JOURNAL OF COMPUTATIONAL PHYSICS 111, 89–109 (1994)

The Two-Phase Method for Finding a Great Number of Eigenpairs of the Symmetric or Weakly Non-symmetric Large Eigenvalue Problems

FRANCISZEK A. DUL AND KRZYSZTOF ARCZEWSKI

Warsaw University of Technology, Institute of Aeronautics and Applied Mechanics, ul. Nowowiejska 24, 00-665 Warsaw, Poland

Received July 29, 1991; revised March 17, 1993

Although it has been stated that "an attempt to solve (very large problems) by subspace iterations seems futile" (H. G. Matthies, *Comput. Struct.* **21** (1985), p. 324), we will show that the statement is not true, especially for extremely large eigenproblems. In this paper a new two-phase subspace iteration/Rayleigh quotient/conjugate gradient method for generalized, large, symmetric eigenproblems $Ax = \lambda Bx$ is presented. It has the ability of solving extremely large eigenproblems, $N = 216{,}000$, for example, and finding a large number of leftmost or rightmost eigenpairs, up to 1000 or more. Multiple eigenpairs, even those with multiplicity 100, can be easily found. The use of the proposed method for solving the big full eigenproblems ($N \sim 10^3$), as well as for large weakly non-symmetric eigenproblems, have been considered also. The proposed method is fully iterative; thus the factorization of matrices is avoided. The key idea consists in joining two methods: subspace and Rayleigh quotient iterations. The systems of indefinite and almost singular linear equations $(A - \sigma B)x = By$ are solved by various iterative conjugate gradient/Lanczos methods. It will be shown that the standard conjugate gradient method can be used without danger of breaking down due to its property that may be called "self-correction towards the eigenvector," discovered recently by us. The use of various preconditioners (SSOR and IC) has also been considered. The main features of the proposed method have been analyzed in detail. Comparisons with other methods, such as, accelerated subspace iteration, Lanczos, Davidson, TLIME, TRACMN, and SRQMCG, are presented. The results of numerical tests for various physical problems (acoustic, vibrations of structures, quantum chemistry) are presented as well. The final conclusion is that our new method is usually much faster than other iterative methods, especially for very large eigenproblems arising from 3D elliptic or biharmonic problems defined on irregular, multiply-connected domains, discretized by the finite element (FEM) or finite difference (FDM) methods.

1. INTRODUCTION

We are interested in solving the generalized symmetric eigenproblem

$$Ax = \lambda Bx, \tag{1}$$

where the matrices A and B are symmetric, large ($N \sim 1000$–$100{,}000$), and sparse and where B is positive definite. Several methods are proposed for solving such eigenproblems if a few smallest eigenpairs are required, $M = 10$–20. However, if a very large number of eigenpairs have to be found, these methods are often ineffective or cannot be used at all because of memory limitations.

For problems having moderate dimensions, the methods using the factorization of matrices, such as Lanczos or subspace iteration, are widely used. The Lanczos method is able to compute several extreme eigenpairs very quickly [1–4] at the expense of: factorization of the shifted matrix $A - \sigma B$, solving linear system with high accuracy at each iteration step [5, 6], and the use of an external memory for storing the converged Ritz vectors. The standard subspace iteration suffers from a small rate of convergence [7], whereas an improved efficiency of the accelerated subspace iteration has been attained at the expense of several factorizations of the shifted matrix $A - \sigma B$ [8].

If the dimension N and the bandwidth of matrices m_b are very large, the factorization cannot be performed at all. When only a small number of eigenpairs are required, one can use pure iterative methods, such as TRACMN [6], SIRQIT [9], SRQMCG2 [10, 11], or Davidson [12, 13]. The TRACMN method, developed by Sameh and Wisniewski, seems to be the most efficient one. The Davidson method is very efficient for quantum chemistry problems, but for other problems its effectiveness is not satisfactory [14, 15]. The iterative methods suffer from a small rate of convergence, especially for higher eigenpairs. Divergence of the iteration process may even occur [11]. If a very large number of eigenpairs has to be computed, their effectiveness is not satisfactory.

Other methods, such as transformation (QR or QZ) or search-type methods (determinant search [7], bisection [16, 17]) cannot be used for large problems, because of memory limitation and the unacceptable cost of computation.

0021-9991/94 $6.00

It is seen that the search for better methods is highly desirable, especially for three types of problems:

(1) non-factorizable ones, where the matrices A and B have very large dimensions N and bandwidths m_b;

(2) the number of eigenpairs that has to be computed is large, $M \sim 1000$;

(3) large full eigenproblems that cannot be solved by transformation methods.

In this paper we present the new two-phase, fully iterative method that fulfills all these requirements, taking advantage of subspace iteration and the Rayleigh quotient method. It can find a large number of leftmost eigenpairs, $M \sim 1000$, for very large problems, e.g., $N = 216{,}000$. It can be used for large full eigenproblems that cannot be solved by transformation methods. Also the weakly non-symmetric problems can be solved by it. Although it is especially suited for the problems mentioned above, it is a general-purpose method.

This paper is divided into ten parts. Section 2 contains a description of the main idea of our method with an outline of the algorithm. In Section 3 some computational aspects are discussed, such as an evaluation of the cost and the required memory. In Section 4 we analyze the choice of iterative solver for indefinite linear systems, preconditioning, the choice of parameters, and the strategy of shifting. In Section 5 we present the numerical evaluation of the cost and some features of the method. In Section 6 we present the results of tests for various physical problems, including very large and full eigenproblems. Section 7 contains comparisons with other methods. In Section 8 we present the results for weakly non-symmetric problems. Section 9 contains the description of the disadvantages and the observed failures of our method. Section 10 contains brief information concerning use of the proposed method for computation of the rightmost eigenpairs.

Throughout this paper standard notation will be used. Matrices will be denoted by capital Roman letters, vectors by small Roman letters, and scalars by Greek letters. The norms will be denoted by $\| \ \|$ and the scalar product of two vectors $a, b \in R^N$ by $\langle a, b \rangle$.

2. DESCRIPTION OF THE ALGORITHM

In recent years several methods for solving eigenproblems (1) were proposed. They are often combinations of known methods, chosen in such way that they cancel, or at least weaken, their disadvantages, e.g.,

— the Rayleigh quotient/bisection method developed by Scott [17] is the standard bisection method, speeded up by the Rayleigh quotient iteration;

— the SRQMCG2 method [10, 11] is the combination of the Rayleigh–Ritz procedure and the Rayleigh quotient minimization method;

— the TLIME method [18] is a combination of the inverse and the Rayleigh quotient iterations;

— the accelerated subspace iteration [8] can be viewed as the combination of the standard subspace iteration with the inverse power method, applied to all the iterated Ritz vectors simultaneously.

The new method developed by us is also a combination of two methods: the subspace iteration and the Reyleigh quotient iteration. If they are used separately, their performances are not satisfactory. The subspace iteration method enables one to obtain a good approximation to several eigenpairs near the current shift very quickly, but it is too slow if high accuracy is required [8]. The Rayleigh quotient iteration is very effective, due to its cubic rate of convergence, but it requires a good initial approximation to the eigenpair; otherwise instabilities may occur [18, 19]. We note that both methods have complementary features.

The main idea of the proposed method is as follows. The iteration process consists of two phases: subspace iteration followed by Rayleigh quotient iteration. The Ritz vectors obtained in the subspace iteration phase are used as the initial approximations for the Rayleigh quotient iteration. If the dimension K of the working space used in the subspace iteration is large enough, then good initial approximations to some eigenpairs will be obtained very quickly. Then the Rayleigh quotient iteration can be performed for these Ritz vectors to obtain the required, high accuracy. After the eigenvectors have been computed, the subspace iteration phase can be repeated with the new Ritz vectors.

Now we can outline the algorithm of our method.

(1) Choose: K linearly independent vectors (K given) $[x_1^0, x_2^0, ..., x_K^0]$;

Set their flags to "NONCONVERGED."

Repeat $n := 1, 2,$ until all M required eigenpairs are converged:

(subspace iteration phase)

(2) Choose the shift σ^n for subspace iteration;

(3) Solve K systems of linear equations by some iterative method

$$(A - \sigma^n B)\, y_j = B x_j^{n-1}, \qquad j = 1, ..., K \tag{2.1}$$

with prescribed accuracy of residuals

$$\| r_j^n \| = \| (A - \sigma^n B)\, y_j - B x_j^{n-1} \| < \tau_1 \qquad (\tau_1 \text{ given}) \tag{2.2}$$

(4) Perform the Rayleigh–Ritz procedure

(a) $U = Y^T A Y$, $V = Y^T B Y$,

where $Y = [y_1, y_2, ..., y_K]$ (2.3)

(b) $UQ = VQ \operatorname{diag}(\lambda_1, \lambda_2, ..., \lambda_K)$, (2.4)

(c) $X^n = YQ$, (2.5)

(5) Perform the Gram–Schmidt orthogonalization for "NONCONVERGED" vectors and compute the Rayleigh quotient for each Ritz vector x_j^n

$$\mu_j^n = \langle x_j^n, Ax_j^n \rangle / \langle x_j^n, Bx_j^n \rangle. \quad (2.6)$$

(6) If there exist $1 \leqslant i \leqslant K$ that

$|\mu_i^n - \mu_i^{n-1}|/|\mu_i^n| < \varepsilon$ (ε given) (2.7)

then

switch the process to the Rayleigh quotient phase—go to step 7;

else

repeat the subspace iteration—go to step 2;

(Rayleigh quotient phase)

(7) Set $\sigma_i = \mu_i^n$ and solve iteratively the system

$$(A - \sigma_i B) y_i = Bx_i^n, \quad (2.8)$$

(8) Update $x_i^n := y_i / \|y_i\|$, compute $\mu_i^n = \mu(x_i^n)$ and error $e_i = (A - \mu(x_i^n) B) x_i^n$

(9) If $\|e_i\| < \tau_2$ (τ_2 given) then (2.9)

(10) (a) Set the flag of the converged eigenvector to "CONVERGED,"

(b) Remove it from the working subspace,

(c) Supply the working subspace with the new Ritz vector x_k;

(10) else repeat the Rayleigh quotient iteration —go to 7;

end

Remarks. 1. In step 1 we have to choose K initial approximations to the eigenvectors. The standard approach is suggested; see Section 4.4.

2. In step 2 we have to choose the shift σ^n for the subspace iteration phase in such way as to obtain a quick convergence of the Ritz vectors with minimal cost of computation, without missing any eigenpair (see Section 4.3).

3. In step 3 the K linear (indefinite for $\sigma^n > \lambda_1$) systems (2.1) have to be solved. To avoid factorization we have to use an iterative method, such as SYMMLQ, MINRES [20], or the standard conjugate gradient algorithm (Section 4.1). The accuracy τ_1 can be moderate; thus the cost of this step will be small.

4. Step 4 is the Rayleigh–Ritz procedure. In substep (2.4) the full $K \times K$ eigenproblem has to be solved, e.g., by the generalized Jacobi method.

5. Step 5 is the Gram–Schmidt orthogonalization, which prevents convergence to the computed eigenpairs. This step is costly when M is large.

6. Step 6 is the switching criterion. If we switch to the Rayleigh quotient iteration too late, then unnecessary subspace iterations are performed; otherwise temporarily missed eigenvalues appear (Sections 4.2, 4.3).

7. Steps 7–9 are the Rayleigh quotient iteration. This is the most time-consuming part of the algorithm. An indefinite, almost singular linear system (2.8) has to be solved by SYMMLQ or MINRES. Surprisingly, the standard conjugate gradient method can be used as well, see Section 4.1.

8. In step 10 the "front" of the computations is moved forward. Dimension K of the working subspace is kept constant. The idea is the same as that in [8].

Some Expected Features of the Algorithm. 1. The proposed method ought to be able to find a large number of eigenpairs effectively, because it joins the advantages of the subspace iteration (quick finding of initial approximations to the eigenpairs) and the Rayleigh quotient method (fast convergence when the initial approximation is good).

2. In opposition to the standard subspace iteration, where the computations have to be continued until a required (high) accuracy of the eigenpairs is obtained, in our method only the most efficient phase of the subspace iteration is exploited. Thus, the cost of the subspace iteration phase will be small.

3. A quick convergence of the Ritz vectors during the subspace iteration phase, which depends on the ratio $\lambda_i/\lambda_{i+K+1}$ [8], can be attained by choosing the dimension K of the working subspace to be large enough, which makes $\lambda_i/\lambda_{i+K+1}$ small.

4. The subspace iteration method finds multiple eigenvalues without difficulty. Thus we expect the same good behavior of our method in such a case, see Section 5.2.

5. The use of preconditioning should speed up the convergence of the iterative solver used in steps (2.1) and (2.8).

6. The algorithm is easily vectorizable, see Section 3.

The Similarities to Other Methods. To our knowledge there is no method that is exactly the same as that presented above. Several similarities to other methods can be found, however. The similarity to the accelerated subspace iteration method [8] lies in the fact that the solution of the linear systems performed there resembles the second phase of our algorithm, but the subspace iteration uses inverse power

iteration (with a direct solver), instead of the Rayleigh quotient iteration (with an iterative solver) used by us.

The TLIME method [18] is similar to the Rayleigh quotient phase of our method, because it joins the inverse power and the Rayleigh quotient iterations with the SYMMLQ, which is used for solving an indefinite system.

The use of the iterative method for solving indefinite systems of equations has been proposed by Ruhe and Wiberg [21], who have used the conjugate gradient method within the inverse power iteration. Paige and Saunders [20] have used their SYMMLQ and MINRES methods for this purpose.

3. COMPUTATIONAL ASPECTS

In this section we discuss: (1) the cost of computations, (2) memory requirements, (3) the autonomy of the code, and (4) the vectorizability.

(1) An evaluation of the cost of our method is obtained under the assumption that the cost of an iterative algorithm used for solving an indefinite and almost singular system of linear equations is of order $O(N^{1+\alpha})$, where $\alpha < 1$ (see Section 5.1). The cost of computation of M eigenpairs can be evaluated as

$$C_M = C_{Ax}\left(n_{RQ}N^{\alpha}\log(\tau_2^{-1})\sum_{i=1}^{M} p(\lambda_i) + Mn_{SI}(KN^{\alpha}\log(\tau_1^{-1}) + r_0 M/2)\right) + Mn_{SI}(10K^3), \tag{3}$$

where

- C_{Ax} the cost of matrix–vector multiplication, equal to $m_0(A)N$,
- $m_0(A)$ the mean number of non-zero elements per row in matrix A,
- $n_{SI}(\varepsilon)$ average number of the first phase steps per eigenpair,
- n_{RQ} average number of the Rayleigh quotient steps per eigenpair,
- r_0 the ratio of mean number of non-zero elements per rows for matrices B and A, $r_0 = m_0(B)/m_0(A)$.

The $p(\lambda_i)$ is the convergence factor of the eigenpair, defined as

$$p(\lambda_i) = \xi_p \sum_{j=1}^{N} |\lambda_j - \lambda_i|^{-1} + \eta_p, \qquad \lambda_i \neq \lambda_j, \tag{4}$$

where ξ_p and η_p are the constants. Its value depends strongly on the distribution of the eigenvalues in the spectrum. Formula (3) cannot be proven exactly, but it has been confirmed entirely by the numerical experiments.

The values of parameters: K, ε, τ_1, n_{RQ}, and n_{SI} are independent of the concrete problem, whereas N, m_0, r_0, α, and $p(\lambda)$ are problem-dependent. Moreover, K, n_{SI}, and n_{RQ} do not depend on N, M, usually: $n_{SI} = 0.4$–1.0, $K = 5$–15, $n_{RQ} = 2$–5; thus we can assume they are of order $O(1)$. It is difficult to evaluate $p(\lambda)$ theoretically, but experiments show that $p(\lambda_i) \sim O(10)$, for $i = 1, ..., M \ll N$. If matrix B is consistent we have $r_0 \sim 1$ and the asymptotic evaluation of the cost is

$$C_M \sim O(m_0 N^{1+\alpha} M) + O(m_0 N M^2) + O(MK^3). \tag{5}$$

We see, that for moderate M the cost depends mainly upon N^{α}, whereas for large M the cost of orthogonalization will be significant also. The cost of the Rayleigh–Ritz procedure is negligible in all cases.

(2) We need a memory for eigenvectors, working vectors, and matrices A and B. Thus, the maximal amount of memory required by our method is equal to

$$M_M = M_W + NM = N(2m_0 + 7 + K) + 5M + NM. \tag{6}$$

The amount of additional working memory depends on the linear solver used, varying between $4N$ and $7N$. Note that the working memory M_W is relatively large if we are searching for a few eigenpairs; otherwise it is negligible. This suggests another version of our method, where the converged eigenvectors are stored in the external memory. This approach may be useful when we must solve a problem that cannot be entirely placed in a working memory. Note, however, that I/O operations will slow down the computations significantly.

(3) The main part of the algorithm can be built up, in principle, as the "black box." It follows from two features of our method:

— the parameters τ_1, ε, and K are almost independent of the problem;

— the forms of the matrices A and B are arbitrary; only the results $y = Ax$, $z = Bx$ for given vector x are required. If the preconditioner P is used, $y = P^{-1}x$ has to be computed. These operations can be performed within the user-defined procedures, independent of the other procedures of the method.

(4) Our algorithm is ideally suited for array or parallel computations, because each system of linear equations from among (2.1) and (2.8) is solved independently. Keeping in mind that the solution of linear equations is the most time-consuming part of our method, we expect that its vectorized version will be very effective.

4. THE ANALYSIS OF THE METHOD

The following questions will be discussed in this section:

(1) choice of the iterative solver for systems of linear equations;

(2) the influence of parameters on the cost and completeness of the computed spectrum;

(3) the shifting strategy and the problem of missing the eigenpairs;

(4) the orthogonalization strategy;

(5) choosing the initial Ritz vectors;

(6) preconditioning the iterative linear solver.

The theoretical results concerned these questions will be published later.

Two types of tests were performed: analytical and "real" ones. As the basic analytical tests we have chosen:

— elliptic. The 2D and 3D Laplace problems,

$$-\nabla^2 u - \lambda u = 0, \tag{7}$$

defined on a rectangle $a \times b$ or a rectangular prism $a \times b \times c$ with Dirichlet ($u|_\Gamma = 0$), Neumann ($\partial u/\partial n|_\Gamma = 0$), or mixed boundary conditions;

— biharmonic. The 2D plate problems,

$$\Delta^2 u - \lambda u = 0, \tag{8}$$

defined on rectangles $a \times b$ with both simply supported ($u|_\Gamma = 0$, $\partial^2 u/\partial n^2|_\Gamma = 0$) or clamped ($u|_\Gamma = 0$, $\partial u/\partial n|_\Gamma = 0$) boundaries.

If they are discretized by FDM, the exact spectra of the counterpart matrix problems are known, which enables us to check many questions precisely. Besides analytical tests we have performed real ones, e.g., for the elliptic problem (7) defined on 2D or 3D irregular, multiply connected domains, discretized by FEM. Also the biharmonic problems were considered (see Section 6.1).

4.1. *Choice of the Iterative Solver for Systems of Linear Equations*

The indefinite, almost singular systems of linear equations can be solved by: (1) SYMMLQ and MINRES, (2) the standard conjugate gradient method applied to normal equations, (3) the standard conjugate gradient method.

(1) The Lanczos-type methods SYMMLQ and MINRES, developed by Paige and Saunders [20], are widely used for solving the singular problems [18, 22]. The use of them for solving (2.1) and (2.8) is highly recommended. A suitable preconditioning will increase their efficiency; see Section 4.5.

(2) The standard conjugate gradient method can be used for the normal equation instead of (2.8), that is, for $(A-\sigma_i B)^{\mathrm{T}}(A-\sigma_i B)\, y_i = (A-\sigma_i B)^{\mathrm{T}} B x_i^n$, whose matrix is positive definite for all shifts. Its conditioning is worse than that of $A-\sigma_i B$, however. Also the lack of efficient preconditioners for normal equations is a disadvantage of this approach [23].

(3) The use of the standard conjugate gradient method for solving (2.1) and (2.8) directly is possible. We have discovered that the conjugate gradient method has the property called by us "self-correction towards the eigenvector": the eigenvector attracts the conjugate direction vectors if they are sufficiently close to it.[1] Thus, the danger that some coefficients of the conjugate gradient method will be undefined is unlikely. In fact, we used the standard conjugate gradients algorithm successfully for all types of eigenproblems and it never broke down. We have to point out that this conclusion is valid only for indefinite problems whose solutions are very close to some eigenvector. For other indefinite problems the "self-correction" property is not preserved and the conjugate gradient method may break down. We have also checked that the use of hyperbolic pairs, suggested in [24, 25], slows down the convergence of our method.

The comparison of various solvers is presented in Fig. 1. Two methods give comparable results: MINRES and SYMMLQ (Figs. 1B, D). The standard conjugate gradient method without preconditioning is less efficient, especially for higher eigenpairs, but if we use the SSOR preconditioning it is almost as efficient as SYMMLQ or MINRES (Fig. 1C). The CG–Gauss method is less efficient than other methods; thus we do not recommend it (Fig. 1A). We have obtained the best results by the modified MINRES method.

4.2. *The Influence of Parameters on the Cost and Completeness of the Computed Spectrum*

The influence of parameters on the behavior of our method will be analyzed based on problem (7). Two aspects are especially important: (1) the influence on the cost of computations and (2) the influence on the completeness of the computed part of the spectrum. In Table I we show the history of five runs of our algorithm with various combinations of parameters ε, τ_1, and K: standard ones ($\varepsilon = 0.05$, $\tau_1 = 5 \times 10^{-2}$, $K = 10$, run 1), weak switching criterion ($\varepsilon = 0.5$, run 2), large dimension of working subspace

[1] We have to point out, that the mentioned property should not be mistaken for that discovered by Paige, which states, that *in finite precision* the Lanczos vectors go to the most converged Ritz vector. In infinite precision it does not hold, however, cf. [2, Eq. (13.4.6)], whereas the "self-correction" property of the conjugate gradient method does not depend on the precision of the arithmetic and concerns the *conjugate direction vectors* (which do not have counterparts in the Lanczos method) rather than the residuals (the CG counterparts of the Lanczos vectors).

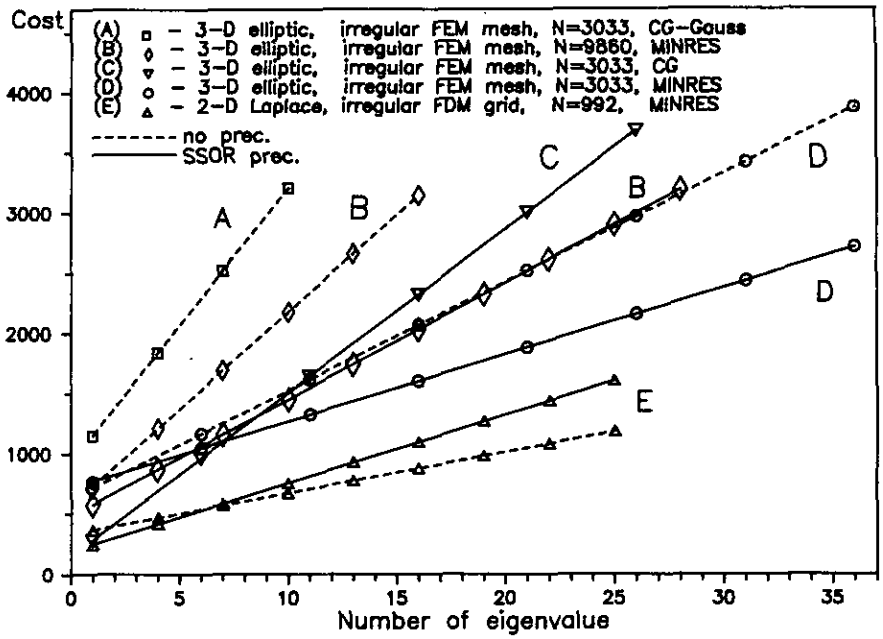

FIG. 1. Effectiveness of various solvers and preconditioning. The cost is expressed by the number of matrix–vector multiplications.

($K=18$, run 3), small $K=2$ (run 4), and sharp tolerance ($\tau_1 = 5\times 10^{-4}$, run 5). For every five iteration steps we present the number of computed eigenpairs M_n, the cumulative cost of computations C_M^n (measured by the number of matrix–vector multiplications), and the number of temporarily missed eigenpairs t_n, defined as follows: let $\lambda_{\max}^n$ be the largest computed eigenvalue; if there are eigenvalues $\lambda_j < \lambda_{\max}^n$ that have not been found yet, there are temporarily missed (see Fig. 2).

TABLE I

A History of Computations for Problem (7), $N=343$, $\tau_2 = 10^{-8}$

	1			2			3			4			5		
No. of iter.	Standard			$\varepsilon=0.5$			$K=18$			$K=2$			$\tau_1 = 5\times 10^{-4}$		
n	M_n	C_M^n	t_n	M_n	C_M^n	t_n	M_n	C_M^n	t_n	M_n	C_M^n	t_n	M_n	C_M^n	t_n
5	1	618	—	8	1809	1	1	1124	—	2	303	6	1	995	—
10	10	2633	—	23	5680	4	15	4943	—	6	1223	30	10	3459	—
15	15	4108	—	43	12505	7	27	8733	—	9	2132	27	17	6020	—
20	21	5954	—	62	19907	7	36	12286	—	12	2984	24	25	8738	—
25	26	7187	1	87	30073	19	47	15900	2	16	4143	20	31	11091	—
30	36	10767	2	99	35277	20	58	20888	1	18	4830	18	36	13094	1
35	39	12296	1				73	26719	—	21	5740	15	45	17087	—
40	47	15195	2				80	30378	—	23	6455	13	47	18506	1
45	49	16427	1				93	36044	2	26	7454	10	52	20932	—
50	57	19425	3				97	37909	2	28	8149	8	59	24015	5
55	62	22118	—							31	9408	36	69	28633	1
60	72	25846	—							34	10524	33	76	31646	—
65	77	28021	—							37	11605	30	80	33827	—
70	82	30552	—							41	13206	26	90	38528	2
75	89	33586	3							44	14471	23	96	41863	2
80	97	37424	3							47	15689	20	98	42857	2
85	98	38203	3							49	16540	18			

(1) The cost of computations depends on the parameters ε and K weakly. In runs 1–4 the computations of $M_n = 47$ and $M_n = 98$ eigenpairs required (on the average) $C_{47} = 15{,}500 \pm 400$ and $C_{98} = 37{,}500 \pm 900$ matrix–vector multiplications, whereas in run 5 the costs were significantly higher: $C_{47} = 18{,}500$, $C_{98} = 42{,}900$. Thus, a very small value for τ_1 causes a noticeable growth in cost. However, τ_1 cannot be too large, because it may slow the convergence of the Ritz vectors during the subspace iteration phase.

The large dimension of the working subspace K increases the cost of the computation of the lowest eigenpairs (compare runs 1 and 3, $n=5$), whereas small K enables one to compute them very quickly (run 4). For higher eigenpairs the costs are comparable in both cases. The large switching tolerance ε influences the cost moderately (run 2), whereas very small ε leads to a noticeable growth of cost, because unnecessary subspace iterations are being performed.

(2) The completeness of the computed part of the spectrum depends strongly upon ε and K. If the switching tolerance ε is too large, many temporarily missed eigenpairs appear (run 2). Moreover, there is a danger than some of them will be missed completely; see Section 4.3.

The influence of the dimension of the working subspace K is also strong. The number of temporarily missed eigenpairs is large if K is too small (run 4). When K increases, the number of temporarily missed eigenpairs is reduced significantly (run 1), but they cannot be eliminated entirely. Too large K increases only the cost (run 3). There exists an optimal value of K that assures both the minimal cost of the com-

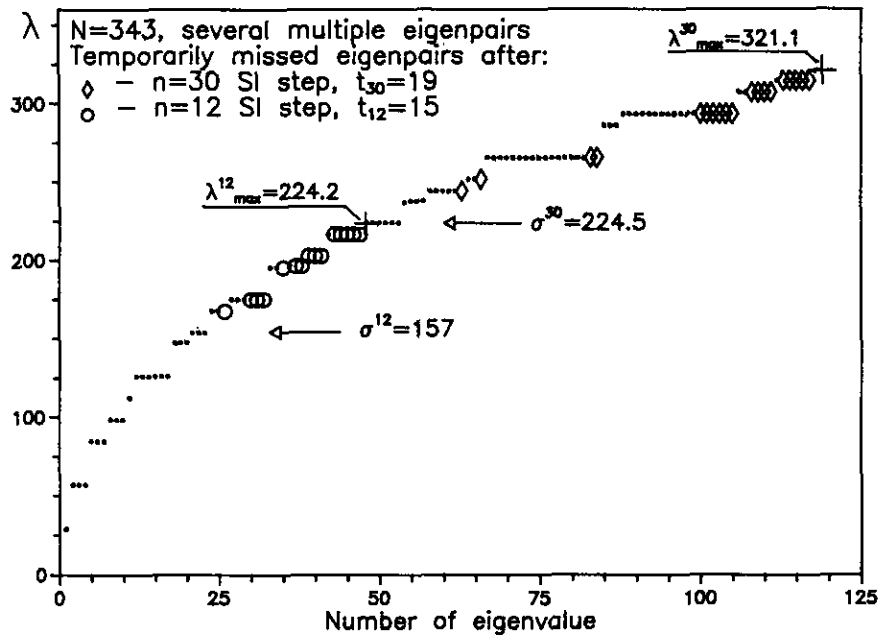

FIG. 2. The temporarily missed eigenpairs for two iteration steps: $n = 12$, $n = 30$.

putations and the minimal number of temporarily missed eigenpairs. This result can be established theoretically.

The influence of the tolerance τ_1 on the completeness of the computed part of the spectrum is negligible (cf. runs 1 and 5).

General conclusions are as follows. The overall cost of computation depends moderately on the values of parameters ε and K, whereas the number of temporarily missed eigenpairs depends on them strongly. Thus, if we want to compute the eigenpairs in the proper sequence, the dimension K should not be small and ε should not be too large. The optimal values of the parameters are not very sensitive to the specific problem; the algorithm can be built up as a "black box."

4.3. *Shifting Strategy and the Problem of Missing the Eigenpairs*

The shift σ^n should be chosen in such way as to obtain good approximations to the eigenvectors as soon as possible. A simplest, but efficient strategy of shifting is to choose $\sigma^n = \delta \cdot \lambda_{\max}^{n-1}$, where $\lambda_{\max}^{n-1}$ is the largest eigenvalue computed after $n-1$ subspace iterations and $\delta < 1$ is the prescribed constant.

A question that is closely related to the shifting strategy, is the problem of missing the eigenpairs. If we increase the shift too quickly, some eigenpairs could be missed completely. The strategy presented above is sufficiently conservative to prevent such events, irrespective of the fact that some eigenpairs are missing temporarily. In Fig. 2 we show two steps of computation for problem (7): $n = 12$ and $n = 30$, for which we have plotted σ^n, λ^n and all the temporarily missed eigenpairs. Note that they lie between σ^n and $\lambda_{\max}^n$. Moreover, all eigenpairs missed temporarily in step $n = 12$ are found before step $n = 30$. The spectrum below σ^n is complete—there are no completely missed eigenpairs. This observation is important, because it suggests a stopping criterion: we should continue the computations until $\sigma^n > \lambda_M$. This criterion is very useful if the Sylvester test cannot be used.

4.4. *The Orthogonalization Strategy*

Step 5 of the algorithm need not be performed at every step of the subspace iteration phase. This leads to a significant improvement of the efficiency, especially for large M. The Ritz vectors ought to be orthogonalized when some of them start to converge to the eigenvector computed previously.

The new Ritz vectors (step 10c) have to be orthogonalized against all computed eigenvectors, which may be ineffective for large M. This suggests including of the step 10c into step 5 to avoid the additional orthogonalization. Another possibility is to use the selective orthogonalization strategy [4].

Orthogonalization of the computed eigenvector (after step 9) is not necessary unless its eigenvalue is equal or very close to some eigenvalues computed previously. In such a case we have to perform the orthogonalization to check whether the new eigenvalue is a multiple or "spurious." Experience shows that this last event happens, but very rarely.

4.5. *Choosing the Initial Ritz Vectors*

We have checked that the standard method, consisting in choosing the random vectors and one constant vector, seems to be quite sufficient for all practical problems [8, 26]; the constant or unit vectors can also be used [11, 13], but they do not show any advantages over the standard approach.

4.6. *Preconditioning of the Iterative Linear Solver*

The preconditioning of eigenproblems was considered by many authors, e.g., [11, 22, 24, 27]. They have shown that it improves the effectiveness of the calculations and sometimes it is even necessary to assure a convergence. We have tested the following preconditioners: diagonal, incomplete Cholesky, and SSOR.

Diagonal scaling has been used with success by Davidson [12, 13]. Unfortunately, this preconditioner is efficient mainly for special types of eigenproblems, arising in quantum chemistry. For other problems it is not so efficient [14, 15]. In our method it always slowed down the convergence; thus it seems to be useless.

The incomplete Cholesky (IC) preconditioner, developed by Meijerink, Van der Vorst [28], and Kershaw [29], is defined as $P = LDL^T = A - E$, where L and D are the lower and diagonal IC factors, whereas E is the unknown error matrix. The IC preconditioners are known to be very efficient, but they require both additional memory and preliminary computations. In our tests it was efficient for the lowest eigenpairs only, breaking down completely for higher eigenpairs because of the negative diagonal elements that had arisen during the incomplete factorization process.

The SSOR preconditioner [24, 27, 30],

$$P = \omega/(2-\omega)(D/\omega - L)\, D^{-1}(D/\omega - L^T), \qquad (9)$$

where L and D are the strict lower triangular and diagonal parts of matrix A ($A = D - L - L^T$) and ω is an acceleration parameter, has important advantages: no additional memory or auxiliary computations are required. The tests show its great usefulness for our purpose.

(1) *Elliptic problems.* The efficiency of the SSOR preconditioner depends on the dimension of problem N. For the 3D elliptic problem defined on an irregular domain we have obtained the following reduction of costs:

— for $N = 3033$, $M = 40$: gain 20% (Fig. 1D); for $M = 100$: gain 35%.

— for $N = 9860$, $M = 20$: gain 25% (Fig. 1B); for $M = 100$: gain 50% (extrapolation).

The use of the SSOR for small or medium problems ($N < 1000$) may lead sometimes to a growth of cost, especially for higher eigenpairs (Fig. 1E).

(2) *Biharmonic problems.* SSOR preconditioning is very useful here, especially for higher eigenpairs and large dimensions. For the plate problem with $N = 400$, the convergence of the non-preconditioned algorithm was poor, whereas the use of the SSOR preconditioner enabled us to solve considerably larger problems: $N = 2404$, $M = 100$ and

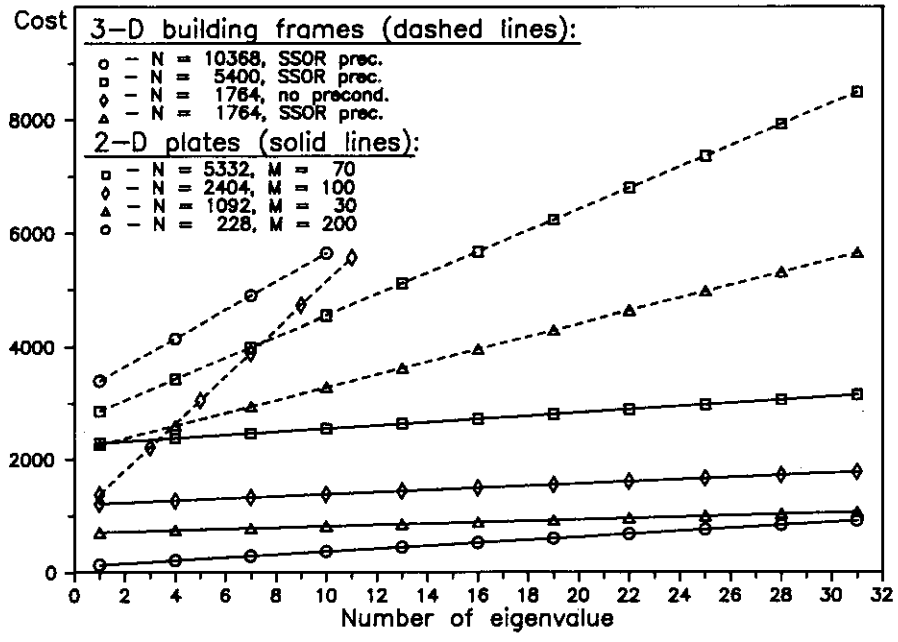

FIG. 3. The cost of solutions of various biharmonic problems. The cost is expressed by the number of matrix–vector multiplications.

$N = 5332$, $M = 70$ (see Fig. 3). Similar results were obtained for the building frame problem ($N = 1764$, $M = 100$), where the convergence of the preconditioned algorithm was much better than that of the non-preconditioned one (Fig. 3). Thus, for large biharmonic problems the use of SSOR preconditioning is highly recommended.

5. THE COST OF COMPUTATIONS

5.1. *The Dependence of Cost upon the Dimensions of the Problem*

We have evaluated the dependence of cost upon the dimension of problem N, assuming that the rate of growth is of order N^{α}. Five types of problems have been used for the estimation of α: 2D and 3D elliptic, discretized by FDM on regular grids and discretized by FEM on irregular meshes, as well as a biharmonic, simply supported plate. The dimensions: $N = 100$–$100{,}000$ for elliptic and $N = 100$–6000 for biharmonic problems enhance the reliability of the evaluations.

The results are presented in Fig. 4. We see that for 3D elliptic problems, $\alpha \sim 0.2$, almost irrespective of the regularity of the mesh. For the 2D cases the influence of discretization is more significant: $\alpha \sim 0.52$ for irregular mesh and $\alpha \sim 0.31$ for a regular one. For the biharmonic problem α is close to 0.7. Note that the values of α decrease for higher eigenpairs.

The conclusion is that the growth of cost, being comparable to that of solving the systems of linear equations [cf. 30], is acceptable.

5.2. *The Dependence of Cost upon the Distribution of Eigenvalues*

The cost of computations depends on the distribution of eigenvalues in the spectrum. Based on formula (4), we can explain the following questions: (1) the distribution of mean cost, (2) the cost of calculation of very close eigenpairs, (3) the influence of discretization on the cost.

(1) The main result we have obtained theoretically is that the cost of the computation of the eigenpair is proportional to the function (4). A typical distribution of the cost is presented in Fig. 5 for the 3D Laplace full eigenproblem ($N = M = 512$). Note the excellent agreement between the theoretical prediction (solid line) and the real mean cost (dashed line). Such a distribution of the cost is characteristic for problems whose spectra are not very far from uniform (see also Fig. 8A, where the distribution of the eigenvalues is almost uniform). Although the mean cost of computation agrees with the theoretical prediction, the cost of the computation of individual eigenpair sometimes differs significantly from the value predicted by (4)—there is a "variance" of the cost. Some of the eigenpairs converge slowly, whereas others are computed with small costs, sometimes less than those of the lowest eigenpairs. The variance of cost is larger for eigenvalues having several close neighbors. Another reason for the variance is the random method of generation of the Ritz vectors; we cannot be assured that the eigenvectors will appear in the proper sequence. Note that the number of steps that the iterative solver used for solving (2.8) varies only moderately; see

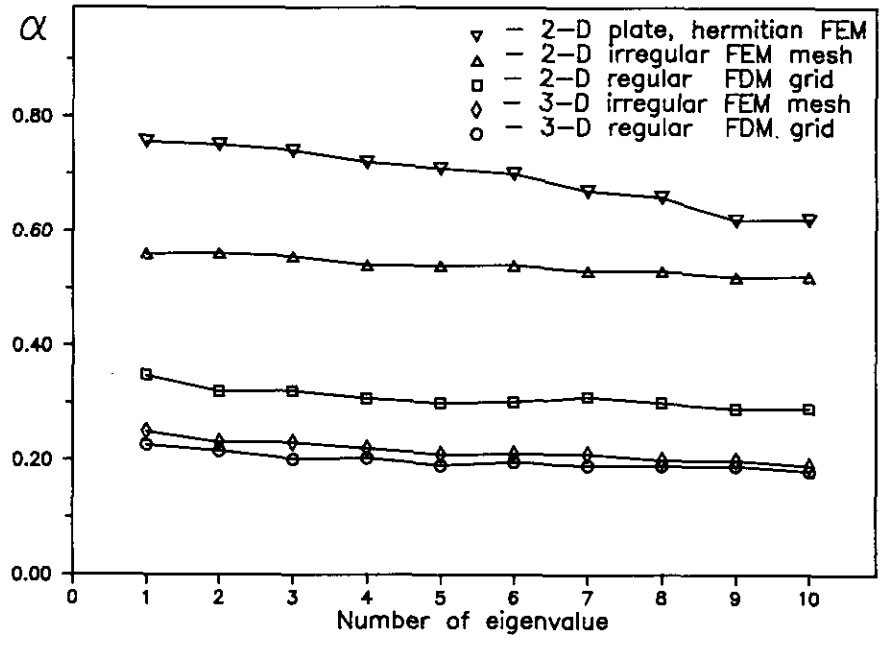

FIG. 4. The rate of growth of the cost for elliptic and biharmonic problems. α is the exponent of cost factor N^{α}.

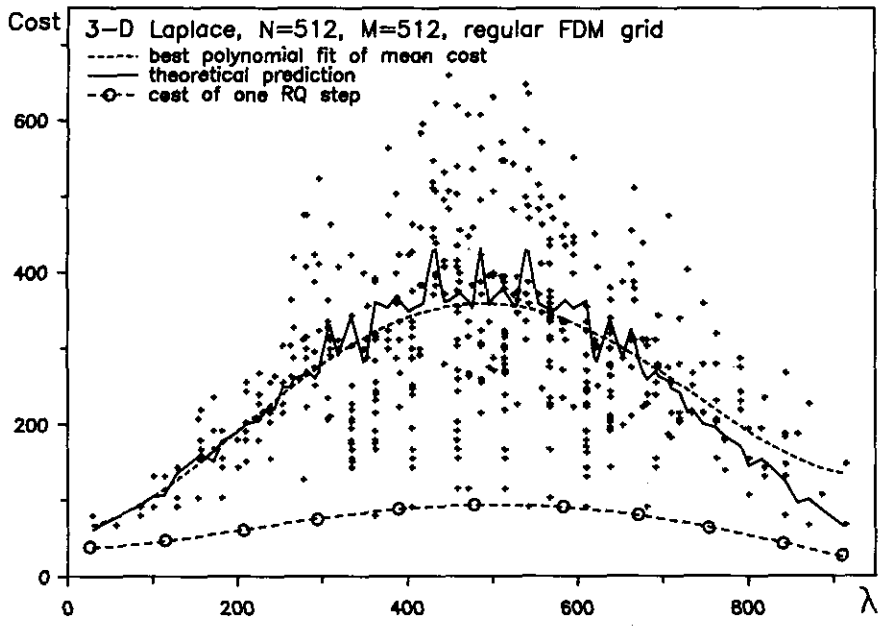

FIG. 5. The comparison of theoretical prediction and results of computations for full elliptic eigenproblem. The cost is expressed by the number of matrix–vector multiplications. The costs of subspace phase and orthogonalization are ommitted.

Fig. 5. This means, that the cost of computation depends weakly on the convergence properties of the solver.

(2) The cost of computations for very close eigenvalues is large because of large values $|\lambda_j - \lambda_i|^{-1}$. If the eigenvalues are multiple or well separated, the cost of computation is smaller. The tests confirm this conclusion entirely; see Fig. 6, where we have plotted the costs of computations for multiple and very close eigenvalues. The multiple eigenvalues are computed without any difficulties. In Fig. 5 almost all the eigenvalues are multiple (multiplicity 3, 6, 12, 21, 24, and 25) and they have been computed correctly, despite the dimension of working subspace, $K = 5$, was five times smaller than the maximal multiplicity.

(3) The distribution of eigenvalues in the spectrum depends both on features of the physical problem (its type, boundary conditions, and shape of domain) and on the method of discretization, which may split multiple eigenvalues of the continuous problem. To explain it, let us consider two cases:

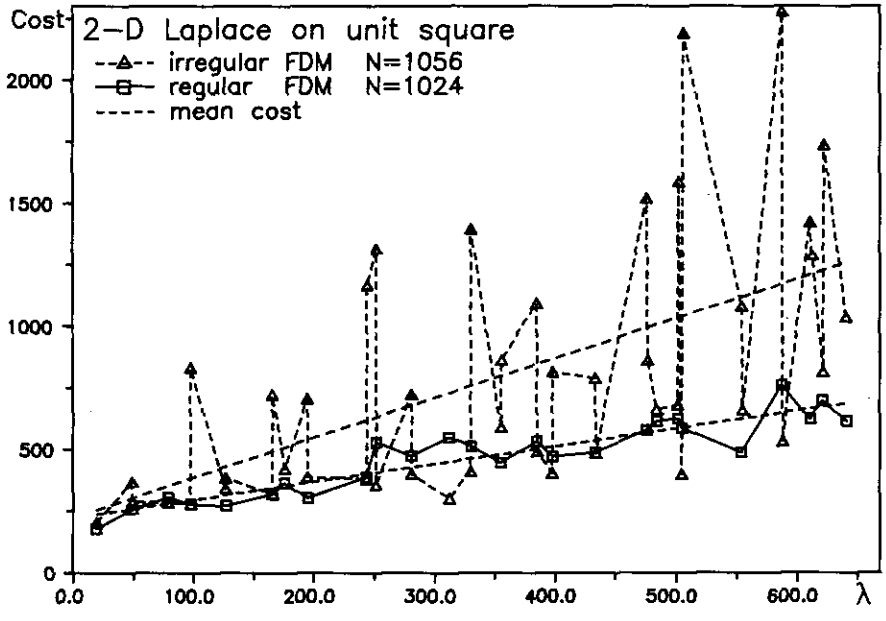

FIG. 6. Influence of splitting of eigenvalues on the cost of computations. The cost is expressed by the number of matrix–vector multiplications.

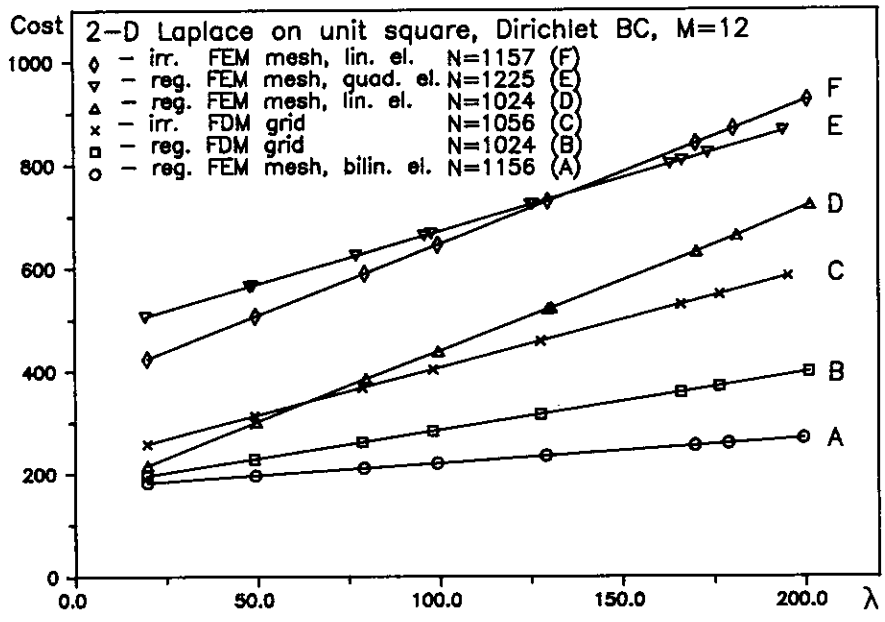

FIG. 7. Influence of discretization on the cost. The cost is expressed by the number of matrix–vector multiplications.

— problem (7) discretized by FDM with $h_x = h_y$ (multiple eigenvalues) or $h_x \neq h_y$ (close, but not multiple eigenvalues);

— problem as above, but discretized by linear or quadratic finite elements, both on regular and highly irregular meshes.

The results are presented in Figs. 6 and 7. The conclusions are as follows:

— If the eigenvalues are exactly multiple (symmetric domains, regular FDM grid, or bilinear FEM mesh), the costs of finding of consecutive eigenpairs are small, being close to the mean cost; the variance is small (Figs. 7A, B).

— If the domain is symmetric, but the mesh is not strictly regular, the multiple eigenvalues of the continuous problem are split. The distances between them are usually small (Figs. 7C, D, E); thus the mean cost grows moderately and the variance is large (see Fig. 6). If the mesh is highly irregular, the mean cost is larger compared with those of regular meshes (Fig. 7F).

The conclusion is that the cost of computation depends on the distances between the eigenvalues in the spectrum that are dependent to some extent on the discretization.

6. THE RESULTS OF TESTS

6.1. *Various Physical Problems*

Now we will show the results obtained for various physical problems that have been solved by our method during its routine exploitation for over three years.

(1) The computation of acoustic modes in the interior of 3D rooms [31]. These problems are of the elliptic type. Both linear and quadratic finite elements were used. The dimensions were: $N = 956$, 3033, 9842, and 20,186; the densities of matrices were small, $m_0 = 10$–25. Usually $M = 20$–100 eigenpairs were computed. The typical costs of computations are presented in Figs. 1B, C, D.

(2) The mode shapes of plates modeled by the biharmonic equation (8), discretized by the standard hermitian finite elements with 4 degrees of freedom in each node. The dimensions were $N = 100$–6000, densities of matrices $m_0 \sim 40$. The convergence was very good, see Fig. 3. The use of SSOR preconditioning was necessary, especially for higher eigenpairs and large dimensions.

(3) The natural modes of vibration for 3D building frames composed of beams having six degrees of freedom in each node. These problems are of mixed type: biharmonic due to bending and elliptic due to torsion and tension. They are similar to those described in [8], but we solved considerably larger problems, $N = 720$, $N = 1746$, $N = 5400$, and $N = 10{,}368$. The densities of the matrices were $m_0 \sim 50$. The SSOR preconditioning was necessary, as in the plate problems (Fig. 3).

For these problems we have performed a comparison with the accelerated subspace iteration method. For $N = 720$ the accelerated subspace iteration method was better, but for $N = 1746$ our method was more efficient (see Section 7).

(4) The computation of mode shapes of linear substructures for dynamic analysis of large non-linear structures

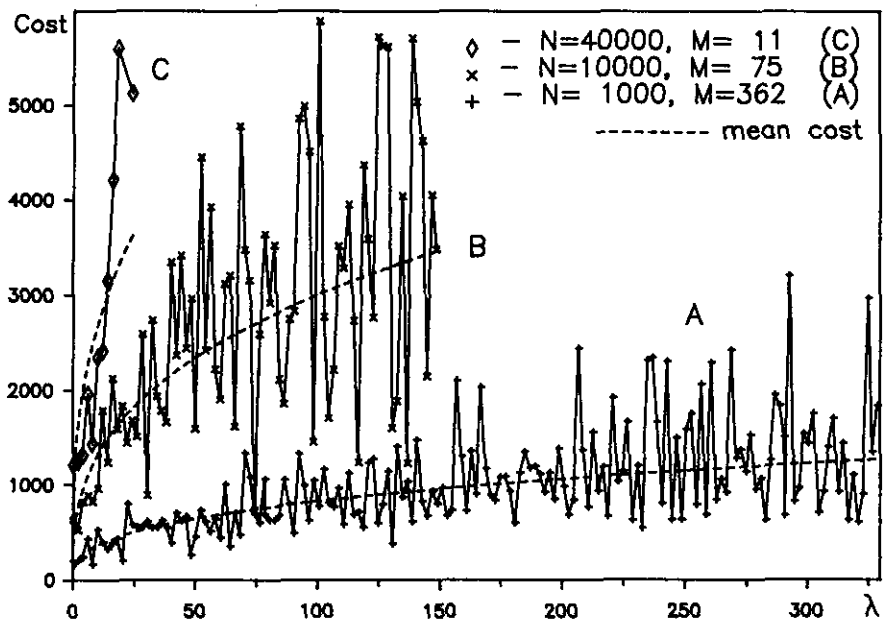

FIG. 8. The results of computaions for various Nesbet matrices. The cost is expressed by the number of matrix–vector multiplications.

[32]. The entire system is composed of several large linear elastic substructures, joined by the small substructures having non-linear damping and elastic properties. The total dimension of the system is of order 10^4. The use of eigenmodes enables us to transform the linear subsystems from physical to modal coordinates, reducing the dimension to 10^2, for which the numerical integration can be performed very effectively. The calculation of eigenmodes for linear subsystems is the most time-consuming part of preliminary computations. The use of our method improves its effectiveness. The dimensions of the linear subsystems were $N = 800$, 1120, 2460, 3847 and $M = 10$–35 eigenmodes were usually computed.

(5) The problem of quantum chemistry concerns the large scale configuration interactions of electronic wavefunctions of atoms and molecules [13]. The matrices $A \equiv X_4$ or X_5 are defined as [13]

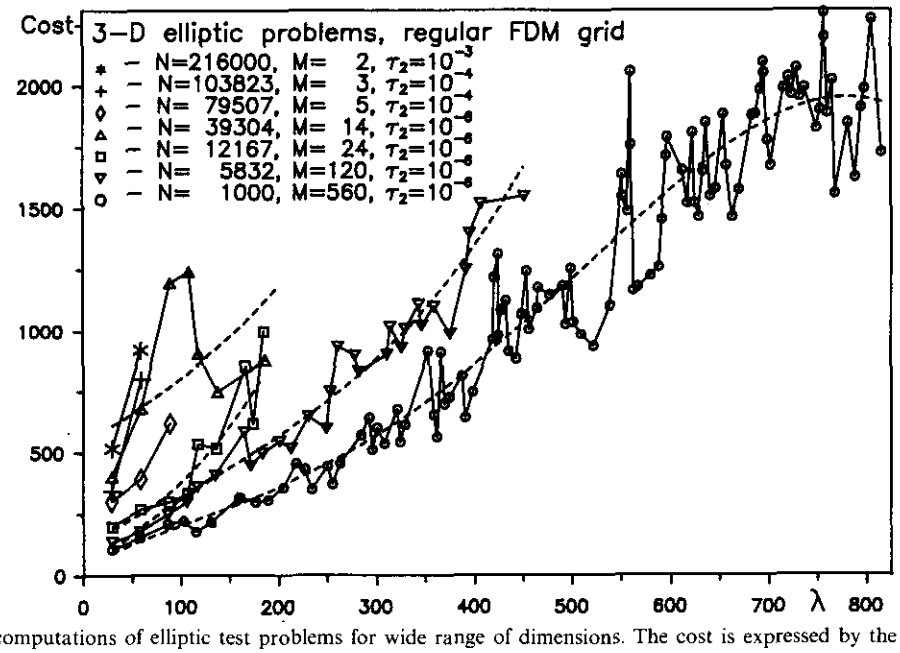

FIG. 9. The cost of computations of elliptic test problems for wide range of dimensions. The cost is expressed by the number of matrix–vector multiplications.

$$X_4: X_{ii} = 2i - 1, \qquad X_{ij} = 1 \text{ if } |i-j| < 50, \qquad 0 \text{ otherwise}; \quad i, j = 1, ..., N, \tag{10}$$

$$X_5: X_{ii} = 1 + 0.1(2i-1), \qquad X_{ij} = 1 \text{ if } |i-j| < 50, \qquad 0 \text{ otherwise}; \quad i, j = 1, ..., N. \tag{11}$$

whereas $B \equiv I$. The matrices are relatively dense, $m_0 = 100$. We have solved (10) for $N = 1000$, $M = 362$ (Fig. 8A), $N = 10{,}000$, $M = 75$ (Fig. 8B), and $N = 40{,}000$, $M = 11$ (Fig. 8C), as well as (11) for $N = 1000$, $M = 250$, with accuracy $\tau_2 = 10^{-6}$. No preconditioning has been used. The mean cost of computation of the consecutive eigenpairs grows moderately due to the almost uniform distribution of the eigenvalues, whereas the variance is large. The behavior of our method is not different, compared to other examples.

6.2. *Very Large Problems*

We have computed a large number of eigenpairs of the elliptic problems of medium size,

3D, regular FDM grid, $N = 8000$, $M = 100$;
$N = 15{,}625$, $M = 50$;

3D, irregular FEM mesh, $N = 3033$, $M = 100$;
$N = 9840$, $M = 50$; $N = 20{,}186$, $M = 25$;

and a small number of eigenpairs for very large 3D problems discretized by FDM,

$N = 27{,}000$, $M = 25$; $N = 39{,}304$, $M = 14$;
$N = 79{,}507$, $M = 5$;

$N = 103{,}823$, $M = 3$; $N = 216{,}000$, $M = 2$.

The results are presented in Fig. 9. They prove the ability of our method to find a large number of eigenpairs (first group), as well as the ability of solving the extremely large eigenproblems (second group). Note that the dimension $N = 216{,}000$ in the last test is almost of order 10^6. The collection of results for various big problems is presented in Fig. 10.

We did not observe any new difficulties in comparison with problems of medium size. The dimensions mentioned above are limited only by the performance of our computers, so we expect that our method can be employed for considerably larger problems as well.

6.3. *Full Eigenproblems*

Problems of this type are solved usually by transformation methods, such as QR or QZ. For large problems they cannot be used because of memory limitations, especially if $m_b \sim N$. The following elliptic full eigenproblems have been solved by our method:

2D, regular FDM grid, $N = M = 210$;

3D, irregular FEM mesh, $N = M = 432$;

3D, regular FDM grid, $N = M = 512$ (Fig. 5);

$N = M = 729$; $\quad N = M = 1000$.

We were able to compute the whole set of eigenpairs without essential difficulties. The most serious problem was the orthogonalization of a large set of vectors. We employed the reorthogonalization to prevent instabilities. The economic strategy of orthogonalization is being investigated now. The conclusion is that our method is not concurrent to the QR or QZ for small problems, but it may be useful for large ones.

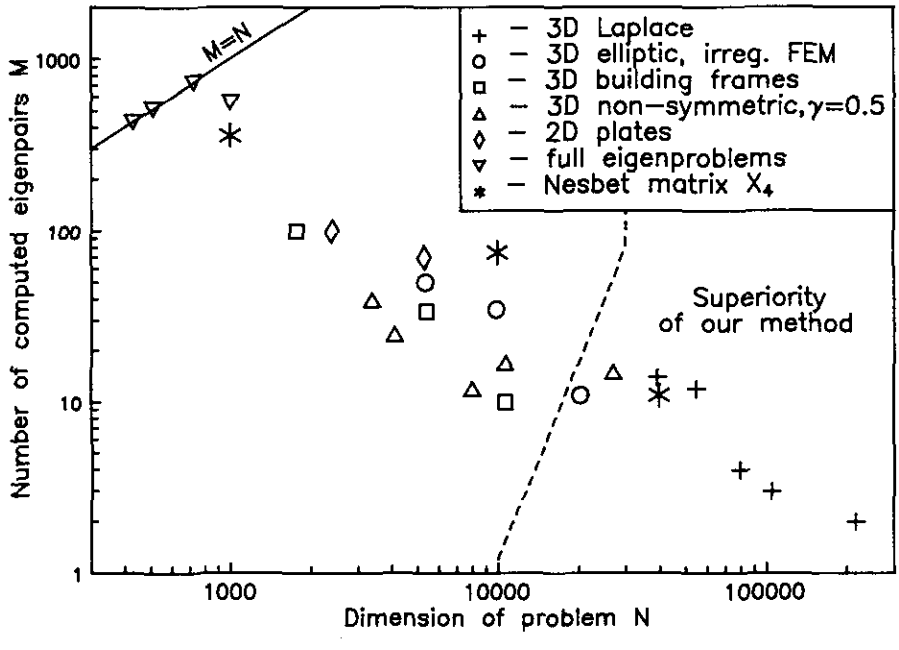

FIG. 10. The compositions of results for various big problems solved by our method.

7. COMPARISONS WITH OTHER METHODS

The comparisons concern mainly the effectiveness measured by the number of arithmetic operations and some important features of the methods. To obtain reasonable comparisons, which do not favor our method, we have assumed the lower evaluation of the costs of concurrent methods. The results of tests for compared methods (except the ASI) have been taken from the literature. Unfortunately, there are few results for large N and M; thus we realize that the comparisons are only tentative and additional tests would be highly desirable.

The following methods have been taken into account: accelerated subspace iteration (ASI) [8], Lanczos [1], Davidson [12–15], SRQMCG2 [10, 11], TRACMN [6], and TLIME [18].

(1) The accelerated subspace iteration method (ASI), used in the ADINA package [8], is one of the best, general-purpose methods. It is the standard subspace iteration applied to the matrix $(A-\sigma B)^{-1}$, accelerated by overrelaxation. Its cost can be evaluated as

$$C_{\mathrm{ASI}} \sim n_f m_b^2 N/2 + n_{\mathrm{SI}}(\tau_2)(2m_b NK + 10K^3) + n_{\mathrm{ort}} m_0 NM^2/2, \tag{12}$$

where $n_f \sim O(1)$ is the number of factorizations, n_{SI} is the mean number of subspace iterations per eigenvalue (evaluated by us as $\sim O(1)$ for $K > M$ and $\sim O(M)$ for $K < M$ [8]), and $n_{\mathrm{ort}} \sim (1)$ [8, p. 322] is the number of orthogonalizations. The cost of the Rayleigh–Ritz procedure is then $O(MK^3)$ for $K < M$ and $O(M^3)$ for $K > M$. Thus the evaluation of the asymptotic cost is

$$C_{\mathrm{ASI}} \sim O(m_b^2 N) + O(m_b NM) + O(m_0 NM^2) + \begin{cases} O(MK^3), & K < M, \\ O(M^3), & K > M. \end{cases} \tag{13}$$

It is seen that the cost depends strongly on factorizations. For large M the cost of the Rayleigh–Ritz procedure is also significant if $K > M$. For Laplace problems (7), discretized by FDM, where $m_b = N^{1/2}$ for the 2D case and $m_b = N^{2/3}$ for the 3D case, we obtain the following comparison of asymptotic costs:

	2D	3D
ASI	$O(N^2)$	$O(N^{7/3})$
Our method	$O(N^{3/2})$	$O(N^{5/4})$

It is evident, that for very large problems our method will be much better than ASI. This conclusion has been verified. In Fig. 11 we present the relative time costs (ratio of the CPU times of ASI and our method) for the lowest eigenpairs. The following problems were compared: (A) the biharmonic, 3D building frame, $N = 600$, $m_b = 126$, $m_0 = 48$; and three elliptic ones; (B) small 2D, $N = 297$, $m_b = 19$; (C) medium 2D, $N = 1225$, $m_b = 36$; and (D) medium 3D, $N = 1320$, $m_b = 111$. We see that ASI is much better for small and medium problems, especially for biharmonic ones, where $m_b \sim m_0$. When the dimensions of problem N and the bandwidth m_b increase the superiority of

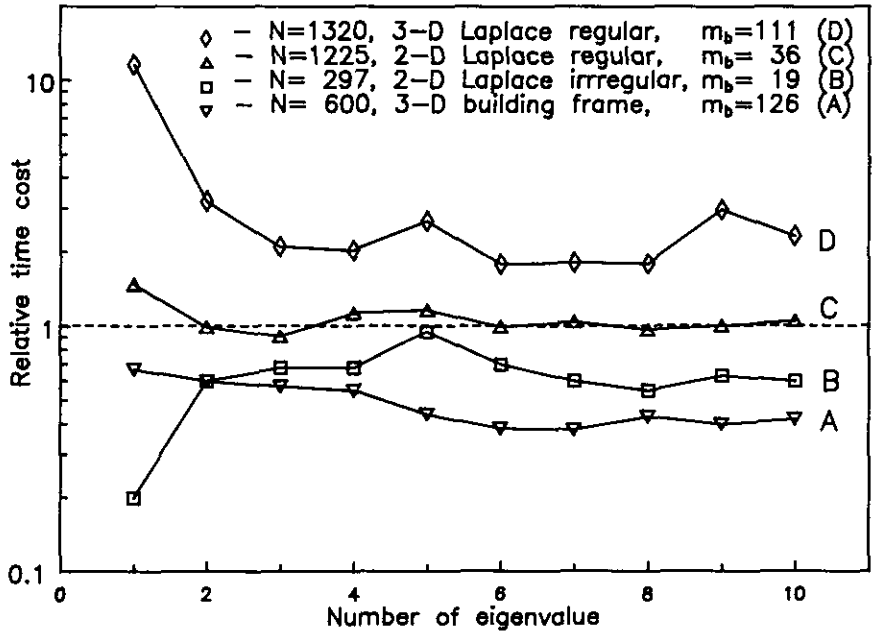

FIG. 11. The comparison of the accelerated subspace iteration with our method.

ASI decreases, disappearing entirely at $N \sim 1300$. It even happened that for $N = 4000$ ASI had not converged at all, because of the errors of factorization (see [30, p. 167; 33]). Our method seems to be better for $N > 2000$. This is especially true for 3D problems, where the bandwidth m_b is relatively large.

(2) The Lanczos algorithm is often considered as the best method for finding a large number of eigenpairs [1–5]. For example, the modified block Lanczos method described in [1] was able to solve the structural eigenproblem with $N = 4479$, $M = 30$ five times faster than the standard subspace iteration for $m_b = 183$, and two times faster for $m_b = 566$. The largest problem solved by this version of Lanczos was $N = 19{,}658$, $m_b = 319$, $M = 180$, which is a very good result. It may then look like the Lanczos method is much better than our method, but a more precise comparison should be based on the evaluation of the cost.

Assuming that the cost of I/O operations can be omitted, the cost of computation of M eigenpairs by the Lanczos method without factorization is

$$C_{\text{Lanczos}} \sim Mn_{\text{it}} m_0 N + n_{\text{ort}} m_0 NM^2/2 + N(Mn_{\text{it}})^2/2, \quad (14)$$

where n_{it} is the number of Lanczos steps per converged eigenpair and n_{ort} is the number of orthogonalizations ($O(1)$ for selective and $O(M)$ for full orthogonalization; we assume $n_{\text{ort}} \sim O(1)$). Due to the extraordinary convergence properties of the Lanczos method, the number of iterations per converged eigenpair is usually small, e.g., $n_{\text{it}} = 2.5$ [3], $n_{\text{it}} = 1.9$ [5], $n_{\text{it}} = 1.7$ [1]; thus $n_{\text{it}} \sim O(1)$. The asymptotic cost of the factorization-free Lanczos method is then

$$C_{\text{Lanczos}} \sim O(m_0 NM) + O(m_0 NM^2) + O(NM^2). \quad (15)$$

Comparison with (5) shows the difference between the first terms; in our method it is equal to $O(m_0 N^{1+\alpha} M) \sim N^{\alpha} O(m_0 NM)$. Since $N^{\alpha} \sim O(10)$–$O(10^2)$, it agrees with the observation that our method has to perform hundreds to thousands of iterations to compute an eigenpair. It is thus evident, that the Lanczos method is much better for these eigenproblems which do not need factorizations like the standard ones (if spectral transformation is not used).

There are two cases when factorization in the Lanczos method is necessary: the generalized eigenproblems and the standard ones if the spectral transformation has to be used for improving the conditioning of the required eigenvalues [3, 36]. Although generalized eigenproblems can be solved by the Lanczos method without factorizations (see below), there is an opinion that all useful variants of the Lanczos method (including its block and band versions) must perform it [5, 14, 34–37]. The cost of the Lanczos method with factorization is (cf. first three rows for UTLA in [34, p. 670])

$$C_{\text{Lanczos}} \sim n_f m_b^2 N/2 + Mn_{\text{it}} 2m_b N + n_{\text{ort}} m_0 NM^2/2 + N(Mn_{\text{it}})^2/2, \quad (16)$$

where $n_f \sim O(1)$ is the number of factorizations; thus the asymptotic cost is

$$C_{\text{Lanczos}} \sim O(m_b^2 N) + O(m_b NM) + O(m_0 NM^2) + O(NM^2). \quad (17)$$

It is seen that the necessity of factorization and the solution of factored systems changes the evaluation completely. For large N and m_b the cost becomes very high, annihilating entirely the benefits arising from the convergence properties of the Lanczos method. In such cases our method will be better. The same conclusions are valid for the shift-and-invert and the block Lanczos methods, if they factorize a matrix.

To illustrate the above conclusions let us consider the 3D elliptic problem $N = 39{,}304$, $m_0 = 7$, $m_b = 1157$. Our method has found $M = 14$ eigenpairs performing 13,980 matrix–vector multiplications, which gives an average of $C_e \sim 2.7 \times 10^8$ arithmetic operations per eigenpair (1 operation = 1 multiplication + 1 addition). The costs of basic matrix operations for this problem are

Matrix–vector multiplication	($\sim m_0 N$),	$C_{Ax} \sim 2.7 \times 10^5$,
Solution of factored system	($\sim 2m_b N$),	$C_{\text{sol}} \sim 9.3 \times 10^7$,
Factorization	($\sim \frac{1}{2} m_b^2 N$),	$C_{\text{Fct}} \sim 2.7 \times 10^{10}$.

It is seen, that our method is able to find $C_{\text{Fct}}/C_e = 100$ eigenpairs before the factorization is finished. Moreover, the cost of solving a factored system is comparable to that of computing one eigenpair in our method ($C_e/C_{\text{sol}} \sim 3$). This means, that for large dimensions, $N > 10^4$, our method will be better than the Lanczos method, and this superiority will increase quickly with increasing N. This is evident when M is small (cf. Fig. 11, D), but our method will probably be better also for large M. Note, also, that the Cholesky factor requires 4.6×10^7 memory units (182 MB in single precision) in the considered case.

We realize that the Lanczos method is more useful for eigenproblems solving currently, because their dimensions are not extremely large. Note, however, that in the 3D case, even coarse discretization leads to a very large dimension. For example, an elliptic problem defined on the cube, whose edges have been divided into 30 intervals, has the dimension $N = 27{,}000$. Since in the future much larger eigenproblems will probably be solved, we think that our method might be useful.

There are two ways of solving generalized eigenproblems

by the Lanczos method without factorizations. The first consists of solving systems of linear equations $Bx = y$ by some iterative solver, e.g., the (preconditioned) CG method. However, the accuracy of such solutions has to be high; otherwise spurious eigenpairs will occur [6]. In this case the cost of M Lanczos steps will be roughly of $m_0 MN^{1+\alpha} \log(\tau^{-1})$, where τ has to be of the order of machine precision [6, p. 1257]. Such cost is comparable to that of our method and the conclusions should be based on numerical tests.

The second way of avoiding factorization in the Lanczos method is to work with the generalized inner product, $\langle x, y\rangle_B = \langle x, By\rangle$ [38]. Such an approach has disadvantages, however: (a) the orthogonalization must be performed in each iteration; (b) the reduced matrix is full rather than tridiagonal. This gives the total cost of order $2m_0 NM + m_0 NM^3/2 + M^4$, which is considerably larger than that of our method. Moreover, solving the generalized eigenproblem inside the Rayleigh–Ritz procedure is a problem itself if M is not small; thus the iterations have to be restarted frequently, which slows down the convergence.

Apart from the necessity of factorization in generalized eigenproblems, there are two other minor disadvantages of the Lanczos method. The first is the loss of orthogonality results in making redundant copies of the eigenpairs, which is still considered as a weak point of the Lanczos method (and also its block version) [39, p. 28]. In our method this problem is not serious. We observed duplications of some eigenpairs but, as opposed to the Lanczos method, this phenomenon is not permanent. The spurious eigenpairs were always eliminated during orthogonalization (step 5 of the algorithm). There was no breaking down or other dangerous phenomena; the only result was a small increase of global cost.

To compute all eigenvectors spanning an invariant subspace by the Lanczos method, one has to perform several runs with different starting vectors, or to use the block Lanczos method with the dimension of the block equal at least to the dimension of this invariant subspace (not known a priori) [35, p. 273; 36, p. 170]. In constrast, our method finds all the invariant subspaces in a single run, independently of initial approximations, due to the "attraction" property of the subspace iteration and the presence of rounding errors. No information on the multiplicities is required. Moreover, the dimension of the working subspace K need not be equal to the dimension of the invariant subspace—it can be smaller without danger of missing any eigenpair, see Section 5.2, p. 97. The maximal multiplicity we have found correctly by our method was 100, whereas the dimension of working subspace was $K = 5$ only.

Finally we would like to state that our method seems to be superior to the Lanczos method for large generalized eigenproblems, especially three-dimensional ones. The Lanczos is better for standard eigenproblems if the spectral transformation is not used, as well as for small and medium generalized eigenproblems.

(3) The Davidson method is the Rayleigh–Ritz procedure applied to the non-Krylov subspace, combined with diagonal [12, 13, 15] or general preconditioning [14]. A superiority of the Davidson method over the Lanczos method for quantum chemistry problems is reported in [12, 13], whereas it is not always observed in other problems [15]. The cost of the Davidson method is

$$C_{\text{Davidson}} \sim n_{\text{it}}(\tau_2)(m_0 NM + m_0 NM^2/2 + 10M^3), \quad (18)$$

where n_{it} is the number of iterations ($\sim O(M)$ for the general case, $N_{\text{guess}} = 0$ [13], based on the analogy of the Lanczos method [12, 14, 15]); thus the asymptotic cost is

$$C_{\text{Davidson}}(N_{\text{guess}} = 0) \sim O(m_0 NM^2) + O(m_0 NM^3) + O(M^4). \quad (19)$$

It is seen that the main part of the cost is that of orthogonalization if $N \gg M$, whereas the cost of the Rayleigh–Ritz procedure is significant for $N \sim M$. For large M the cost of orthogonalizations and the generalized Rayleigh–Ritz procedure will be significant [14]; thus the Davidson method may be worse in this case. We suppose, however, that the Davidson method may be better for standard eigenproblems, especially when a small number of eigenpairs is sought. Unfortunately, we do not know the results of solving large generalized eigenproblems by the Davidson method; thus quantitative comparison with our method is not possible. We compared, thus, our method with the results of [13, Table VIII], for matrix X_4, $N = 1000$, $N_{\text{guess}} = 0$, unit starting vectors, $\tau_2 = 10^{-5}$ (which is equivalent to $q_k^2 < 10^{-10}$ in [13]). The results for various M are given in Table II.

It is seen, that the Davidson method is significantly better for the considered problem. This agrees with the conclusion that the Davidson method, being a variant of the Lanczos method, is much better for standard eigenproblems. On the other hand, the number of orthogonalizations and Rayleigh–Ritz steps is larger for the Davidson method. Comparison with the other results presented in [13] suggest also that the Davidson method is better for the lowest eigenpairs if good initial approximations can be

TABLE II

		$M = 1$	$M = 5$	$M = 10$
Davidson	Matr.–vec.mult.	37–56	105–125	190–210
	Nb orth.	11–12	11–13	9–18
	Nb Rayl.–Ritz	11–12	11–13	9–18
Our method	Matr.–vec.mult.	77	550	1200
	Nb orth.	2	4	7
	Nb Rayl.–Ritz	2	4	7

chosen (e.g., for $N_{guess} = 10$, $\tau_2^0 \sim 10^{-4}$ [13]). Note, however, that for higher eigenpairs such a choice is difficult [13]. Moreover, there is the danger that the Davidson method may break down when it begins with a bad initial subspace [15, p. 59].

The good behavior of the Davidson method is mainly due to preconditioning [13, 14]. However, there is no hint of how to choose an effective preconditioner for a general case. In our method the SSOR preconditioner is general and effective. It would be interesting to know wheter it could be used for improving the Davidson method.

It is known that the behavior of the Davidson method is not so good for other eigenproblems, such as those of nuclear modeling [15]. Nothing is known to us about the application of the Davidson method for structural eigenproblems. Bearing in mind the conclusion of [15], it would be interesting to compare both methods for various types of eigenproblems.

(4) The SRQMCG2 method consists in minimization of the generalized Rayleigh quotient over the set of Ritz vectors by the preconditioned conjugate gradient iterations. According to [11, p. 61], it is faster than other minimization methods, such as SIRQIT [9]. The cost of SRQMCG2 is

$$C_{SRQMCG2} \sim n_{CG}(\tau_2)(7Mm_0N + m_0NM^2/2 + 10M^3), \quad (20)$$

where $n_{CG} \sim O(N^\alpha)$ is the number of CG iterations. Such an assumption is suggested by analyzing Figs. 1–4 in [11], where $M = 15 = \text{const}$. Lower estimation of the asymptotic cost is then

$$C_{SRQMCG2} \sim O(m_0N^{1+\alpha}M) + O(m_0N^{1+\alpha}M^2) + O(N^\alpha M^3). \quad (21)$$

We see that the cost of SRQMCG2 is comparable to that of our method only if N is small. For large N and M the cost of orthogonalization is much larger in SRQMCG2 than in our method, since the orthogonalization has to be performed in each iteration of SRQMCG2. Moreover, the number of matrix–vector multiplications per step is large (4–7) compared with other methods (1–2). This is the common feature of all the methods of Rayleigh quotient minimization type [9, 11].

For elliptic problems with $M = 15$, $\tau_2 = 10^{-6}$ we have obtained the following results (expressed by the number of matrix–vector multiplications):

Dimension N	100	800	1800	2300	
SRQMCG2	13,000	18,000	20,000	28,000	(±10%)
Our method	2,000	4,500	6,000	7,000	

The costs of SRQMCG2 have been calculated based on [11, algorithm and Figs. 1–4]. It seems that our method is better even for small N and M and all the more so for large ones.

(5) The TRACMN seems to be the best minimization-type method [6]. Its rate of convergence is similar to that of SI [6, p. 1248]; then the cost is

$$C_{TRACMN} \sim n_{out}(\tau_2)(n_{CG}(\tau_1)\, Mm_0N + m_0NM^2/2 + 10M^3), \quad (22)$$

where $n_{out} \sim O(1)$ is the number of outer iterations and $n_{CG} \sim O(N^\alpha)$ is the number of inner (conjugate gradient) iterations. Thus the asymptotic cost is

$$C_{TRACMN} \sim O(m_0N^{1+\alpha}M) + O(m_0NM^2) + O(M^3). \quad (23)$$

The costs of TRACMN and our method are then comparable if $n_{out} \sim O(1)$; otherwise ($n_{out} \sim O(M)$) our method is better. To check it, we have performed the tests described in [6]. Test 8 is the Laplace problem of medium size $N = 992$, $M = 2$, $\tau_2 = 10^{-12}$. We present two results, for $N = 992$ with nonequal grid sizes ($h_x = \frac{1}{32}$, $h_y = \frac{1}{33}$) and for $N = 1024$ with equal ones ($h_x = h_y = \frac{1}{33}$). The numbers of the matrix–vector multiplications were

	Dimension N	Cost	
TRACMN,	992	1345	[6, p. 1225, Table 2]
our method	992	707	
our method	1024	450	(no results for TRACMN).

It is seen that our method is about two times faster in this case.

The next two tests are the small biharmonic problems. The results are:

	TRACMN	Our method
Test 6, $N = 64$, $M = 5$, simply supported plate	1462	1810
Test 7, $N = 64$, $M = 10$, clamped plate	2434	3772

It is seen, that TRACMN behaves better in both cases. This is not especially surprising if we note that N is very small; thus we could not expect our method to be better. The comparison for large problems would be interesting.

(6) The TLIME method of Szyld [18] is especially suited to the searching of eigenpairs within the prescribed interval $(\gamma - \alpha, \gamma + \alpha)$. A suitable strategy of switching between inverse and Rayleigh quotient iterations assures convergence to the required eigenvector without jumps to any other. Omitting the cost of initial approximation, we obtain the cost of computing of an individual eigenpair

$$C_{TLIME}(M = 1) \sim n_{it}(\tau_2, N^\alpha)(m_0N) \sim O(m_0N^{1+\alpha}), \quad (24)$$

where n_{it} is the number of iterations. The asymptotic cost

of TLIME is essentially the same as that of the second phase of our method due to the same idea—using the Rayleigh quotient method with preconditioned MINRES or SYMMLQ as a linear solver.To compare our method with TLIME we have solved the problems described in [22] (standard Laplace on a unit square and the Laplace with variable coefficients [22, p. 171]) with $\tau_2 = 10^{-10}$ and SSOR preconditioning, obtaining the results given in Table III (expressed by the number of iterations, as in [22]).

We see that the total number of iterations is less in our method (except in the case of λ_{17}, where the TLIME is somewhat better). Moreover, the number of RQ iterations is almost half of those in TLIME, despite a similarity of TLIME and the second phase of our algorithm. The reason is that in our method RQ iteration is performed when the initial approximations are sufficiently converged. It confirms the observation that the convergence of TLIME depends strongly on the quality of the initial approximations [18, p. 1372].

In our opinion, the lack of a general strategy for the selection of the initial approximation is an important weakness of the TLIME. This may be a serious drawback when a large number of eigenpairs have to be computed. We suppose, however, that TLIME may be better than our method when one is searching for the eigenpair within an interval, whereas our method is better for other problems.

Conclusions Based on Comparisons. (1) We may note that three factors are crucial for the effectiveness of all the methods described above: factorization, the frequency of orthogonalizations, and the influence of dimension N on the cost. Other factors, being specific to concrete methods, are hard to evaluate theoretically.

(2) Methods performing factorization, such as the Lanczos method (including its block version) or accelerated subspace iteration, are efficient for small and medium problems. However, for large dimensions, $N \sim 10^4$, they are expensive, especially the ASI. Thus, for large N and small or medium M our method seems to be better. For large M our method is better than ASI; a similar conclusion for the Lanczos method cannot be drawn without additional testing. In Fig. 10 we present the range of dimensions for which our method seems to be superior to all other methods using factorization.

(3) Methods performing orthogonalization on each iteration, such as the Davidson or SRQMCG2, are inefficient for large M. Thus, our method seems to be better in this case. For small M the Davidson method may be better, however.

(4) The costs of TRACMN and TLIME are similar to that of our method. The comparisons show a little superiority for our method, but the final conclusions should be based on more general testing.

TABLE III

		$N=961$				$N=3969$			
		Laplace		Var. coef.		Laplace		Var. coef.	
		λ_4	λ_{17}	λ_2	λ_8	λ_4	λ_{17}	λ_2	λ_8
Our method	No. of RQ iter.	90	172	80	133	188	253	114	260
	Total no. of iter.	120	282	112	184	250	382	152	310
TLIME	[22, Tables 1–4]	219	340	177	334	302	378	238	—

8. GENERALIZATION TO THE WEAKLY NON-SYMMETRIC PROBLEMS

Our method can be used for solving the weakly non-symmetric problems, having real eigenvalues only. An important class of such problems is the Laplace problem (7) supplied by first-order derivative terms,

$$-\nabla^2 u + \gamma(\partial u/\partial x + \partial u/\partial y + \partial u/\partial z) - \lambda u = 0, \qquad (25)$$

where γ can be thought of as a measure of the non-symmetry; if γ goes to zero, the problem goes to the symmetric one. We assume, that γ is so small, that the spectrum is pure real. After discretization of (25) by FDM we obtain the non-symmetric matrix problem. The algorithm from Section 2 has been modified to handle unsymmetric matrices. The first test was that 2D one with $N = 225$, $\gamma = 1.0$, described in [40], where the first eigenpair was found by the Arnoldi method after 90 matrix–vector multiplications. Our method had to perform 223 such multiplications; thus we see that the Arnoldi method is better. This is not surprising if we recall that our method is not recommended for small problems. On the other hand, we were able to find several consecutive eigenpairs with moderate growth of cost. In the second test we checked the influence of non-symmetry on the behavior of our method. We assumed $N = 960$, $M = 20$. The costs for $\gamma = 0.4$, 1.0, and 2.0 are presented in Figs. 12A, B, C. For $\gamma < 1.0$ the results are acceptable, whereas for $\gamma = 2.0$ the convergence was slow and irregular, so we do not recommend our method if γ is large. In the last tests we checked the behavior of our method for large dimensions. The 3D problems with $N = 4096$, $M = 45$; $N = 8000$, $M = 25$; $N = 27{,}000$, $M = 12$ have been solved for $\gamma = 0.5$ (cf. Fig. 12D, E). No differences in comparisons with medium problems have been noted.

We appreciate that the above results are surprisingly good, because we did not expect such behavior at all. An important advantage of our method is that the cost of computation of consecutive eigenpairs grows moderately,

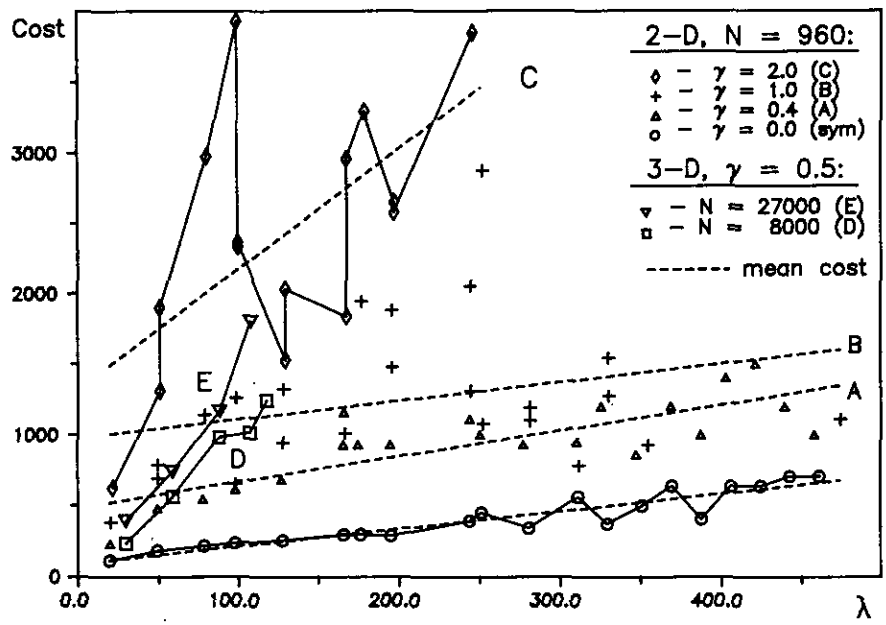

FIG. 12. The results of computations for weakly non-symmetric problems. The cost is expressed by the number of matrix–vector multiplications.

similar to the symmetric case; thus we can find several of them very effectively. We think that our method can be recommended for problems with "small" non-symmetry, $\gamma < 1.0$.

9. WEAK POINTS AND FAILURES OF OUR METHOD

The results presented above show, in our opinion, the good performance of our method. However, it has also disadvantages. The main weak point of our method is that there are parameters which have to be chosen properly. Although most of them can be defined once and for all for a wide range of eigenproblems, there are situations when some of the parameters should be "tuned." This means, that some trials may be required to obtain satisfactory results. Thus, the method is not recommended for sporadic users, but rather for those which are especially interested in solving very large eigenproblems routinely—the extensive users [37]. However, there is the possibility of preparing a guide which might help to choose the parameters properly.

In almost all cases the proposed method behaved well. However, some interesting failures were observed during testing. Here are a few:

(1) The bad initial approximations to the higher eigenvectors and the orthogonalization instabilities. When we attempted to compute the whole spectrum of the Laplace problem (7) with $N = M = 1000$, it happened that, after finding ~ 760 eigenpairs, copies of the lower ones (the first to tenth) appeared again and again. The orthogonalization could not reject them (although double orthogonalization was used) and the process became looped.

(2) The inadequate shifting strategy. For the large biharmonic problem, $N = 10{,}000$, after computing the 50th eigenpair there was a jump to the 80th one. Almost 30 of them were missed. The next 20 were computed properly, however. The shifting strategy was too conservative.

(3) The troubles with the conditioning. When we tried to solve the biharmonic problems without preconditioning, the cost of the first eigenpair was exceptionally high, whereas the next ones were computing with less, but still large, cost. The large problems could not be solved at all. Using the SSOR preconditioning caused a drastic reduction of costs and enabled us to solve very large eigenproblems.

(4) The troubles with the start. When we tried to solve the structural eigenproblems having rigid modes of vibrations (those with $\lambda = 0$), the iterations did not converge, although the same problem without rigid modes converged very quickly. Improving the shifting strategy preserved such situations.

(5) Low precision of arithmetic. The method was not able to compute the third eigenpair for the huge Laplace problem (7), $N = 216{,}000$ (although the first two had been computed). The Rayleigh quotient iterations were irregular and did not converge after a large number of steps. We suppose that the precision of arithmetic (~ 7 digits) was too low.

10. COMPUTATION OF THE RIGHTMOST EIGENPAIRS

Recently we have used our method with success for the computation of several rightmost eigenpairs. The behavior of our method was essentially the same as in the case for the

computation of leftmost eigenpairs. For the 3D Laplace problem (7) we have solved the following rightmost eigenproblems: $N = 1000$, $M = 300$; $N = 5832$, $M = 50$; $N = 9831$, $M = 30$; $N = 27{,}000$, $M = 16$.

It is important, that the costs of computations are almost identical to those of the leftmost eigenpairs for the same problems. This agrees with the theoretical prediction.

11. CONCLUSIONS

1. The proposed two-phase subspace iteration/Rayleigh quotient method enables one to compute effectively a large number of eigenpairs of the generalized symmetric eigenproblem $Ax = \lambda Bx$.

2. The method is best suited to very large, 3D elliptic problems, discretized by FEM or FDM.

3. For large, 3D biharmonic problems our method can be also recommended.

4. Our method is not recommended for small problems whose matrices have the bandwith m_b comparable to the number of non-zero elements per row, $m_0 \sim m_b$.

5. The method can be used for non-symmetric problems if the non-symmetry is moderate. Several leftmost eigenpairs can be computed effectively.

6. Our method seems to be the best for problems where the factorization is not allowed because of memory limitations.

7. For extremely large 3D problems ($N \sim 10^5$) our method is much better than all methods performing factorizations, like the Lanczos or the accelerated subspace iteration methods. For large M it is also better than methods performing orthogonalization in each iteration, such as the Davidson or SRQMCG2 methods.

8. Our method can be used for solving full generalized eigenproblems that cannot be solved by the transformation methods because of memory limitations.

9. The application of conjugate gradient/Lanczos-type iterative methods as linear solvers makes the method "factorization-free."

10. The use of the SSOR preconditioner improves the effectiveness for large problems, whereas using it for small problems is not always successful.

11. The missing eigenpairs are unlikely if we use a proper strategy of shifting for the subspace iteration phase.

12. The proposed method is able to compute multiple eigenvalues without any difficulties.

13. The cost of computation depends mainly on the dimensions of the problem (roughly as $N^{1.2}$–$N^{1.8}$) and on the distribution of the eigenvalues in the spectrum.

14. The values of the control parameters are not critical and the standard ones can be used for various problems. Sometimes (rarely) they should be "tuned."

15. The main part of the algorithm can be built up as a "black box."

16. The method can be vectorized easily due to the independence of the iterative solution processes performed in both phases of the computation.

17. Several rightmost eigenpairs can be computed also by our method.

The algorithm presented in this paper is being developed. We are searching for better preconditioners, more effective switching criteria, and a shifting strategy. Moreover, we have developed another method based on a similar idea.

ACKNOWLEDGMENTS

The authors are indebted to Professor R. Gutowski of the Institute of Aeronautics and Applied Mechanics, Warsaw University of Technology for his help and stimulating discussions. The authors also thank the referee for comments that led to improvements in the section on comparisons. This work was supported in part by the KBN (State Committee for Scientific Research Republic of Poland) Grant No. 309559101.

REFERENCES

1. H. G. Matthies, *Comput. Struct.* **21**, 319 (1985).
2. B. N. Parlett, *The Symmetric Eigenvalue Problems* (Prentice-Hall, Englewood Cliffs, NJ, 1980).
3. T. Ericsson and A. Ruhe, *Math. Comput.* **35**, 1251 (1980).
4. B. N. Parlett and D. S. Scott, *Math. Comput.* **33**, 217 (1979).
5. B. Nour-Omid, B. N. Parlett, and R. L. Taylor, *Int. J. Numer. Methods Eng.* **19**, 859 (1983).
6. A. H. Sameh and J. A. Wisniewski, *SIAM J. Numer. Anal.* **19**, 1243 (1982).
7. K. J. Bathe and L. E. Wilson, *Int. J. Numer. Methods Eng.* **6**, 213 (1973).
8. K. J. Bathe and S. Ramaswamy, *Comput. Methods Appl. Mech. Eng.* **23**, 313 (1980).
9. D. E. Longsine and S. F. McCormick, *Linear Algebra Appl.* **34**, 195 (1980).
10. G. Gambolatti, G. Pini, and F. Sartoretto, *J. Comput. Phys.* **74**, 41 (1988).
11. F. Sartoretto, G. Pini, and G. Gambolatti, *J. Comput. Phys.* **81**, 53 (1989).
12. E. R. Davidson, *J. Comput. Phys.* **17**, 87 (1975).
13. N. Kosugi, *J. Comput. Phys.* **55**, 426 (1984).
14. R. B. Morgan and D. S. Scott, *SIAM J. Sci. Statist. Comput.* **7**, 817 (1986).
15. T. Z. Kalamboukis, *J. Phys. A: Math. Gen.* **13**, 57 (1980).
16. J. H. Wilkinson and C. Reinsch, *Handbook for Automatic Computation* (Springer-Verlag, New York, 1971).
17. D. S. Scott, *SIAM J. Sci. Statist. Comput.* **5**, 658 (1984).

18. D. B. Szyld, *SIAM J. Numer. Anal.* **25**, 1369 (1988).
19. C. Beattie and D. W. Fox, *SIAM J. Matrix Anal. Appl.* **10**, 80 (1989).
20. C. C. Paige and M. A. Saunders, *SIAM J. Numer. Anal.* **12**, 617 (1975).
21. A. Ruhe and T. Wiberg, *BIT* **12**, 543 (1972).
22. D. B. Szyld and O. Widlund, in *Advances in Computer Methods for Partial Differential Equations III*, Proceedings, IMACS, 1979, edited by S. Vichnevetsky and R. S. Stepelman (Rutgers University, New Brunswick), p. 167.
23. A. Jennings and M. A. Ajiz, *SIAM J. Sci. Statist. Comput.* **5**, 978 (1984).
24. D. J. Evans, *Computing* **32**, 139 (1984).
25. D. G. Luenberger, *SIAM J. Appl. Math.* **17**, 1263 (1969).
26. U. I. Ojalvo, *Comput. Struct.* **20**, 115 (1985).
27. D. J. Evans and J. Shanehchi, *Comput. Methods Appl. Mech. Eng.* **31**, 251 (1982).
28. J. A. Meijerink and H. A. van der Vorst, *Math. Comput.* **31**, 148 (1977).
29. D. S. Kershaw, *J. Comput. Phys.* **26**, 43 (1978).
30. O. Axelsson, *BIT* **25**, 166 (1985).
31. D. J. Nefske, J. A. Wolf, and L. J. Howell, *J. Sound Vibr.* **80**, 247 (1982).
32. R. W. Clough and E. L. Wilson, *Comput. Methods Appl. Mech. Eng.* **17/18**, 107 (1979).
33. O. Axelsson and V. A. Barker, *Finite Element Solutions of Boundary Value Problems. Theory and Applications* (Academic Press, New York, 1988).
34. B. Nour-Omid, B. N. Parlett, T. Ericsson, and P. S. Jensen, *Math. Comput.* **48**, 663 (1987).
35. D. S. Scott, *Comput. Phys. Commun.* **53**, 271 (1989).
36. B. N. Parlett and B. Nour-Omid, *Comput. Phys. Commun.* **53**, 169 (1989).
37. B. N. Parlett, *SIAM J. Sci. Statist. Comput.* **5**, 590 (1984).
38. T. Z. Kalamboukis, *J. Comput. Phys.* **53**, 82 (1984).
39. W. Kerner, *J. Comput. Phys.* **85**, 1 (1989).
40. Y. Saad, *Linear Algebra Appl.* **34**, 269 (1980).