalization condition to $\{\mathbf{v}_j\}_{j=1}^{i-1}$ using $\{\mu_i, \mathbf{v}_i\}$ as initial eigenpair approximation.

Recall that the main idea of the FAS-EV method is in the construction of the coarse level problem so that its solution is a just the fine level solution transferred to the coarse level. Using it we obtain the following sequence of problems

$$\begin{aligned}
A^{(k)} \mathbf{w}_i^{(k)} = \hat{\mu}_i^{(k)} \mathbf{w}_i^{(k)} + \mathbf{b}_i^{(k)}, \quad \mathbf{b}_i^{(k)} = R_{k-1}^k \mathbf{b}_i^{(k-1)} + (A^{(k)} R_{k-1}^k - R_{k-1}^k A^{(k-1)}) \mathbf{v}_i^{(k-1)}, \\
\phi_k(\mathbf{w}_i^{(k)}) = \sigma_k, \quad \mathbf{b}_i^{(l)} = \mathbf{0}, \quad k = l, l+1, \dots, L, \quad i = 1, \dots, p.
\end{aligned} \tag{6}$$

where ϕ_k is a normalization condition, which guarantee the uniqueness, and σ_k specifies the size of the solution.

Let $\mathbf{u}_j^{(l)}$ is the solution on level l , then $R_l^{l+1}\mathbf{u}_j^{(l)}$ is the solution on next level, and hence, if we would like to find $\mathbf{u}_i^{(l)}$, $i \neq j$, we have to orthogonalize the current eigenvector approximation $\mathbf{v}_i^{(l+1)}$ to $R_l^{l+1}\mathbf{u}_j^{(l)}$. Unfortunately, we do not know the exact solution $\mathbf{u}_j^{(l)}$ (it is our original problem!), but we have $\mathbf{v}_j^{(l)}$, $j < i$, which is a good approximation to $\mathbf{u}_j^{(l)}$. Thus, from a practical point of view $R_l^{l+1}\mathbf{v}_j^{(l)}$ is an approximation to $R_l^{l+1}\mathbf{u}_j^{(l)}$, which one can use to keep orthogonalization on the level $l+1$. Now we define the following orthonormalization condition

$$\phi_k(\mathbf{v}_i^{(k)}) = (\mathbf{v}_i^{(k)}, R_l^k \mathbf{v}_j^{(l)}), \quad \sigma_k = (R_{k-1}^k \mathbf{v}_i^{(k)}, R_l^k \mathbf{v}_j^{(l)}) \quad j = 1, \dots, i. \quad (7)$$

$$\begin{aligned} \{\mu_{i,it}^{(k)}, \mathbf{v}_{i,it}^{(k)}\} &= \mathbf{Relax} (A^{(k)}, \mu_{i,it-1}^{(k)}, \mathbf{v}_{i,it-1}^{(k)}, \{\mathbf{v}_s^{(k)}\}_{s=1}^{i-1}, \mathbf{b}^{(k)}) \\ \mathbf{w}_{i,0}^{(k)} &= \mathbf{GSI} (A^{(k)} - \mu_{i,it-1}^{(k)} I_k, \mathbf{v}_{i,it-1}^{(k)}, \mathbf{b}^{(k)}) \\ \mathbf{for} \quad s &= 1, \dots, i-1 \quad (\text{orthogonalization}) \\ \alpha_s &= \frac{(R_l^k \mathbf{v}_s^{(l)}, \mathbf{w}_{i,s-1}^{(k)}) - (R_l^k \mathbf{v}_s^{(l)}, R_{k-1}^k \mathbf{v}_i^{(k-1)})}{(R_l^k \mathbf{v}_s^{(l)}, R_l^k \mathbf{v}_s^{(l)})} \\ \mathbf{w}_{i,s}^{(k)} &= \mathbf{w}_{i,s-1}^{(k)} - \alpha_s \cdot R_l^k \mathbf{v}_s^{(l)} \\ \mathbf{v}_{i,it}^{(k)} &= \frac{(R_{k-1}^k \mathbf{v}_i^{(k-1)}, R_l^k \mathbf{v}_i^{(l)})}{(\mathbf{w}_{i,i-1}^{(k)}, R_l^k \mathbf{v}_i^{(l)})} \mathbf{w}_{i,i-1}^{(k)} \\ \mu_{i,it}^{(k)} &= \frac{(A^{(k)} \mathbf{v}_{i,it}^{(k)} - \mathbf{b}^{(k)}, \mathbf{v}_{i,it}^{(k)})}{(\mathbf{v}_{i,it}^{(k)}, \mathbf{v}_{i,it}^{(k)})} \end{aligned}$$

Here $\mathbf{GSI} (A, \mathbf{y}, \mathbf{b})$ denotes the Gauss-Seidel iteration step applied for solving the linear system $A\mathbf{x} = \mathbf{b}$ using \mathbf{y} as initial guess.

Nevertheless, we have to note that the condition (7) does not guarantee a uniqueness of the solution (6), since we work with approximations rather than exact solutions. In our case it may happen that the sequence of test vectors $\{R_l^k \mathbf{v}_j^{(l)}\}_{j=1}^{i-1}$ used for the orthogonalization can be linear dependent, and hence, on the level k we will compute $\mathbf{v}_i^{(k)}$, which is a good approximation to $\mathbf{v}_j^{(l)}$, $j < i$, and which we already know, instead the desired approximation to $\mathbf{v}_i^{(l)}$. It is confirmed by numerical results.

To start the nested iteration process we have to find the coarse level approximation $\{\mu_i^{(L)}, \mathbf{v}_i^{(L)}\}$ to the solution $\{\lambda_i^{(L)}, \mathbf{u}_i^{(L)}\}$. To solve these problems we use the Gauss-Seidel iteration process followed by the standard normalization condition until a solution with a prescribed accuracy ε_{FMG} is not reached. On the other hand, for solving the coarse level problem in the inner multilevel process one can use the similar relaxation process as for the smoothing with analogous stopping criteria ε_{FAS} .

In order to investigate the whole computational complexity of the FMG-EV(p) method we recall that the solution of eigenvalue problems by the FAS-EV method and Ritz projection requires the largest computational costs. Assume that the number of nodes decreases in a geometrical ratio with a factor ρ defined by

$$\frac{n_{k+1}}{n_k} = \rho_k \leq \rho < 1, \quad k = 0, 1, \dots, L-1.$$

and hence, we have

$$\sum_{k=0}^{\infty} n_k \leq (1 - \rho)^{-1} n_0. \quad (8)$$

Denote by $W_V^{(l)}$ the whole computational complexity of the FAS-EV method applied for solving (5) on the level l . Let C_A and C_P denote the upper bound of the arithmetic work per unknowns defined by the matrix-vector multiplication with matrices $A^{(k)}$ and $P_{k+1}^k (R_k^{k+1})$, respectively. Moreover, we assume that the computational costs for solving the coarse level problem is proportional to the number of unknowns on the fine level, i.e., there is a positive constant C_0 such that

$$W_V^{(L)} \leq C_0 n_0. \quad (9)$$

Thus, taking into account (8) and (9) we have

$$\begin{aligned} \hat{W}_{V,i}^{(l)} &\leq \sum_{k=l}^{L-1} \left[(\nu_1 + \nu_2) \left\{ (3C_A + 9)n_k + (i-1) \left[(1-\rho)^{-1} C_P n_l + 7n_k \right] \right\} \right. \\ &\quad \left. + (2C_A + 6C_P + 4)n_k \right] + C_0 n_0 \\ &\leq [(\nu_1 + \nu_2) \{3C_A + 9 + (i-1)((L-l)C_P + 7)\} + 2C_A + 6C_P + 4] (1-\rho)^{-1} n_l + C_0 n_0. \end{aligned}$$

Next summing by i we obtain

$$\begin{aligned} \hat{W}_V^{(l)} &\leq \sum_{i=1}^p \hat{W}_{V,i}^{(l)} \leq p(\nu_1 + \nu_2)(3C_A + 9)(1-\rho)^{-1} n_l + \frac{1}{2}p(p-1)((L-l)C_P + 7)(1-\rho)^{-1} n_l \\ &\quad + p(2C_A + 6C_P + 4)(1-\rho)^{-1} n_l + pC_0 n_0. \end{aligned}$$

On the other hand, the Ritz projection method on the level l requires

$$W_{Ritz}^{(l)} = \frac{1}{2}p(5p+3)n_l + pC_A n_l + O(p^3).$$

Now taking into account the prolongation of p vectors from level n_{l+1} to n_l and the computation of their Ritz projection on the level l we obtain the computational complexity of the FMG-EV method for the transfer from level $l+1$ to level l

$$\begin{aligned} \hat{W}^{(l)} &\leq q\hat{W}_V^{(l)} + pC_P n_l + W_{Ritz}^{(l)} \\ &\leq Cq(\nu_1 + \nu_2) (p^2(L-l)(1-\rho)^{-1} n_l + pn_0) + O(p^2 n_l) + O(p^3). \end{aligned}$$

Finally, summing by levels we obtain that the total computational complexity denoted by \hat{W} is proportional to the number of nodes on the fine level, the number of level used and the number of desired eigenvectors, i.e.,

$$\begin{aligned} \hat{W} &= \sum_{l=0}^L \hat{W}^{(l)} \leq Cq(\nu_1 + \nu_2) \left(p^2 \frac{L}{\ln \rho^{-1}} (1-\rho)^{-2} n_0 + pLn_0 \right) + O(p^2 Ln_0) + O(Lp^3) \\ &\leq Cq(\nu_1 + \nu_2) p^2 Ln_0 = O(q(\nu_1 + \nu_2) p^2 Ln_0). \end{aligned} \quad (10)$$

where C is a positive constant. Hence, the total computational costs of the FMG-EV method is nearly optimal.

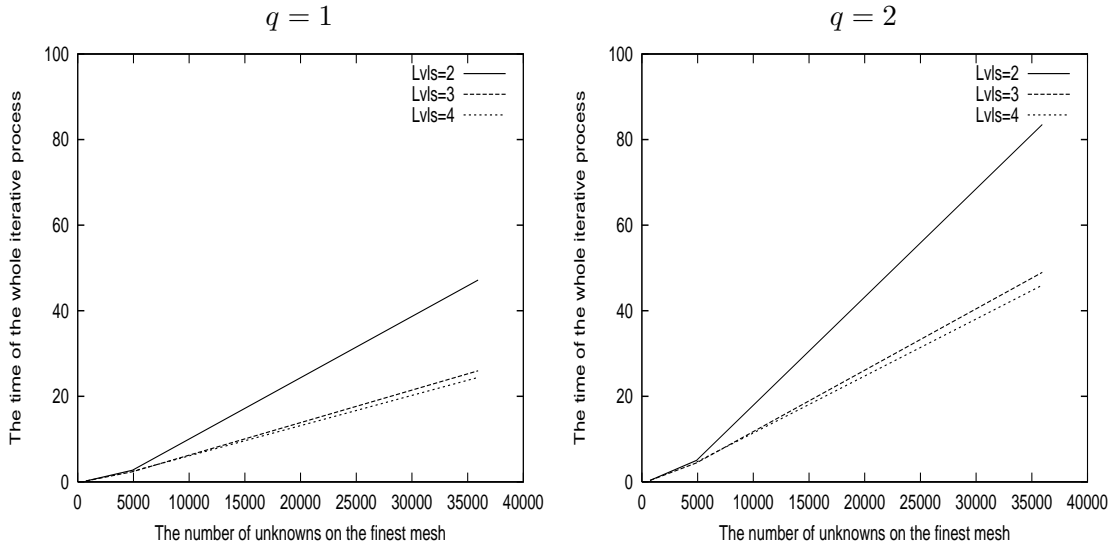


Fig. 2. The time of the whole iterative process for finding the first eigenpair with respect to the number of unknowns on the finest level

4 Numerical results

To test the method we consider the eigenvalue problem (3), which corresponds to the piecewise-linear finite-element discretization of the three dimensional second-order elliptic problems

$$\begin{aligned}
 -\Delta u &= \lambda u & \text{in } \Omega, \\
 u &= 0 & \text{on } \Gamma_D = \partial\Omega, \\
 \|u\| &= 1
 \end{aligned} \tag{11}$$

in the cube domain $\Omega = [0, 1]^3$ on a uniform Cartesian mesh Υ_h with stepsize h . This problem has the exact solution.

To illustrate the quality of the eigensolver used we shall show that the accuracy of the eigenvalue approximation is tended to zero when we increase the number of the relaxation sweeps per level and the number of inner FAS-EV iterations and at the same moment the computational complexity is still optimal with respect to the different parameters used.

The first group of numerical experiments were performed for test the accuracy. It is well-known that in the case of multigrid method when the number of smoothing steps increase and when we find an exact solution on coarse levels, then the multigrid solver for linear system of equation is going to be a direct solver, i.e., the difference between exact and computed solutions tends to a machine accuracy. We expect the similar behaviour for our eigensolver. To guarantee the exact solvers on the coarse level we define the following values for $\varepsilon_{FMG} = 10^{-31}$ and $\varepsilon_{FAS} = 10^{-16}$.

In Tables 1–2 the results of the experiments for the FMG-EV method for different choices of relaxation steps and inner iterations are given. In all tables of the paper ν is the number of pre- and post-smoothing iterations on each level ($\nu = \nu_1 = \nu_2$), q is the number of inner FAS-EV iterations applied on each level and Lvl is the number of levels used. Moreover, μ is the computed approximation to the exact eigenvalue λ_i , and hence, the magnitude $|\lambda_i -$

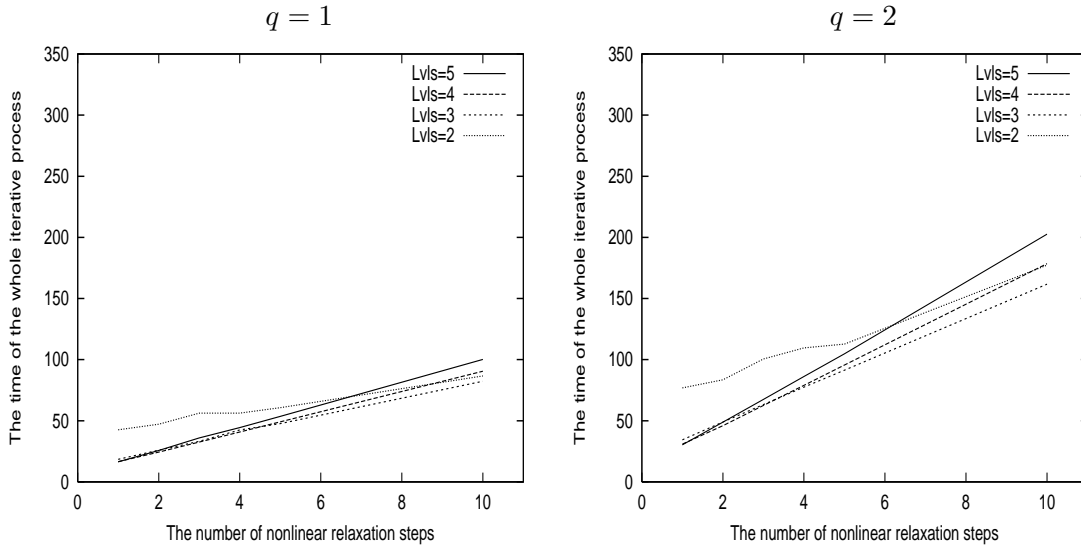


Fig. 3. The time of the whole iterative process for finding the first eigenpair with respect to the number of pre- and post- smoothing steps used

$|\mu_i|$ measures the difference between exact and computed eigenvalues, whereas $\|\mathbf{A}\mathbf{v}_i - \mu_i\mathbf{v}_i\|$ measures the quality of the approximated pair $\{\mu_i, \mathbf{v}_i\}$.

The first results in [5] shows that the accuracy of the approximated solution dose not depend on the order of the eigenproblem to be solved, and hence, in what follows we present the results only for $N = 32$ to avoid a large number of similar figures and tables for different grids.

From the presented numerical results we see that if we increase the number of the relaxation steps per level without changes of other parameters, then the accuracy of all approximated eigenvalues is improved. Comparing results for $q = 1$ with corresponding ones for $q = 2$ in all Tables one can see that in the second case the accuracy of the approximated solution is *always* better, i.e, increasing the number of inner FAS-EV steps lead to the improving accuracy of the approximated eigenpairs $\{\mu_i, \mathbf{v}_i\}$.

The accuracy results in Tables 1–2 for different number of levels shows that the accuracy of approximated solutions does not depend on the number of level used, except one remark. Indeed, in two- and three-level cases we surprisely have a good convergence for the last eigenvalue λ_5 with respect to disconvergence for four- and five-level cases. It is happens since in the later cases the coarse level problem is not sufficiently reach to generate a good approximation to these eigenspaces, and hence, it is more important to find a good initial coarse level approximation, then trying to improve it later by the inner multilevel process.

q	ν	$ \lambda_1 - \mu_1 $	$ \lambda_2 - \mu_2 $	$ \lambda_3 - \mu_3 $	$ \lambda_4 - \mu_4 $	$ \lambda_5 - \mu_5 $
Lvls = 5, NFine = 29791, NCoarse = 1, $\varepsilon_{FMG} = 10^{-31}$, $\varepsilon_{FAS} = 10^{-16}$						
1	1	$0.83528 \cdot 10^{-2}$	$0.94833 \cdot 10^{-3}$	$0.94864 \cdot 10^{-3}$	$0.94896 \cdot 10^{-3}$	$0.83950 \cdot 10^{-1}$
	2	$0.69550 \cdot 10^{-6}$	$0.31825 \cdot 10^{-5}$	$0.31826 \cdot 10^{-5}$	$0.31827 \cdot 10^{-5}$	$0.43920 \cdot 10^{-1}$
	3	$0.16185 \cdot 10^{-9}$	$0.89322 \cdot 10^{-9}$	$0.89323 \cdot 10^{-9}$	$0.89323 \cdot 10^{-9}$	$0.21761 \cdot 10^{-1}$
	4	$0.34395 \cdot 10^{-13}$	$0.33402 \cdot 10^{-12}$	$0.33402 \cdot 10^{-12}$	$0.33403 \cdot 10^{-12}$	$0.19062 \cdot 10^{-1}$
	5	$0.86509 \cdot 10^{-17}$	$0.88854 \cdot 10^{-16}$	$0.88855 \cdot 10^{-16}$	$0.88862 \cdot 10^{-16}$	$0.18896 \cdot 10^{-1}$
	10	$0.20247 \cdot 10^{-29}$	$0.66829 \cdot 10^{-31}$	$0.26770 \cdot 10^{-31}$	$0.19259 \cdot 10^{-31}$	$0.18890 \cdot 10^{-1}$
2	1	$0.66492 \cdot 10^{-6}$	$0.28152 \cdot 10^{-5}$	$0.28162 \cdot 10^{-5}$	$0.28166 \cdot 10^{-5}$	$0.19049 \cdot 10^{-2}$
	2	$0.36474 \cdot 10^{-12}$	$0.55080 \cdot 10^{-11}$	$0.29096 \cdot 10^{-12}$	$0.30506 \cdot 10^{-12}$	$0.18892 \cdot 10^{-1}$
	3	$0.11061 \cdot 10^{-20}$	$0.48379 \cdot 10^{-15}$	$0.44919 \cdot 10^{-15}$	$0.11955 \cdot 10^{-19}$	$0.18891 \cdot 10^{-1}$
	4	$0.75144 \cdot 10^{-28}$	$0.38784 \cdot 10^{-18}$	$0.12579 \cdot 10^{-19}$	$0.11994 \cdot 10^{-26}$	$0.18890 \cdot 10^{-1}$
	5	$0.15214 \cdot 10^{-31}$	$0.15702 \cdot 10^{-21}$	$0.33236 \cdot 10^{-22}$	$0.10977 \cdot 10^{-31}$	$0.18890 \cdot 10^{-1}$
	10	$0.24459 \cdot 10^{-31}$	$0.19837 \cdot 10^{-31}$	$0.50074 \cdot 10^{-32}$	$0.32548 \cdot 10^{-31}$	$0.18890 \cdot 10^{-1}$
Lvls = 4, NFine = 29791, NCoarse = 27, $\varepsilon_{FMG} = 10^{-31}$, $\varepsilon_{FAS} = 10^{-16}$						
1	1	$0.29206 \cdot 10^{-3}$	$0.97645 \cdot 10^{-3}$	$0.97648 \cdot 10^{-3}$	$0.97726 \cdot 10^{-3}$	0.1805769470
	2	$0.78025 \cdot 10^{-6}$	$0.31911 \cdot 10^{-5}$	$0.31911 \cdot 10^{-5}$	$0.31911 \cdot 10^{-5}$	0.1722472399
	3	$0.15763 \cdot 10^{-9}$	$0.82750 \cdot 10^{-9}$	$0.82750 \cdot 10^{-9}$	$0.82751 \cdot 10^{-9}$	0.1719186764
	4	$0.42301 \cdot 10^{-13}$	$0.32098 \cdot 10^{-12}$	$0.32098 \cdot 10^{-12}$	$0.32099 \cdot 10^{-12}$	0.1718761739
	5	$0.86888 \cdot 10^{-17}$	$0.85300 \cdot 10^{-16}$	$0.85300 \cdot 10^{-16}$	$0.85307 \cdot 10^{-16}$	0.1718697271
	10	$0.94370 \cdot 10^{-32}$	$0.75111 \cdot 10^{-32}$	$0.19066 \cdot 10^{-31}$	$0.21955 \cdot 10^{-31}$	0.1718685606
2	1	$0.72755 \cdot 10^{-6}$	$0.28289 \cdot 10^{-5}$	$0.28290 \cdot 10^{-5}$	$0.28291 \cdot 10^{-5}$	0.1719674484
	2	$0.39828 \cdot 10^{-13}$	$0.29559 \cdot 10^{-12}$	$0.29563 \cdot 10^{-12}$	$0.29565 \cdot 10^{-12}$	0.1718698729
	3	$0.10626 \cdot 10^{-20}$	$0.10586 \cdot 10^{-19}$	$0.12209 \cdot 10^{-19}$	$0.16090 \cdot 10^{-19}$	0.1718685910
	4	$0.55921 \cdot 10^{-28}$	$0.10966 \cdot 10^{-26}$	$0.20140 \cdot 10^{-25}$	$0.25174 \cdot 10^{-24}$	0.1718685612
	5	$0.19451 \cdot 10^{-31}$	$0.62785 \cdot 10^{-31}$	$0.37420 \cdot 10^{-30}$	$0.14590 \cdot 10^{-29}$	0.1718685605
	10	$0.16370 \cdot 10^{-31}$	$0.15312 \cdot 10^{-21}$	$0.41675 \cdot 10^{-21}$	$0.18266 \cdot 10^{-23}$	0.1718685605
Lvls = 3, NFine = 29791, NCoarse = 343, $\varepsilon_{FMG} = 10^{-31}$, $\varepsilon_{FAS} = 10^{-16}$						
1	1	$0.29204 \cdot 10^{-3}$	$0.97639 \cdot 10^{-3}$	$0.97655 \cdot 10^{-3}$	$0.97723 \cdot 10^{-3}$	$0.18430 \cdot 10^{-2}$
	2	$0.78025 \cdot 10^{-6}$	$0.31911 \cdot 10^{-5}$	$0.31911 \cdot 10^{-5}$	$0.31912 \cdot 10^{-5}$	$0.71561 \cdot 10^{-5}$
	3	$0.15763 \cdot 10^{-9}$	$0.82750 \cdot 10^{-9}$	$0.82751 \cdot 10^{-9}$	$0.82751 \cdot 10^{-9}$	$0.23087 \cdot 10^{-8}$
	4	$0.42301 \cdot 10^{-13}$	$0.32098 \cdot 10^{-12}$	$0.32098 \cdot 10^{-12}$	$0.32099 \cdot 10^{-12}$	$0.11690 \cdot 10^{-11}$
	5	$0.86888 \cdot 10^{-17}$	$0.85299 \cdot 10^{-16}$	$0.85299 \cdot 10^{-16}$	$0.85300 \cdot 10^{-16}$	$0.38678 \cdot 10^{-15}$
	10	$0.11362 \cdot 10^{-31}$	$0.82814 \cdot 10^{-32}$	$0.71259 \cdot 10^{-31}$	$0.81659 \cdot 10^{-31}$	$0.10400 \cdot 10^{-31}$
2	1	$0.72755 \cdot 10^{-6}$	$0.28289 \cdot 10^{-5}$	$0.28290 \cdot 10^{-5}$	$0.28291 \cdot 10^{-5}$	$0.61068 \cdot 10^{-5}$
	2	$0.39828 \cdot 10^{-13}$	$0.29558 \cdot 10^{-12}$	$0.29560 \cdot 10^{-12}$	$0.29561 \cdot 10^{-12}$	$0.10645 \cdot 10^{-11}$
	3	$0.10626 \cdot 10^{-20}$	$0.10593 \cdot 10^{-19}$	$0.11592 \cdot 10^{-19}$	$0.16149 \cdot 10^{-19}$	$0.52750 \cdot 10^{-19}$
	4	$0.55899 \cdot 10^{-28}$	$0.10728 \cdot 10^{-26}$	$0.50317 \cdot 10^{-25}$	$0.25403 \cdot 10^{-24}$	$0.85392 \cdot 10^{-26}$
	5	$0.10977 \cdot 10^{-31}$	$0.30814 \cdot 10^{-31}$	$0.16555 \cdot 10^{-29}$	$0.22887 \cdot 10^{-29}$	$0.25422 \cdot 10^{-31}$
	10	$0.14829 \cdot 10^{-31}$	$0.41214 \cdot 10^{-31}$	$0.10014 \cdot 10^{-31}$	$0.13481 \cdot 10^{-31}$	$0.10168 \cdot 10^{-30}$
Lvls = 2, NFine = 29791, NCoarse = 3375, $\varepsilon_{FMG} = 10^{-31}$, $\varepsilon_{FAS} = 10^{-16}$						
1	1	$0.28836 \cdot 10^{-3}$	$0.96578 \cdot 10^{-3}$	$0.96578 \cdot 10^{-3}$	$0.96578 \cdot 10^{-3}$	$0.18266 \cdot 10^{-2}$
	2	$0.78025 \cdot 10^{-6}$	$0.31911 \cdot 10^{-5}$	$0.31911 \cdot 10^{-5}$	$0.31911 \cdot 10^{-5}$	$0.71560 \cdot 10^{-5}$
	3	$0.15763 \cdot 10^{-9}$	$0.82750 \cdot 10^{-9}$	$0.82750 \cdot 10^{-9}$	$0.82750 \cdot 10^{-9}$	$0.23087 \cdot 10^{-8}$
	4	$0.42301 \cdot 10^{-13}$	$0.32098 \cdot 10^{-12}$	$0.32098 \cdot 10^{-12}$	$0.32098 \cdot 10^{-12}$	$0.11690 \cdot 10^{-11}$
	5	$0.86888 \cdot 10^{-17}$	$0.85299 \cdot 10^{-16}$	$0.85299 \cdot 10^{-16}$	$0.85299 \cdot 10^{-16}$	$0.38678 \cdot 10^{-15}$
	10	$0.44296 \cdot 10^{-32}$	$0.18681 \cdot 10^{-31}$	$0.18103 \cdot 10^{-31}$	$0.20029 \cdot 10^{-31}$	$0.80889 \cdot 10^{-32}$
2	1	$0.72754 \cdot 10^{-6}$	$0.28289 \cdot 10^{-5}$	$0.28289 \cdot 10^{-5}$	$0.28289 \cdot 10^{-5}$	$0.61063 \cdot 10^{-5}$
	2	$0.39827 \cdot 10^{-13}$	$0.29557 \cdot 10^{-12}$	$0.29557 \cdot 10^{-12}$	$0.29557 \cdot 10^{-12}$	$0.10644 \cdot 10^{-11}$
	3	$0.10626 \cdot 10^{-20}$	$0.10585 \cdot 10^{-19}$	$0.10585 \cdot 10^{-19}$	$0.10585 \cdot 10^{-19}$	$0.52701 \cdot 10^{-19}$
	4	$0.55911 \cdot 10^{-28}$	$0.10640 \cdot 10^{-26}$	$0.10641 \cdot 10^{-26}$	$0.10641 \cdot 10^{-26}$	$0.84829 \cdot 10^{-26}$
	5	$0.90518 \cdot 10^{-32}$	$0.19644 \cdot 10^{-31}$	$0.12903 \cdot 10^{-31}$	$0.24459 \cdot 10^{-31}$	$0.16948 \cdot 10^{-31}$
	10	$0.24844 \cdot 10^{-31}$	$0.47570 \cdot 10^{-31}$	$0.20029 \cdot 10^{-31}$	$0.25614 \cdot 10^{-31}$	$0.49688 \cdot 10^{-31}$

Table 1. Accuracy of computed eigenvalues μ_i as a function of ν and q , $p = 5$

q	ν	$\ \mathbf{Av}_1 - \mu_1 \mathbf{v}_1\ $	$\ \mathbf{Av}_2 - \mu_2 \mathbf{v}_2\ $	$\ \mathbf{Av}_3 - \mu_3 \mathbf{v}_3\ $	$\ \mathbf{Av}_4 - \mu_4 \mathbf{v}_4\ $	$\ \mathbf{Av}_5 - \mu_5 \mathbf{v}_5\ $
Lvls = 5, NFine = 29791, NCoarse = 1, $\varepsilon_{FMG} = 10^{-31}$, $\varepsilon_{FAS} = 10^{-16}$						
1	1	$0.29266 \cdot 10^{-1}$	$0.71637 \cdot 10^{-1}$	$0.71649 \cdot 10^{-1}$	$0.71658 \cdot 10^{-1}$	$0.83950 \cdot 10^{-1}$
	2	$0.22693 \cdot 10^{-2}$	$0.45634 \cdot 10^{-2}$	$0.45634 \cdot 10^{-2}$	$0.45635 \cdot 10^{-2}$	$0.65110 \cdot 10^{-1}$
	3	$0.32745 \cdot 10^{-4}$	$0.76292 \cdot 10^{-4}$	$0.76293 \cdot 10^{-4}$	$0.76293 \cdot 10^{-4}$	$0.23570 \cdot 10^{-1}$
	4	$0.53360 \cdot 10^{-6}$	$0.14788 \cdot 10^{-5}$	$0.14788 \cdot 10^{-5}$	$0.14788 \cdot 10^{-5}$	$0.57555 \cdot 10^{-2}$
	5	$0.76464 \cdot 10^{-8}$	$0.24110 \cdot 10^{-7}$	$0.24110 \cdot 10^{-7}$	$0.24110 \cdot 10^{-7}$	$0.98312 \cdot 10^{-3}$
	10	$0.45221 \cdot 10^{-15}$	$0.45557 \cdot 10^{-16}$	$0.80728 \cdot 10^{-16}$	$0.38392 \cdot 10^{-16}$	$0.52163 \cdot 10^{-5}$
2	1	$0.21825 \cdot 10^{-2}$	$0.42567 \cdot 10^{-2}$	$0.42571 \cdot 10^{-2}$	$0.42573 \cdot 10^{-2}$	$0.94627 \cdot 10^{-2}$
	2	$0.52210 \cdot 10^{-6}$	$0.14352 \cdot 10^{-5}$	$0.14352 \cdot 10^{-5}$	$0.14352 \cdot 10^{-5}$	$0.40164 \cdot 10^{-3}$
	3	$0.85727 \cdot 10^{-10}$	$0.28196 \cdot 10^{-9}$	$0.27972 \cdot 10^{-9}$	$0.27937 \cdot 10^{-9}$	$0.50707 \cdot 10^{-4}$
	4	$0.19581 \cdot 10^{-13}$	$0.31330 \cdot 10^{-12}$	$0.10004 \cdot 10^{-12}$	$0.87396 \cdot 10^{-13}$	$0.50515 \cdot 10^{-5}$
	5	$0.14869 \cdot 10^{-16}$	$0.45261 \cdot 10^{-15}$	$0.60726 \cdot 10^{-16}$	$0.24530 \cdot 10^{-16}$	$0.69688 \cdot 10^{-6}$
	10	$0.23540 \cdot 10^{-26}$	$0.21185 \cdot 10^{-19}$	$0.17954 \cdot 10^{-19}$	$0.20190 \cdot 10^{-19}$	$0.46533 \cdot 10^{-9}$
Lvls = 4, NFine = 29791, NCoarse = 27, $\varepsilon_{FMG} = 10^{-31}$, $\varepsilon_{FAS} = 10^{-16}$						
1	1	$0.39956 \cdot 10^{-1}$	$0.72774 \cdot 10^{-1}$	$0.72776 \cdot 10^{-1}$	$0.97726 \cdot 10^{-1}$	0.1867057169
	2	$0.22781 \cdot 10^{-2}$	$0.45793 \cdot 10^{-2}$	$0.45793 \cdot 10^{-2}$	$0.45793 \cdot 10^{-2}$	$0.19440 \cdot 10^{-1}$
	3	$0.32745 \cdot 10^{-4}$	$0.76292 \cdot 10^{-4}$	$0.76293 \cdot 10^{-4}$	$0.76293 \cdot 10^{-4}$	$0.31491 \cdot 10^{-2}$
	4	$0.53360 \cdot 10^{-6}$	$0.14788 \cdot 10^{-5}$	$0.14788 \cdot 10^{-5}$	$0.14788 \cdot 10^{-5}$	$0.12095 \cdot 10^{-2}$
	5	$0.76464 \cdot 10^{-8}$	$0.24110 \cdot 10^{-7}$	$0.24110 \cdot 10^{-7}$	$0.24110 \cdot 10^{-7}$	$0.47354 \cdot 10^{-3}$
	10	$0.44894 \cdot 10^{-17}$	$0.73339 \cdot 10^{-16}$	$0.43861 \cdot 10^{-16}$	$0.76619 \cdot 10^{-16}$	$0.43801 \cdot 10^{-5}$
2	1	$0.21843 \cdot 10^{-2}$	$0.42567 \cdot 10^{-2}$	$0.42571 \cdot 10^{-2}$	$0.42572 \cdot 10^{-2}$	$0.15604 \cdot 10^{-1}$
	2	$0.52210 \cdot 10^{-6}$	$0.14352 \cdot 10^{-5}$	$0.14352 \cdot 10^{-5}$	$0.14352 \cdot 10^{-5}$	$0.48879 \cdot 10^{-3}$
	3	$0.85727 \cdot 10^{-10}$	$0.28322 \cdot 10^{-9}$	$0.28063 \cdot 10^{-9}$	$0.27935 \cdot 10^{-9}$	$0.76028 \cdot 10^{-4}$
	4	$0.19535 \cdot 10^{-13}$	$0.33968 \cdot 10^{-12}$	$0.12704 \cdot 10^{-12}$	$0.87305 \cdot 10^{-13}$	$0.11685 \cdot 10^{-4}$
	5	$0.39948 \cdot 10^{-17}$	$0.38703 \cdot 10^{-15}$	$0.18923 \cdot 10^{-15}$	$0.30853 \cdot 10^{-16}$	$0.17964 \cdot 10^{-5}$
	10	$0.64462 \cdot 10^{-31}$	$0.28575 \cdot 10^{-16}$	$0.18990 \cdot 10^{-16}$	$0.54995 \cdot 10^{-17}$	$0.15441 \cdot 10^{-9}$
Lvls = 3, NFine = 29791, NCoarse = 343, $\varepsilon_{FMG} = 10^{-31}$, $\varepsilon_{FAS} = 10^{-16}$						
1	1	$0.39687 \cdot 10^{-1}$	$0.71641 \cdot 10^{-1}$	$0.71644 \cdot 10^{-1}$	$0.71656 \cdot 10^{-1}$	$0.97039 \cdot 10^{-1}$
	2	$0.22757 \cdot 10^{-2}$	$0.45634 \cdot 10^{-2}$	$0.45634 \cdot 10^{-2}$	$0.45635 \cdot 10^{-2}$	$0.67757 \cdot 10^{-2}$
	3	$0.32745 \cdot 10^{-4}$	$0.76292 \cdot 10^{-4}$	$0.76293 \cdot 10^{-4}$	$0.76293 \cdot 10^{-4}$	$0.12983 \cdot 10^{-3}$
	4	$0.53360 \cdot 10^{-6}$	$0.14788 \cdot 10^{-5}$	$0.14788 \cdot 10^{-5}$	$0.14788 \cdot 10^{-5}$	$0.28500 \cdot 10^{-5}$
	5	$0.76464 \cdot 10^{-8}$	$0.24110 \cdot 10^{-7}$	$0.24110 \cdot 10^{-7}$	$0.24110 \cdot 10^{-7}$	$0.51928 \cdot 10^{-7}$
	10	$0.44894 \cdot 10^{-17}$	$0.62019 \cdot 10^{-16}$	$0.33795 \cdot 10^{-16}$	$0.28857 \cdot 10^{-16}$	$0.10870 \cdot 10^{-15}$
2	1	$0.21843 \cdot 10^{-2}$	$0.42567 \cdot 10^{-2}$	$0.42570 \cdot 10^{-2}$	$0.42572 \cdot 10^{-2}$	$0.61919 \cdot 10^{-2}$
	2	$0.52210 \cdot 10^{-6}$	$0.14352 \cdot 10^{-5}$	$0.14352 \cdot 10^{-5}$	$0.14352 \cdot 10^{-5}$	$0.27464 \cdot 10^{-5}$
	3	$0.85727 \cdot 10^{-10}$	$0.28461 \cdot 10^{-9}$	$0.28173 \cdot 10^{-9}$	$0.27938 \cdot 10^{-9}$	$0.64873 \cdot 10^{-9}$
	4	$0.19535 \cdot 10^{-13}$	$0.31547 \cdot 10^{-13}$	$0.11360 \cdot 10^{-12}$	$0.87354 \cdot 10^{-12}$	$0.25379 \cdot 10^{-12}$
	5	$0.39948 \cdot 10^{-17}$	$0.62357 \cdot 10^{-15}$	$0.21039 \cdot 10^{-15}$	$0.35207 \cdot 10^{-16}$	$0.11873 \cdot 10^{-15}$
	10	$0.66504 \cdot 10^{-31}$	$0.32661 \cdot 10^{-22}$	$0.79563 \cdot 10^{-26}$	$0.19303 \cdot 10^{-22}$	$0.25208 \cdot 10^{-25}$
Lvls = 2, NFine = 29791, NCoarse = 3375, $\varepsilon_{FMG} = 10^{-31}$, $\varepsilon_{FAS} = 10^{-16}$						
1	1	$0.39700 \cdot 10^{-1}$	$0.72378 \cdot 10^{-1}$	$0.72378 \cdot 10^{-1}$	$0.72378 \cdot 10^{-1}$	$0.99144 \cdot 10^{-1}$
	2	$0.22781 \cdot 10^{-2}$	$0.45793 \cdot 10^{-2}$	$0.45793 \cdot 10^{-2}$	$0.45793 \cdot 10^{-2}$	$0.68194 \cdot 10^{-2}$
	3	$0.32312 \cdot 10^{-4}$	$0.73423 \cdot 10^{-4}$	$0.73423 \cdot 10^{-4}$	$0.73423 \cdot 10^{-4}$	$0.12174 \cdot 10^{-3}$
	4	$0.53015 \cdot 10^{-6}$	$0.14496 \cdot 10^{-5}$	$0.14496 \cdot 10^{-5}$	$0.14496 \cdot 10^{-5}$	$0.27490 \cdot 10^{-5}$
	5	$0.75984 \cdot 10^{-8}$	$0.23622 \cdot 10^{-7}$	$0.23622 \cdot 10^{-7}$	$0.23622 \cdot 10^{-7}$	$0.49987 \cdot 10^{-7}$
	10	$0.44587 \cdot 10^{-17}$	$0.28209 \cdot 10^{-16}$	$0.28209 \cdot 10^{-16}$	$0.28209 \cdot 10^{-16}$	$0.10411 \cdot 10^{-15}$
2	1	$0.21859 \cdot 10^{-2}$	$0.42678 \cdot 10^{-2}$	$0.42678 \cdot 10^{-2}$	$0.42678 \cdot 10^{-2}$	$0.62206 \cdot 10^{-2}$
	2	$0.51935 \cdot 10^{-6}$	$0.14153 \cdot 10^{-5}$	$0.14153 \cdot 10^{-5}$	$0.14153 \cdot 10^{-5}$	$0.26837 \cdot 10^{-5}$
	3	$0.84032 \cdot 10^{-10}$	$0.26284 \cdot 10^{-9}$	$0.26284 \cdot 10^{-9}$	$0.26284 \cdot 10^{-9}$	$0.58309 \cdot 10^{-9}$
	4	$0.19277 \cdot 10^{-13}$	$0.83157 \cdot 10^{-13}$	$0.83157 \cdot 10^{-13}$	$0.83157 \cdot 10^{-13}$	$0.23354 \cdot 10^{-12}$
	5	$0.39426 \cdot 10^{-17}$	$0.22772 \cdot 10^{-16}$	$0.22772 \cdot 10^{-16}$	$0.22772 \cdot 10^{-16}$	$0.79539 \cdot 10^{-16}$
	10	$0.75781 \cdot 10^{-31}$	$0.11026 \cdot 10^{-27}$	$0.53513 \cdot 10^{-28}$	$0.25093 \cdot 10^{-28}$	$0.38020 \cdot 10^{-17}$

Table 2. Accuracy of computed eigenpairs $\{\mu_i, \mathbf{v}_i\}$ as a function of ν and q , $p = 5$

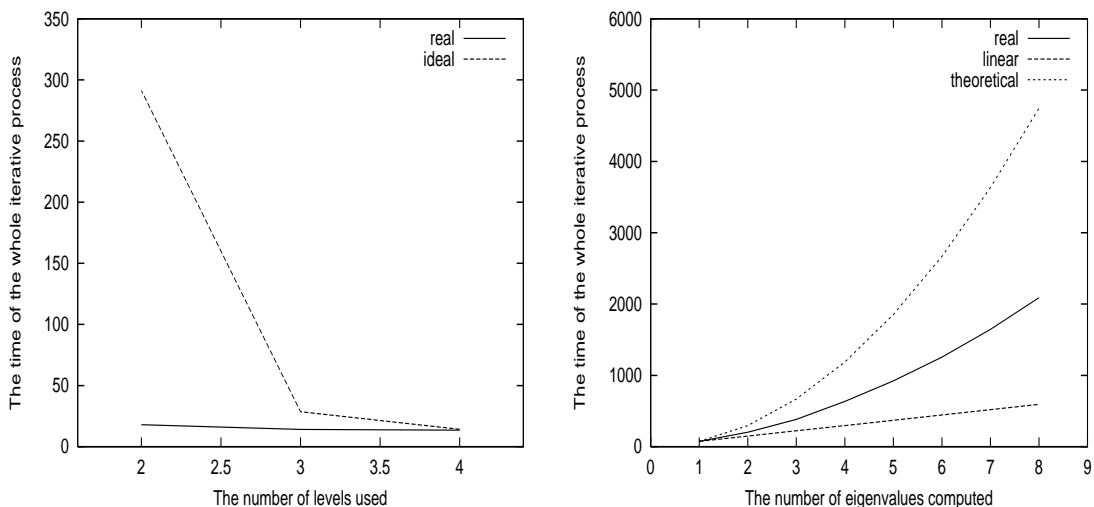


Fig. 4. The time of the whole iterative process with respect to the number of levels and the number of eigenpairs computed

Now compare the results from Tables 1 and 2 for the first eigenpair $\{\lambda_1, \mathbf{v}_1\}$ with ones from [5] one can see that when more eigenvectors, whether or not they converge, are included in the process, all approximations are improved.

Hence, we show that the FMG-EV method can compute the first eigenvalues with a desired order of accuracy by corresponding choices of eigensolver parameters, when we solve the coarse level problems exactly. Hence, in practice, $q = 1$ and $\nu = 3$ or close to these values are sufficient to ensure the convergence of the FMG-EV(p) method.

The second group of numerical experiments were performed for various combinations of the values ε_{FAS} and ε_{FMG} , when the number of inner iterations and relaxation steps are fixed and is always taken as $q = 1$ and $\nu = 2$. The behavior both of the accuracy of eigenvalue approximationis and of the time of the whole iterative process with respect to ε_{FAS} and ε_{FMG} for different number of levels are shown in Figures 1.1–1.4. Here, the value of ε_{FMG} and ε_{FAS} are changed in a range from 10^{-1} to 10^{-10} , and the power of 10 are given along the left axis for ε_{FMG} and along the right axis for ε_{FAS} .

From the presented numerical results we see that the accuracy of approximated solution depends only slightly on the values of ε_{FAS} and there is a strong dependence on the values of ε_{FMG} . Here we have to note that starting from some values of ε_{FMG} and ε_{FAS} the accuracy does not changed, i.e., we reach the maximal accuracy for the given numbers ν and q . On the other hand, there is a strong dependence of the computing time on the values of ε_{FAS} and, at the same time, a weak dependence on ε_{FMG} . Hence, in practice, the optimal choice of the iterative parameters seems to be $\varepsilon_{FMG} = 10^{-4}$ and $\varepsilon_{FAS} = 10^{-2}$ or close to these values.

Moreover, the present experiments shows that the FMG-EV(p) method is of capable to deal with a multiply eigenvalue. Indeed, we have $\lambda_1 = \lambda_2 = \lambda_3$ and how one can see from the numerical results presented here we find this eigenvalue and its eigensubspace without problems.

The next group of numerical experiments were performed for various problem sizes and

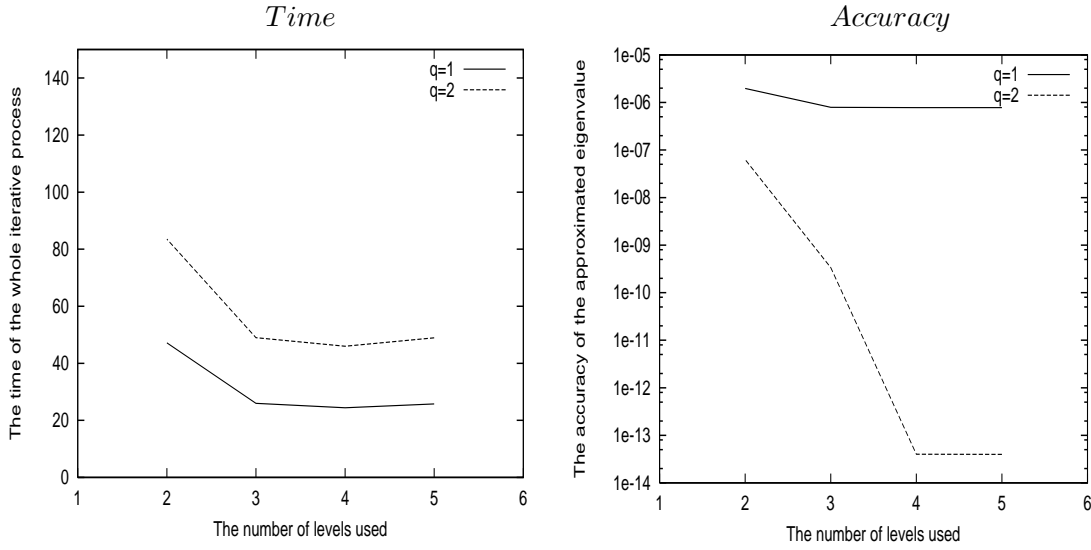


Fig. 5. The time of the whole iterative process and the accuracy for the first eigenvalue with respect to the number of level used

for different numbers of relaxation steps used, when ε_{FMG} and ε_{FAS} are optimal in above mentioned sense and are fixed. The time of the whole iterative process for various values of Lvl_s with respect to the number of unknowns on the finest mesh and the number of relaxation steps are given in Figures 2 and 3, respectively. Moreover, in Figure 4 we present the dependence of the time of eigenvalue solver on the number of levels and on the number of eigenvectors. Based on the presented numerical results we can see that the FMG-EV(p) method has an nearly optimal computational complexity, i.e., the time of the whole iterative process does not depend on the number of unknowns on the finest level and does only slightly depend on the number of level used. Moreover, the whole computational complexity grows linearly with respect to q and ν , and quadratically with respect to p . All results in a good agreement with the theoretical result in (10).

In Figure 5 we compare the method with these parameters, denoted by "real", to the method with $q = 1$, $\nu = 3$, $\varepsilon_{FMG} = 10^{-16}$ and $\varepsilon_{FAS} = 10^{-16}$, denoted by "ideal", from which one can see that our choice is close to optimal from the accuracy point of view and more attractive from the timing point of view. Moreover, one can see that there is a limit accuracy for each pair q and ν , which can not be improved by decreasing ε_{FMG} and/or ε_{FAS} .

Now we want to mention a problem with the eigenvector orthogonalization for the inner nonlinear method. Indeed, when we try to solve the coarse level problem for the second, third or other eigenvectors during the FAS method with some given accuracy, we always perform the maximal number of iterations, i.e. we could not reach the desired accuracy. It seems that it is not a strong restriction on the suggested method since the method is still convergent. However, the proposed technique with corresponding block-type modifications for the inner nonlinear method can be done. Another open questions are the method for computing the sequence of matrices $A^{(k)}$ and also the proof of the convergence rate of the method. The future investigations will be directed in these ways.

As a final conclusion, we have found that the FMG-EV(p) method, applied for computing

the smallest eigenvalues and their eigenvectors, leads to an iterative eigensolver with an nearly optimal order of the computational complexity and an uniform convergence behaviour. Moreover, the experimental study of the FMG-EV(p) method for linear elasticity problems give us the same results [6].

References

- [1] A. Brandt, S. McCormick and J. Ruge, *Multigrid methods for differential eigenproblems*, SIAM J. Sci. Stat. Comput., 4(2) (1983), pp. 244–260.
- [2] W. Hackbusch, *Multigrid Methods and Applications*, Springer-Verlag, Berlin - Heidelberg - New York, 1985.
- [3] A. V. Knyazev, *Preconditioned eigensolvers – an oxymoron?*, ETNA, 7 (1998), pp. 104–123.
- [4] A.V. Knyazev, *Toward the Optimal Preconditioned Eigensolver: Locally Optimal Block Preconditioned Conjugate Gradient Method*, SIAM J. Sci. Comput., 23 (2001), pp. 517–541.
- [5] M. Larin, *An optimal multilevel method for computing the smallest eigenpair*, Bull. Novosib. Comput. Center., In: *Numerical Analysis*, 9 (2000), pp. 59–67.
- [6] M. Larin, *On a multigrid eigensolver for linear elasticity problems*, Lecture Notes in Computer Science, 2542 (2003), pp. 182-191.
- [7] B. Parlett, *The Symmetric Eigenvalue Problem*, Prentice-Hall, Inc., 1980.
- [8] U. Trottenberg, C. Oosterlee and A. Schüller, *Multigrid*, Academic Press, 2001.
- [9] P. Wesseling, *An introduction to multigrid methods*, John Wiley & Sons Ltd., 1992.

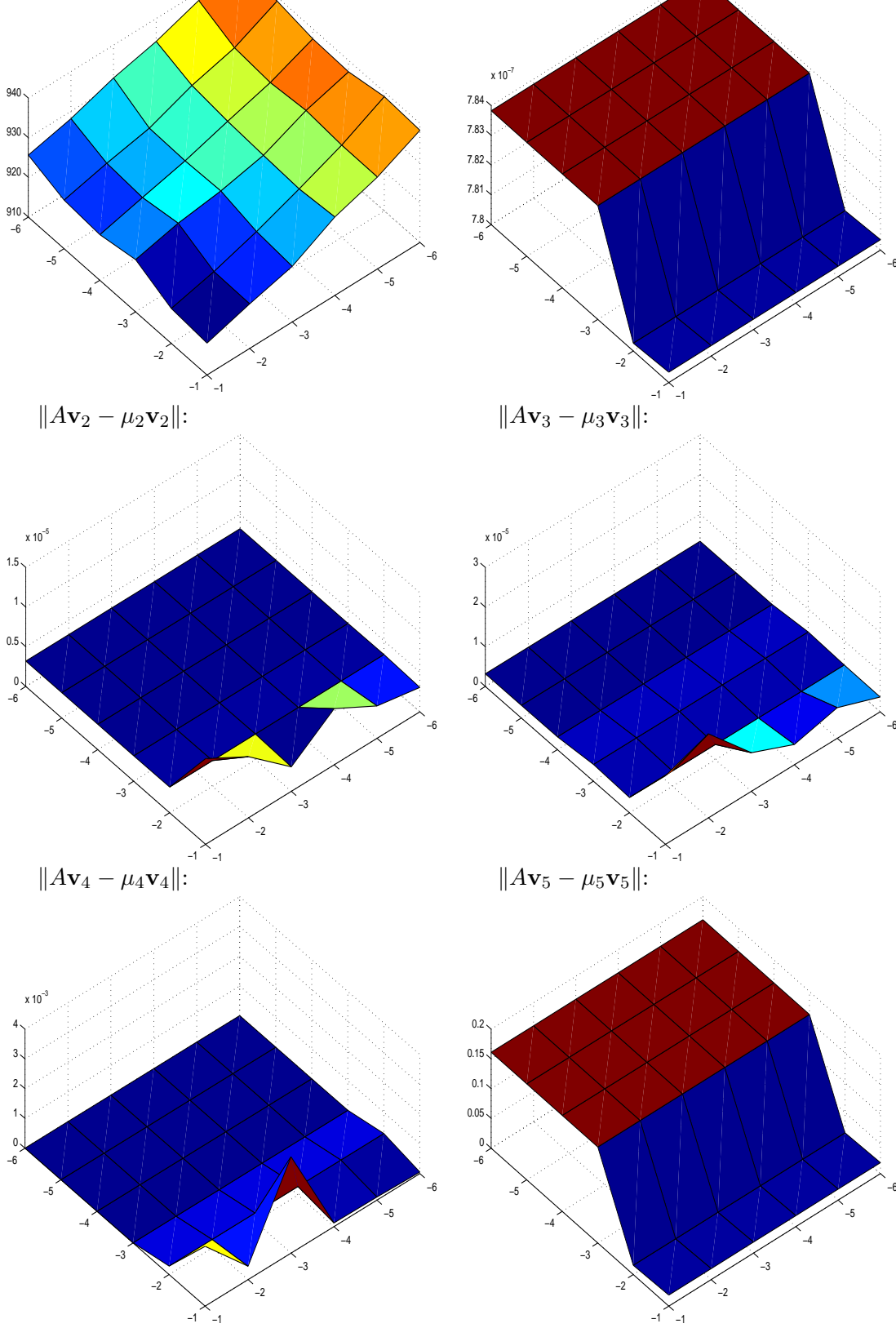


Fig. 1.1. The time of the whole iterative process (Time) and the accuracy for five smallest eigenvalues ($\|A\mathbf{v}_i - \mu_i\mathbf{v}_i\|$) vs. the inner stopping criteria ε_{FMG} and ε_{FAS} , $Llvs = 5$

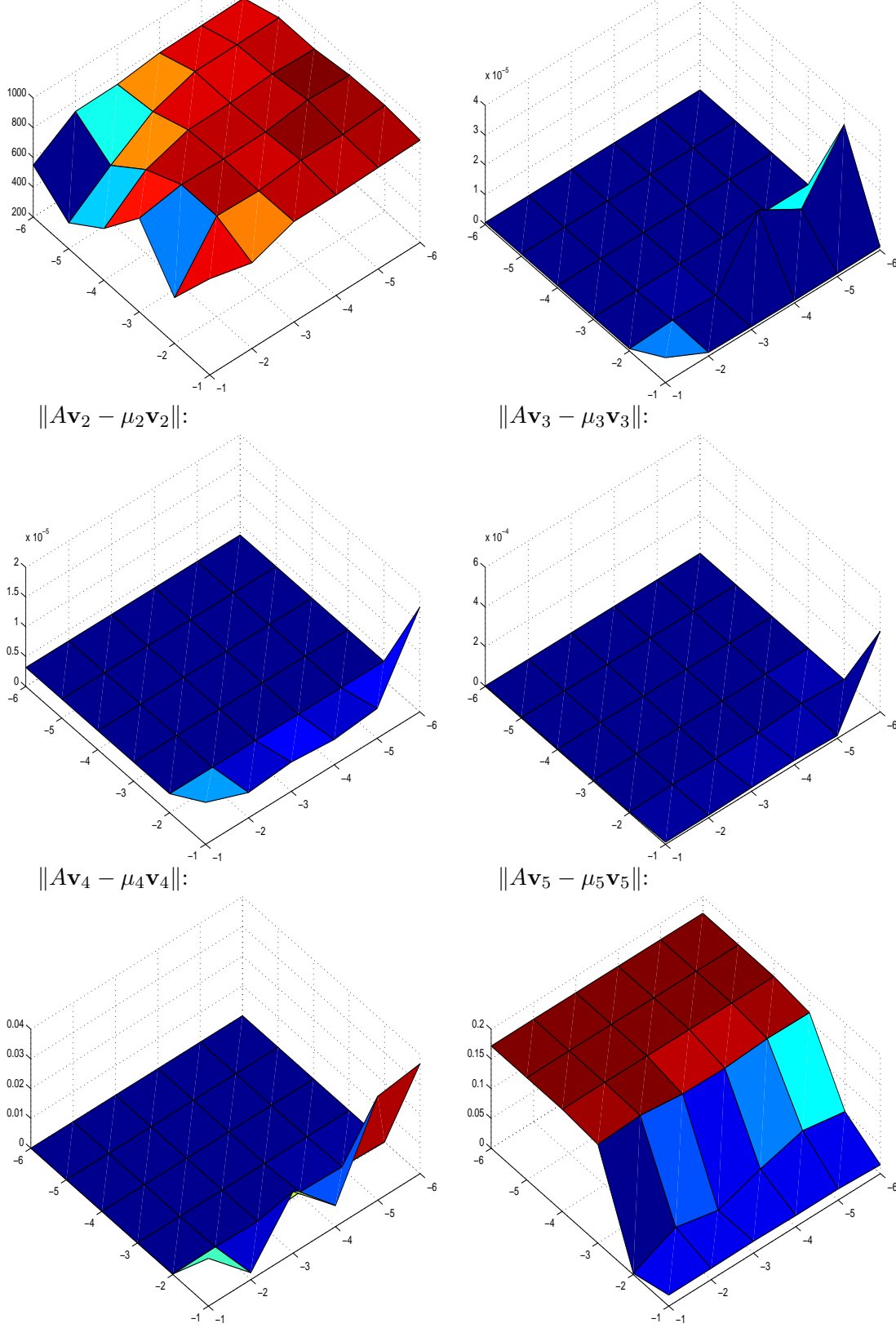
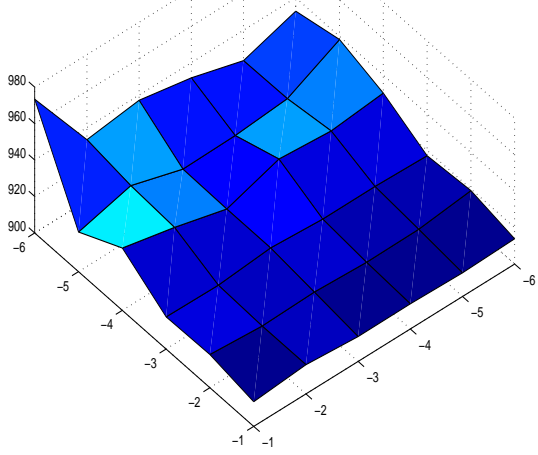
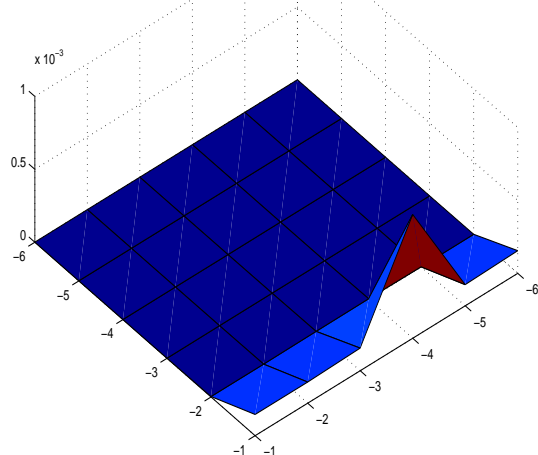


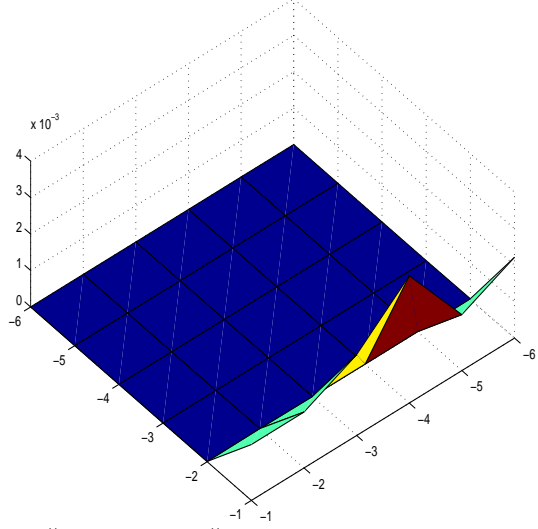
Fig. 1.2. The time of the whole iterative process (Time) and the accuracy for five smallest eigenvalues ($\|A\mathbf{v}_i - \mu_i\mathbf{v}_i\|$) vs. the inner stopping criteria ε_{FMG} and ε_{FAS} , $Lvs = 4$



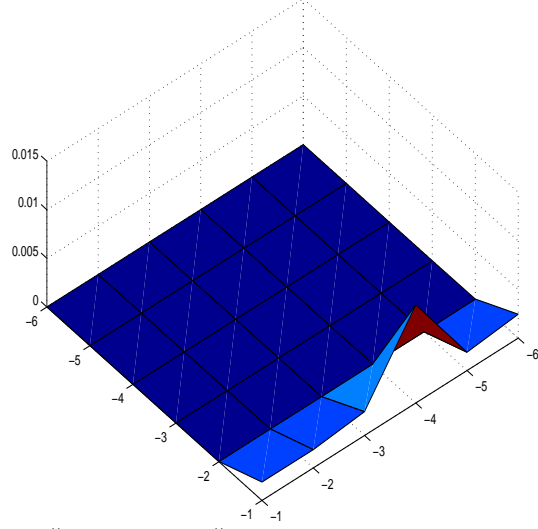
$$\|A\mathbf{v}_2 - \mu_2\mathbf{v}_2\|:$$



$$\|A\mathbf{v}_3 - \mu_3\mathbf{v}_3\|:$$



$$\|A\mathbf{v}_4 - \mu_4\mathbf{v}_4\|:$$



$$\|A\mathbf{v}_5 - \mu_5\mathbf{v}_5\|:$$

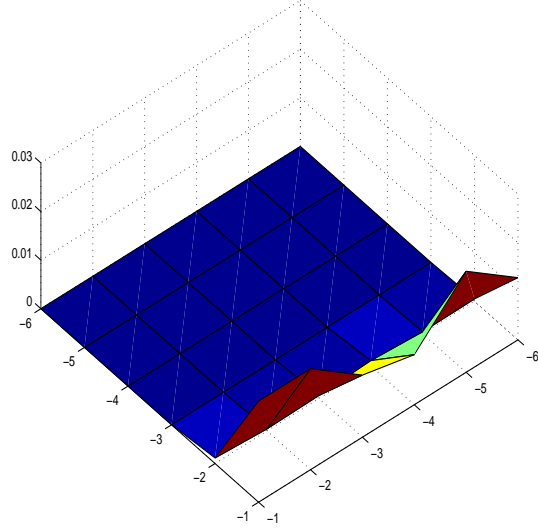
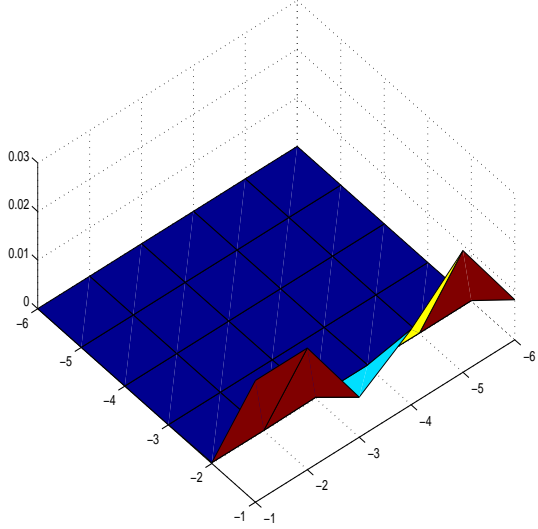


Fig. 1.3. The time of the whole iterative process (Time) and the accuracy for five smallest eigenvalues ($\|A\mathbf{v}_i - \mu_i\mathbf{v}_i\|$) vs. the inner stopping criteria ε_{FMG} and ε_{FAS} , $Lvs = 3$

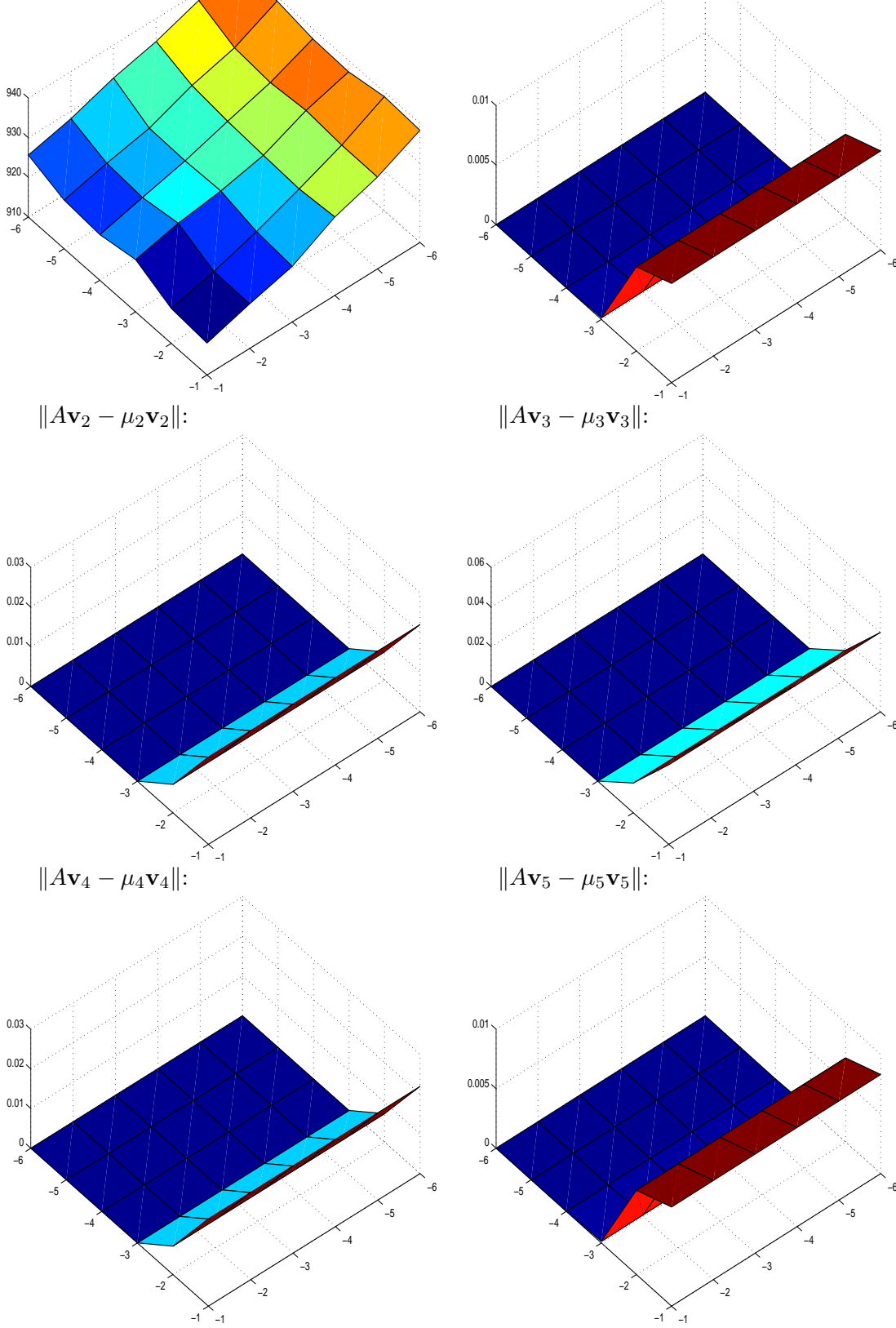


Fig. 1.4. The time of the whole iterative process (Time) and the accuracy for five smallest eigenvalues ($\|A\mathbf{v}_i - \mu_i\mathbf{v}_i\|$) vs. the inner stopping criteria ε_{FMG} and ε_{FAS} , $Lvs = 2$