Ann. nucl. Energy, Vol. 18, No. 4, pp. 205–227, 1991
Printed in Great Britain

0306-4549/91 $3.00+0.00
Pergamon Press plc

CONJUGATE GRADIENT LIKE METHODS AND THEIR APPLICATION TO EIGENVALUE PROBLEMS FOR NEUTRON DIFFUSION EQUATION

E. SUETOMI† and H. SEKIMOTO

Research Laboratory for Nuclear Reactors,

Tokyo Institute of Technology,

O-okayama, Meguro-ku, Tokyo 152, Japan

(*Received 20 November 1990*)

Abstract — This paper presents conjugate gradient (CG) like methods for solving generalized eigenvalue problems for neutron diffusion. The CG method to minimize the Rayleigh quotient was proposed to one-group eigenvalue problems. In order to accelerate the convergence, we employed preconditioners based on incomplete factorization of the coefficient matrix. In case of few-group eigenvalue problems, we introduce a residual norm and minimize it by ORTHOMIN(1) (OR) method. We also proposed preconditioned OR methods. In order to vectorize the preconditioned CG like methods based on the incomplete factorization, the hyperplane method was used and implemented on vector computers.

Numerical experiments showed that the preconditioned CG like methods required much smaller number of iterations and computational time compared with the conventional inner-outer iterative scheme for both one-group and few-group eigenvalue problems.

Present address: † Computer Software Development Co., Ltd.,
Shibakoen, Minato-ku, Tokyo, 105, Japan.

I. INTRODUCTION

The application of the finite difference or finite element method to an eigenvalue problem for neutron diffusion equation leads to a generalized matrix eigenvalue problem. This problem is usually solved with the power method, which consists of inner iteration (solving systems of linear equations) and outer iteration (calculating of source and eigenvalue). Many techniques for solving linear equations in the inner iteration and accelerating the outer iteration have been developed. These methods require an acceleration parameter, whose exactly optimum value is difficult to be obtained.

On the other hand, it is well known that the minimum value of the Rayleigh quotient corresponds to the minimum eigenvalue for symmetric generalized eigenvalue problems and minimization technique by using a gradient like method does not need inner-outer iterations and any acceleration parameter. One of the oldest and best known method for seeking the minimum is the steepest descent (SD) method, first proposed by Hestenes and Karush (1951). The SD method converges very slowly if the matrix size is not small. By using the Fletcher-Reeves method (1964), Bradbury and Fletcher (1966) developed the conjugate gradient (CG) method to minimize the Rayleigh quotient. Geradin (1971) proposed an efficient way to determine the minimum by the CG method and Suetomi and Sekimoto (1988) applied it to one-group neutron diffusion equations.

The coefficient matrices of multi-group diffusion equations, however, become nonsymmetric, and any efficient CG like methods to solve the generalized eigenvalue problems have not been developed. Recently, the ORTHOMIN(1) method has been applied to solve nonsymmetric systems of linear equations of which the symmetric part of the coefficient matrix was positive definite (Vinsome, 1976; Eisenstat *et al.*, 1983). In the present paper, a residual norm is introduced for the eigenvalue problem for few-group diffusion equations and minimized with the ORTHOMIN(1) method to solve the problem.

In order to enhance the convergence rate of the CG like methods, we employed preconditioners based on the incomplete factorization of the coefficient matrices.

The CG like methods can be vectorized easily since they consist of vector operations and matrix multiplications. Hence, they are ones of the most suitable algorithm on vector computers. On the other hand, preconditioned CG (PCG) like methods based on the incomplete factorization require the forward and backward substitution algorithm, so that they can not be directly vectorized. In the present study we employ the hyperplane method, which was proposed by Ushiro (1984) for the vectorization of the PCG like methods.

In Chap. 2 the CG and PCG methods are described for their application to one-group eigenvalue problems. In Chap. 3 the ORTHOMIN(1) method is proposed to determine the effective multiplication factor of few-group eigenvalue problems and the preconditioners based on the incomplete factorization are discussed. The numerical experiments for one- and few-group eigenvalue problems are performed in Chap. 4. The PCG like methods with the hyperplane vectorization are compared on the vector computers: the HITAC S-820/80 and ETA10-E.

II. ONE-GROUP EIGENVALUE PROBLEM

II. 1. Rayleigh quotient minimization by conjugate gradient method

The time-independent one-group neutron diffusion equation in multiplying media can be written as

$$-\nabla\cdot D(x)\nabla\phi(x) + \Sigma_a(x)\phi(x) = \frac{1}{k}\nu\Sigma_f(x)\phi(x) \ , \tag{1}$$

where $\phi(x)$, $D(x)$, $\Sigma_a(x)$, $\nu\Sigma_f(x)$ and k denote the neutron flux at point x in space, diffusion coefficient, absorption cross section, average number of neutrons released per fission times fission cross section and effective multiplication factor, respectively.

Finite difference or finite element discretization of (1) in space x leads to a generalized eigenvalue problem of the form

$$A\phi = \lambda \mathrm{B}\phi \ , \tag{2}$$

where ϕ is the scalar neutron flux vector and λ the eigenvalue which is the inverse of the effective multiplication factor. The matrix A is an $N\times N$, real, symmetric and positive definite matrix, composed of the discretized Laplacian operator and the absorption term on the left side of (1). The matrix B is a diagonal matrix representing the fission source term on the right side of (1). The order of matrix for two- or three-dimensional geometry becomes large and iterative methods are frequently adopted. In the present work we propose the conjugate gradient (CG) method and preconditioned conjugate gradient (PCG) method to solve (2).

When the matrices A and B are symmetric and one of them is positive definite, the smallest value of the Rayleigh quotient

$$R(\phi) = \frac{(\phi, A\phi)}{(\phi, B\phi)} \tag{3}$$

gives the minimum eigenvalue of (2), and the corresponding eigenvector ϕ gives the neutron flux.

Let us consider the problem of finding the unconstrained minimum of a continuous function $F(\phi)$ of N variables. In the neighborhood of the required minimum, ϕ^*, the function can be expanded in the form

$$F(\phi) = F(\phi^*) + \frac{1}{2}(\phi - \phi^*)^T H(\phi - \phi^*) + \text{higher terms} \ , \tag{4}$$

where H denotes the second-order derivatives of $F(\phi)$ (Hessian matrix) and we assume that H is a real, symmetric and positive definite matrix. There is a large number of iterative methods for minimizing the above function $F(\phi)$. Most of these methods have the general form

$$\phi_{i+1} = \phi_i + a_i s_i \ , \tag{5}$$

where the vector s_i is a direction vector at the i-th iteration and the scalar a_i locally minimizes $F(\phi)$ along s_i. One of the oldest and best known method for seeking the minimum is the steepest descent (SD) method, first proposed by Hestenes and Karush (1951). In this method, the direction vector is given by the gradient vector of (4). Unfortunately, it does not generally converge to the required minimum in a finite procedure and shows slow convergence unless the size of the matrix is small. A quadratically convergent gradient method was proposed by Fletcher and Reeves (1964), which is based on the CG method.

Geradin (1971) proposed an efficient way to find the minimum of (3) by the CG method, and Suetomi and Sekimoto (1988) applied it to one-group neutron diffusion equations. This method does not need the inner-outer iterations, and shows the quadratic convergence in the neighborhood of the eigensolution. The algorithm of the CG method to minimize the Rayleigh quotient can be derived from the following scheme.

The parameter a_i in (5) is chosen to minimize

$$R(\phi_{i+1}) = R(\phi_i + a_i s_i) = \frac{(s_i, As_i)a_i^2 + 2(\phi_i, As_i)a_i + (\phi_i, A\phi_i)}{(s_i, Bs_i)a_i^2 + 2(\phi_i, Bs_i)a_i + (\phi_i, B\phi_i)} . \quad (6)$$

Therefore

$$\frac{\partial R(\phi_i + a_i s_i)}{\partial a_i} = \frac{2(aa_i^2 + ba_i + c)}{\{(s_i, Bs_i)a_i^2 + 2(\phi_i, Bs_i)a_i + (\phi_i, B\phi_i)\}^2} = 0 , \quad (7)$$

where

$$a = (s_i, As_i)(\phi_i, Bs_i) - (\phi_i, As_i)(s_i, Bs_i) , \quad (8a)$$

$$b = (s_i, As_i)(\phi_i, B\phi_i) - (\phi_i, A\phi_i)(s_i, Bs_i) , \quad (8b)$$

$$c = (\phi_i, As_i)(\phi_i, B\phi_i) - (\phi_i, A\phi_i)(\phi_i, Bs_i) , \quad (8c)$$

From (7) we get the following two solutions.

$$a_i^+ = \frac{-b + \sqrt{b^2 - 4ac}}{2a} , \quad (9a)$$

$$a_i^- = \frac{-b - \sqrt{b^2 - 4ac}}{2a} . \quad (9b)$$

The choice of a_i^+ always yields the minimum of (6).

The new direction vector s_{i+1} is given by

$$s_{i+1} = -g_{i+1} + \beta_i s_i , \quad (10)$$

where g_{i+1} denotes the gradient vector of (3) at the point ϕ_{i+1} and it can be obtained by

$$g_{i+1} = \frac{\partial R(\phi_{i+1})}{\partial \phi_{i+1}} = \frac{2(A\phi_{i+1} - \lambda_{i+1}B\phi_{i+1})}{(\phi_{i+1}, B\phi_{i+1})} \ . \tag{11}$$

The direction vector is generated such that two successive direction vectors are conjugate with respect to the local Hessian matrix $H(\phi_{i+1})$

$$(s_{i+1}, H(\phi_{i+1})s_i) = 0 \ . \tag{12}$$

The local Hessian matrix is given by

$$H(\phi_{i+1}) = \frac{2}{(\phi_{i+1}, B\phi_{i+1})} [A - \lambda_{i+1}B - B\phi_{i+1}g_{i+1}^T - g_{i+1}\phi_{i+1}^T B] \, . \tag{13}$$

Substituting (10) and (13) into (12), the parameter β_i is obtained as

$$\beta_i = [g_{i+1}^T(A - \lambda_{i+1}B)s_i - g_{i+1}^T g_{i+1}(\phi_{i+1}, Bs_i)] \,/\, s_i^T(A - \lambda_{i+1}B)s_i \ . \tag{14}$$

Therefore the CG method for one-group eigenvalue problem can be given by the following algorithm:

ALGORITHM 1 (CG method for one-group eigenvalue problem)

$$Choose\ \phi_0 \ . \tag{15a}$$

$$\lambda_0 = \frac{(\phi_0, A\phi_0)}{(\phi_0, B\phi_0)} \ , \tag{15b}$$

$$g_0 = \frac{2(A\phi_0 - \lambda_0 B\phi_0)}{(\phi_0, B\phi_0)} \ , \tag{15c}$$

$$s_0 = -g_0 \ . \tag{15d}$$

For $i=0$ Step 1 Until Convergence Do

$$a = (s_i, As_i)(\phi_i, Bs_i) - (\phi_i, As_i)(s_i, Bs_i) \ , \tag{15e}$$

$$b = (s_i, As_i)(\phi_i, B\phi_i) - (\phi_i, A\phi_i)(s_i, Bs_i) \ , \tag{15f}$$

$$c = (\phi_i, As_i)(\phi_i, B\phi_i) - (\phi_i, \mathrm{A}\phi_i)(\phi_i, Bs_i) \ , \tag{15g}$$

$$a_i = \frac{-b + \sqrt{b^2 - 4ac}}{2a} , \tag{15h}$$

$$\phi_{i+1} = \phi_i + a_i s_i , \tag{15i}$$

$$\lambda_{i+1} = \frac{(\phi_{i+1}, A\phi_{i+1})}{(\phi_{i+1}, B\phi_{i+1})} , \tag{15j}$$

$$g_{i+1} = \frac{2(A\phi_{i+1} - \lambda_{i+1}B\phi_{i+1})}{(\phi_{i+1}, B\phi_{i+1})} , \tag{15k}$$

$$\beta_i = [g_{i+1}^T(A - \lambda_{i+1}B)s_i - g_{i+1}^T g_{i+1}(\phi_{i+1}, Bs_i)] \,/\, s_i^T(A - \lambda_{i+1}B)s_i , \tag{15l}$$

$$s_{i+1} = -g_{i+1} + \beta_i s_i , \tag{15m}$$

The iteration is terminated when the relative residual norm $\| \lambda_i B\phi_i - A\phi_i \|_2 / \| \lambda_i B\phi_i \|_2$ satisfies the following inequality

$$\| \lambda_i B\phi_i - A\phi_i \|_2 / \| \lambda_i B\phi_i \|_2 < \varepsilon , \tag{16}$$

where $\| \cdot \|_2$ is the Euclidean norm and ε is an input parameter.

II. 2. Preconditioned conjugate gradient method

The convergence rate of the algorithm for minimization of the Rayleigh quotient depends on the condition number of the Hessian matrix $\kappa(H)$ defined by the ratio of the largest to smallest eigenvalue (Gambolati *et al.*, 1988).

Let us denote by λ^* and ϕ^* the minimum value of the Rayleigh quotient (namely the minimum eigenvalue) and the corresponding eigenvector, respectively. The Hessian matrix of the Rayleigh quotient at a point ϕ^* is given by By denoting the smallest eigenvalue (different from zero) and the

$$H(\phi^*) = \frac{2(A - \lambda^* B)}{(\phi^*, B\phi^*)} . \tag{17}$$

largest eigenvalue of an arbitrary symmetric matrix K by $\Lambda_{\min}(K)$ and $\Lambda_{\max}(K)$, respectively, the condition number can be written as

$$\kappa(H(\phi^*)) = \frac{\Lambda_{\max}(A - \lambda^* B)}{\Lambda_{\min}(A - \lambda^* B)} . \tag{18}$$

Using a symmetric and positive definite matrix K, we transform the general eigenvalue problem (2) to

$$K^{-1/2}AK^{-1/2}\Psi = \lambda K^{-1/2}BK^{-1/2}\Psi \ , \tag{19}$$

where $\Psi = K^{1/2}\phi$. In this case, the Rayleigh quotient can be written in the form

$$R(\Psi) = \frac{(\Psi, K^{-1/2}AK^{-1/2}\Psi)}{(\Psi, K^{-1/2}BK^{-1/2}\Psi)} \ . \tag{20}$$

The Hessian matrix of (20) at a point Ψ^* ($= K^{1/2}\phi^*$) is

$$H(\Psi^*) = \frac{2(K^{-1/2}AK^{-1/2} - \lambda^* K^{-1/2}BK^{-1/2})}{(\Psi^*, K^{-1/2}BK^{-1/2}\Psi^*)} \ . \tag{21}$$

Since

$$\begin{aligned} K^{-1/2}AK^{-1/2} - \lambda^* K^{-1/2}BK^{-1/2} &= K^{-1/2}(A - \lambda^* B)K^{-1/2} \\ &= K^{-1/2}(A - \lambda^* B)K^{-1}K^{1/2} \ , \end{aligned} \tag{22}$$

$K^{-1/2}AK^{-1/2} - \lambda^* K^{-1/2}BK^{-1/2}$ is similar to $(A - \lambda^* B)K^{-1}$, then both matrices have the same eigenvalues.

When K^{-1}, is equal to A^{-1}, the condition number for the transformed generalized eigenvalue problem (19) is given by

$$\kappa(H(\Psi^*)) = \frac{\Lambda_{\max}(A - \lambda^* B)}{\Lambda_{\min}(A - \lambda^* B)} \cdot \frac{\Lambda_{\min}(A)}{\Lambda_{\max}(A)} \ . \tag{23}$$

For a large scale matrix arising from the discretization of (1),

$$\Lambda_{\min}(A) \ll \Lambda_{\max}(A) \ , \tag{24}$$

and hence (23) is much smaller than (18). Therefore, the CG method for minimization of the transformed or the preconditioned Rayleigh quotient (20) will converge much faster than that for the original quotient (3).

Since the calculation of A^{-1} is expensive and requires a large memory storage, we propose an approximation matrix to A^{-1} based on the incomplete Choleski (IC) factorization (Meijerink *et al.*, 1977). An approved method for solving $A\phi = b$ (A is an $N \times N$, real, symmetric and positive definite matrix) is obtained using the Choleski factorization

$$A = LDL^T \ , \tag{25}$$

where L is a lower triangular matrix with unit diagonal elements. For a large sparse matrix A, the entries of L corresponding to the zero elements of A are usually small. Hence, putting these entries equal to zero may yield a reasonable approximation of A;

$$A = \tilde{L}\tilde{D}\tilde{L}^T + E \ , \tag{26}$$

where $\| E \|$ is hopefully small compared with $\| A \|$. In the incomplete factorization, the required replacement of the element of $\tilde{L}$ with zero value is performed during the factorization process. We adopt the same sparsity pattern as the lower triangular part of A. We employ $K^{-1} = (\tilde{L}\tilde{D}\tilde{L}^T)^{-1}$ instead of A^{-1}.

The IC factorization can be modified slightly by lumping the neglected values of L to the diagonal elements. This is so-called the modified incomplete Choleski (MIC) factorization, which has a strong effect on the eigenvalue distribution of the preconditioned matrix (Gustafsson, 1978).

On the other hand, the scaling of row and columns of the matrix A is widely used as a simple preconditioning. In this preconditioning the preconditioner K is selected as follows:

$$K^{-1/2} = diag(1/\sqrt{a_{11}}, \cdots, 1/\sqrt{a_{NN}}) \ , \tag{27}$$

where a_{ii} is the diagonal element of A.

One can write the preconditioner by a positive symmetric matrix K in the form

$$K = \tilde{L}\tilde{D}\tilde{L}^T \ . \tag{28}$$

Then one can transform (2) in the form

$$\tilde{A}\tilde{\phi} = \lambda\tilde{B}\tilde{\phi} \ , \tag{29}$$

where

$$\tilde{\phi} = (\tilde{L}\tilde{D}^{1/2})^T\phi \ , \tag{30}$$

$$\tilde{A} = (\tilde{L}\tilde{D}^{1/2})^{-1}A(\tilde{L}\tilde{D}^{1/2})^{-T} \ , \tag{31}$$

$$\tilde{B} = (\tilde{L}\tilde{D}^{1/2})^{-1}B(\tilde{L}\tilde{D}^{1/2})^{-T} \ . \tag{32}$$

The vectors $\tilde{\phi}_i = (\tilde{L}\tilde{D}^{1/2})^T\phi_i$ are the preconditioned conjugate gradient iterates expressed in terms of the original variable. In order to find the algorithm which is represented by ϕ_i in place of $\tilde{\phi}_i$, we set

$$\begin{aligned}\tilde{g}_0 &= \frac{2(\tilde{A}\tilde{\phi}_0 - \lambda_0\tilde{B}\tilde{\phi}_0)}{(\tilde{\phi}_0, \tilde{B}\tilde{\phi}_0)} = \frac{2(\tilde{L}\tilde{D}^{1/2})^{-1}(A\phi_0 - \lambda_0 B\phi_0)}{(\phi_0, B\phi_0)} \\ &= (\tilde{L}\tilde{D}^{1/2})^{-1}g_0 \ . \end{aligned} \tag{33}$$

We set also the vector $\bar{s}_0$ to satisfy $\bar{s}_0 = (\tilde{L}\tilde{D}^{1/2})^T s_0$ and $\bar{s}_0 = -\bar{g}_0$. After a little rearrangement, the following preconditioned conjugate gradient (PCG) method is obtained.

ALGORITHM 2 (PCG method for one-group eigenvalue problem)

$$\textit{Choose}\ \phi_0\ . \tag{34a}$$

$$\lambda_0 = \frac{(\phi_0, A\phi_0)}{(\phi_0, B\phi_0)}\ , \tag{34b}$$

$$g_0 = \frac{2(A\phi_0 - \lambda_0 B\phi_0)}{(\phi_0, B\phi_0)}\ , \tag{34c}$$

$$s_0 = -(\tilde{L}\tilde{D}\tilde{L}^T)^{-1} g_0\ . \tag{34d}$$

For $i=0$ Step 1 Until Convergence Do

$$a = (s_i, As_i)(\phi_i, Bs_i) - (\phi_i, As_i)(s_i, Bs_i)\ , \tag{34e}$$

$$b = (s_i, As_i)(\phi_i, B\phi_i) - (\phi_i, A\phi_i)(s_i, Bs_i)\ , \tag{34f}$$

$$c = (\phi_i, As_i)(\phi_i, B\phi_i) - (\phi_i, A\phi_i)(\phi_i, Bs_i)\ , \tag{34g}$$

$$a_i = \frac{-b + \sqrt{b^2 - 4ac}}{2a}\ , \tag{34h}$$

$$\phi_{i+1} = \phi_i + a_i s_i\ , \tag{34i}$$

$$\lambda_{i+1} = \frac{(\phi_{i+1}, A\phi_{i+1})}{(\phi_{i+1}, B\phi_{i+1})}\ , \tag{34j}$$

$$g_{i+1} = \frac{2(A\phi_{i+1} - \lambda_{i+1} B\phi_{i+1})}{(\phi_{i+1}, B\phi_{i+1})}\ , \tag{34k}$$

$$\begin{aligned}\beta_i = {} & [\{(\tilde{L}\tilde{D}\tilde{L}^T)^{-1} g_{i+1}\}^T (A - \lambda_{i+1} B) s_i \\ & - \{(\tilde{L}\tilde{D}\tilde{L}^T)^{-1} g_{i+1}\}^T g_{i+1} (\phi_{i+1}, Bs_i)] \,/\, s_i^T (A - \lambda_{i+1} B) s_i\ ,\end{aligned} \tag{34l}$$

$$s_{i+1} = -(\tilde{L}\tilde{D}\tilde{L}^T)^{-1}g_{i+1} + \beta_i s_i \ , \tag{34m}$$

In the present paper the algorithm employing the IC factorization is called as the ICCG (Incomplete Choleski & Conjugate Gradient) method and the algorithm employing the MIC factorization as the MICCG (Modified Incomplete Choleski & Conjugate Gradient) method. The method obtained by using (27) instead of the IC factorization is called as the SCG (Scaled Conjugate Gradient) method.

III. FEW-GROUP EIGENVALUE PROBLEM

III. 1. Residual norm minimization by ORTHOMIN(1) method

In this section, we discuss iterative methods for solving the following few-group diffusion equations:

$$\begin{aligned} -\nabla\cdot D_g(x)\nabla\phi_g(x) + \Sigma_{Rg}(x)\phi_g(x) - \sum_{g'=1}^{G}\Sigma_{sg'g}(x)\phi_{g'}(x) \\ = \frac{1}{k}\,\chi_g\sum_{g'=1}^{G}\nu_{g'}\Sigma_{fg'}(x)\phi_{g'}(x) \ , \\ g = 1,\ \dots,\ G \ , \end{aligned} \tag{35}$$

where $\phi_g(x)$, $D_g(x)$, $\Sigma_{Rg}(x)$, $\Sigma_{sg'g}(x)$, χ_g, $\nu_{g'}\Sigma_{fg'}(x)$ and k denote the neutron flux in energy group g, at point x in space, diffusion coefficient, removal cross section, scattering cross section from group g' to g, probability that a fission neutron will be born with energy in group g, average number of fission neutrons released per fission induced by a neutron with energy in group g' times fission cross section and effective multiplication factor. We can rewrite (35) as the matrix eigenvalue problem (2) by using the finite difference or finite element method. However, in this case, the matrices A and B become nonsymmetric, so we can not adopt the CG method.

For the nonsymmetric generalized eigenvalue problem, we define a residual vector as

$$r = \rho(\phi)B\phi - A\phi \ , \tag{36}$$

where $\rho(\phi)$ is given by

$$\rho(\phi) = \frac{(A\phi, B\phi)}{(B\phi, B\phi)} \ . \tag{37}$$

If ϕ converges to the eigenvector, $\rho(\phi)$ converges to the corresponding eigenvalue. We adopt the following residual norm

$$F(r) = \|\, r \,\|_2^2 = (r, r) \ , \tag{38}$$

as a functional. If we minimize (38) by a gradient method, then the following relation holds:

$$A\phi = \rho(\phi)B\phi \ . \tag{39}$$

Hence, if we choose a good approximation for the eigenvector corresponding to the neutron flux as a initial guess vector, then we can obtain the minimum eigenvalue (effective multiplication factor) and corresponding eigenvector (neutron flux).

Recently, the ORTHOMIN(1) method (the conjugate residual method) was applied to solve nonsymmetric systems of linear equations of which the symmetric parts of the coefficient matrices were positive definite [see APPENDIX (Vinsome, 1976; Eisenstat *et al.*, 1983)]. We consider the minimization of (38) by the ORTHOMIN(1) method.

The iterative scheme of this method is based on (5). The scalar a_i is chosen to minimize

$$F(a_i) = \| \rho(\phi_i)B(\phi_i + a_i s_i) - A(\phi_i + a_i s_i) \|_2^2 \ , \tag{40}$$

in the direction s_i, so that a_i satisfies

$$\frac{\partial F(a_i)}{\partial(a_i)} = 0 \ . \tag{41}$$

Then we have

$$a_i = \{(r_i, As_i) - \lambda_i(r_i, Bs_i)\} / \{(As_i, As_i) - 2\lambda_i(As_i, Bs_i) + \lambda_i^2(Bs_i, Bs_i)\} \ , \tag{42}$$

where $\lambda_i = \rho(\phi_i)$.

The new direction vector s_{i+1} is evaluated by

$$s_{i+1} = r_{i+1} + \beta_i s_i \ , \tag{43}$$

where the parameter β_i satisfies the following relation

$$((A - \lambda_{i+1}B)s_{i+1}, (A - \lambda_{i+1}B)s_i) = 0 \ . \tag{44}$$

By substituting (43) into (44), β_i is given by

$$\beta_i = -[(Ar_{i+1}, As_i) - \lambda_{i+1}\{(Ar_{i+1}, Bs_i) + (As_i, Br_{i+1})\} + \lambda_{i+1}^2(Br_{i+1}, Bs_i)] / [(As_i, As_i) - 2\lambda_{i+1}(As_i, Bs_i) + \lambda_{i+1}^2(Bs_i, Bs_i)] \ . \tag{45}$$

Therefore, we obtain the following ORTHOMIN(1) (OR) method for the few-group eigenvalue problem:

ALGORITHM 3 (OR method for few-group eigenvalue problem)

$$\textit{Choose } \phi_0 \ . \tag{46a}$$

$$\lambda_0 = \frac{(A\phi_0, B\phi_0)}{(B\phi_0, B\phi_0)} \ , \tag{46b}$$

$$r_0 = \lambda_0 B\phi_0 - A\phi_0 \ , \tag{46c}$$

$$s_0 = r_0 \ . \tag{46d}$$

For $i=0$ Step 1 Until Convergence Do

$$a_i = \{(r_i, As_i) - \lambda_i(r_i, Bs_i)\} / \{(As_i, As_i) - 2\lambda_i(As_i, Bs_i) + \lambda_i^2(Bs_i, Bs_i)\} , \tag{46e}$$

$$\phi_{i+1} = \phi_i + a_i s_i \ , \tag{46f}$$

$$\lambda_{i+1} = \frac{(A\phi_{i+1}, B\phi_{i+1})}{(B\phi_{i+1}, B\phi_{i+1})} \ ,$$

$$r_{i+1} = \lambda_{i+1} B\phi_{i+1} - A\phi_{i+1} \ , \tag{46h}$$

$$\begin{aligned}\beta_i = -[&(Ar_{i+1}, As_i) - \lambda_{i+1}\{(Ar_{i+1}, Bs_i) + (As_i, Br_{i+1})\} \\ &+ \lambda_{i+1}^2(Br_{i+1}, Bs_i)] / [(As_i, As_i) - 2\lambda_{i+1}(As_i, Bs_i) + \lambda_{i+1}^2(Bs_i, Bs_i)] \ .\end{aligned} \tag{46i}$$

$$s_{i+1} = r_{i+1} + \beta_i s_i \ . \tag{46j}$$

The convergence criterion for the iteration is given by the inequality (16).

III. 2. Preconditioned ORTHOMIN(1) method

The above mentioned algorithm is regarded as a solver for the nonlinear equations

$$(A\phi - \rho(\phi)B)\phi = 0 \ . \tag{47}$$

The OR method for solving a nonsymmetric system of linear equations converges to the solution, if the symmetric part of coefficient matrix A, $M = (A + A^T) / 2$, is positive definite. If $\{r_i\}$ is a sequence of residuals for the system of linear equations generated by the OR method and M is positive definite, then the residual norm is bounded as follows (Eisentat *et al.*, 1983) :

$$\| r_i \|_2 \leqq \left[1 - \frac{\Lambda_{\min}(M)^2}{\Lambda_{\max}(A^{\mathrm{T}}A)} \right]^{i/2} \| r_0 \|_2 \ . \tag{48}$$

The bound (48) shows that the convergence rate of the OR method for the system of linear equations depends on the eigenvalue distribution of A, and can be enhanced by preconditioning techniques.

In order to enhance the convergence rate of the OR method for (47), we precondition (39) with a nonsingular matrix K^{-1} applied on the right:

$$\mathrm{AK}-1(\mathrm{K}\phi) = \rho(\phi)\mathrm{BK}-1(\mathrm{K}\phi) \ . \tag{49a}$$

or applied on the left:

$$K^{-1}A\phi = \rho(\phi)K^{-1}B\phi \ . \tag{49b}$$

Since the numerical experiments have shown that the right preconditioning worked much better than the left preconditioning (Suetomi and Sekimoto, 1989), we employ the ORTHOMIN(1) method with the right preconditioning (PORMR method). The algorithm of the PORMR method for the few-group eigenvalue problem is shown below:

ALGORITHM 4 (PORMR method for few-group eigenvalue problem)

$$\textit{Choose}\ \phi_0 \ . \tag{50a}$$

$$\lambda_0 = \frac{(A\phi_0, B\phi_0)}{(B\phi_0, B\phi_0)} \ , \tag{50b}$$

$$r_0 = \lambda_0 B\phi_0 - A\phi_0 \ , \tag{50c}$$

$$s_0 = K^{-1}r_0 \ . \tag{50d}$$

For $i=0$ Step 1 Until Convergence Do

$$a_i = \{(r_i, As_i) - \lambda_i(r_i, Bs_i)\} / \{(As_i, As_i) - 2\lambda_i(As_i, Bs_i) + \lambda_i^2(Bs_i, Bs_i)\} \ , \tag{50e}$$

$$\phi_{i+1} = \phi_i + a_i s_i \ , \tag{50f}$$

$$\lambda_{i+1} = \frac{(A\phi_{i+1}, B\phi_{i+1})}{(B\phi_{i+1}, B\phi_{i+1})} \ , \tag{50g}$$

$$r_{i+1} = \lambda_{i+1} B\phi_{i+1} - A\phi_{i+1} \ , \tag{50h}$$

$$\begin{aligned}\beta_i = & -[(AK^{-1}r_{i+1}, As_i) - \lambda_{i+1}\{(AK^{-1}r_{i+1}, Bs_i) + (As_i, BK^{-1}r_{i+1})\} \\ & + \lambda^2_{i+1}(BK^{-1}r_{i+1}, Bs_i)] \\ & / [(As_i, As_i) - 2\lambda_{i+1}(As_i, Bs_i) + \lambda^2_{i+1}(Bs_i, Bs_i)] \ ,\end{aligned} \tag{50i}$$

$$s_{i+1} = K^{-1}r_{i+1} + \beta_i s_i \ . \tag{50j}$$

To avoid the singularity of $A - \lambda^* B$, we adopted the incomplete factorization of A

$$K = \tilde{L}\tilde{D}\tilde{U} \ (= A - E) \ , \tag{51}$$

where $\| E \|$ is hopefully small compared with $\| A \|$.

The six parallel lines shown in Fig. 1 denote non-zero elements of the coefficient matrix A arising from the five-point discrete approximation to the two-group two-dimensional ($m \times n$ meshes) diffusion equation. The elements of penta-diagonal part (b_i, c_i, d_i, c_{i+1} and b_{i+m}) denote the summation of the discretized Laplacian operator and neutron removal term. The elements of the mn-th lower diagonal part a_i denote the neutron slowing-down-in term.

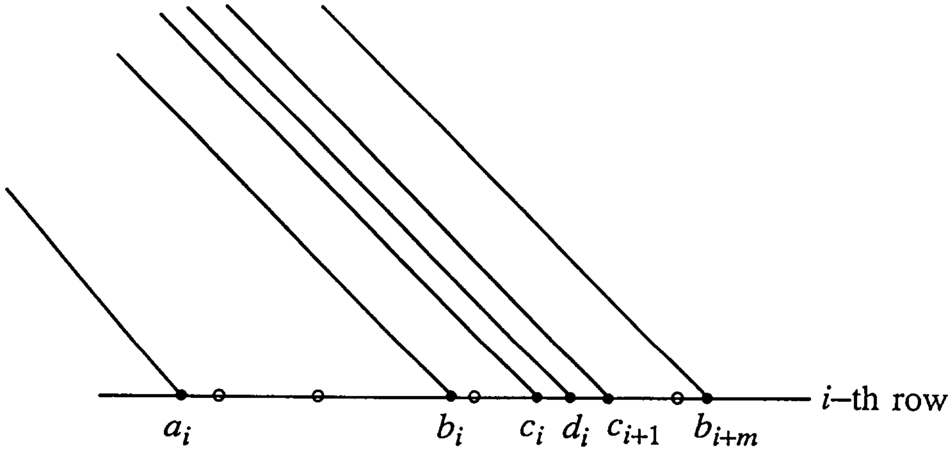

Figure 1: Non-zero elements of coefficient matrix A.

Here, the matrix K is chosen so that its incomplete factorization factor $\tilde{L}$ and $\tilde{U}$ have the same sparsity pattern as the lower triangular and upper triangular part of A, so that we need only compute and store the diagonal matrix $\tilde{D}$. If the diagonal elements of $\tilde{L}$, $\tilde{D}$ and $\tilde{U}$ are denoted by $\tilde{l}_i$, $\tilde{d}_i$ and $\tilde{u}_i$, then the following relations hold:

$$\tilde{l}_i = \tilde{u}_i = d_i - b_i^2 \tilde{d}_{i-m} - c_i^2 \tilde{d}_{i-1} \,, \tag{52a}$$

$$\tilde{d}_i = \tilde{l}_i^{-1} = \tilde{u}_i^{-1} \,, \tag{52b}$$

The matrix $\tilde{L}\tilde{D}\tilde{U}$ is equal to A except for four extra diagonal elements, $a_i c_{i-mn+m} \tilde{d}_{i-mn}$, $a_i b_{i-mn+m} \tilde{d}_{i-mn}$; $b_i c_{i-m+1} \tilde{d}_{i-m}$ and $c_i b_{i-1+m} \tilde{d}_{i-1}$, as indicated by the circular marks ○ shown in Fig. 1. The effect of these non-zero elements can be diminished by introducing the following modified incomplete factorization (Gustafsson, 1978)

$$\begin{aligned} \tilde{l}_i = \tilde{u}_i = (1+\delta) d_i - b_i^2 \tilde{d}_{i-m} - c_i^2 \tilde{d}_{i-1} - a_i c_{i-mn+1} \tilde{d}_{i-mn} \\ - a_i b_{i-mn+m} \tilde{d}_{i-mn} - b_i c_{i-m+1} \tilde{d}_{i-m} - c_i b_{i-1+m} \tilde{d}_{i-1} \,, \end{aligned} \tag{53a}$$

$$\tilde{d}_i = \tilde{l}_i^{-1} = \tilde{u}_i^{-1} , \tag{53b}$$

where δ is a small positive value to avoid $\tilde{d}_i < 0$. If $\tilde{d}_i$ becomes negative, then we cannot perform the incomplete factorization and the matrix $\tilde{L}\tilde{D}\tilde{U}$ is no longer appropriate for the approximation of A.

Unfortunately, numerical experiments for few-group eigenvalue problems have shown that the OR method with the modified incomplete factorization did not converge at all (Suetomi and Sekimoto, 1989). The reason why the OR method with the modified incomplete factorization does not converge for these cases is due to the fact that some of the elements $\tilde{d}_i$ are not positive.

If the matrix A is strictly diagonally dominant, the penta-diagonal part of A is strictly diagonally dominant

$$|d_i| > |b_i| + |c_i| + |c_{i+1}| + |b_{i+m}| \,, \tag{54}$$

so that the inequality $\tilde{d}_i > 0$ holds for any positive δ (Gustafsson, 1978). On the other hand, if the slowing-down-in term a_i is greater than the removal term, then the coefficient matrix is not diagonally dominant and the inequality $\tilde{d}_i > 0$ is not always guaranteed.

In the present study, we propose the following two types of incomplete factorizations satisfying the inequality $\tilde{d}_i > 0$.

Scheme 1: The matrices $\tilde{L}$ and $\tilde{U}$ have the same sparsity pattern as the original matrix A.
Scheme 2: The matrices $\tilde{L}$ and $\tilde{U}$ have the same sparsity pattern as the penta-diagonal part of A.

In order to avoid $\tilde{d}_i < 0$, we set $a_i = 0$ during the factorization process. Then the following relations hold for the modified incomplete factorization.

$$\tilde{l}_i = \tilde{u}_i = (1+\delta)d_i - b_i^2\tilde{d}_{i-m} - c_i^2\tilde{d}_{i-1} - b_i c_{i-m+1}\tilde{d}_{i-m}$$
$$- c_i b_{i-1+m}\tilde{d}_{i-1} , \tag{55a}$$

$$\tilde{d}_i = \tilde{l}_i^{-1} = \tilde{u}_i^{-1}, \tag{55b}$$

On the other hand, recurrent relations for the incomplete factorization are equivalent to (52). Calculation of $\tilde{d}_i$ using (52) and (55) are equivalent to the IC factorization and MIC factorization of penta-diagonal part of A, respectively.

Hence, the following schemes are obtained.

IC factorization (Scheme 1): The matrix $\tilde{L}$ and $\tilde{U}$ have the same sparsity pattern as the original matrix A. The diagonal elements of $\tilde{D}$ are given by (52).
MIC factorization (Scheme 1): The matrix $\tilde{L}$ and $\tilde{U}$ have the same sparsity pattern as the original matrix A. The diagonal elements of $\tilde{D}$ are given by (55).
IC factorization (Scheme 2): The matrix $\tilde{L}$ and $\tilde{U}$ have the same sparsity pattern as the penta-diagonal part of A. The diagonal elements of $\tilde{D}$ are given by (52).
MIC factorization (Scheme 2): The matrix $\tilde{L}$ and $\tilde{U}$ have the same sparsity pattern as the penta-diagonal part of A. The diagonal elements of $\tilde{D}$ are given by (55).

We can vectorize the calculation of $q = (\tilde{L}\tilde{D}\tilde{U})^{-1}r$ using the hyperplane method (Ushiro, 1984). In the few-group diffusion problem such as $N_x \times N_y$ meshes and N_g groups calculation, we define the p-th hyperplane as a set of lattice sites, (n_x, n_y, n_g), satisfying

$$n_x + n_y + n_g = h_p , \tag{56}$$

where h_p runs from 3 to $N_x+N_y+N_g$. The average vector length for this problem is

$$l = \frac{N_x \times N_y \times N_g}{N_x + N_y + N_g - 2} . \tag{57}$$

IV. NUMERICAL EXPERIMENTS

IV.1. One-group eigenvalue problem

For a problem of the one-group diffusion equation described in (1), we compared the following six methods:

MICCG: CG method with MIC factorization.

ICCG: CG method with IC factorization.
SCG: Scaled CG method.
CG: CG method.
SOR: Inner-outer iterative scheme (the successive over-relaxation method is used for the inner iteration and the power method without any acceleration for the outer iteration).
CITATION: Multi-group diffusion code (Fowler, *et al.*,).

All calculations were performed on the HITAC M-660K scalar computer in the double precision (64 bit) arithmetic to avoid round-off errors. To evaluate the effect of the hyperplane method on vector computers, we measured the CPU times for the MICCG, ICCG, SCG and SOR methods on two different vector computers. The first one is the HITAC S-820/80 which performs quite well for short vector length and indirect addressing. The second one is the ETA10-E super-computer. Though the ETA10-E has eight central processing units (CPUs) and each CPU is connected to a shared memory, we used only one CPU in the present study. The computations have been carried out in the 64 bit arithmetic.

The details of the test problem are as follows:

Problem 1

An eigenvalue problem of two-dimensional rectangular reactor of heterogeneous medium as shown in Fig. 2 was investigated.

The boundary conditions at the top (y=0cm) and left boundaries (x=0 cm) are reflective, and bottom (y=150 cm) and right boundaries (x=150cm) are vacuum. The mesh widths are $\Delta x = \Delta y = 1.5$ cm, so that the order of matrix eigenvalue problem is $N = 10{,}000$. The group constants shown in Table 1 give strong heterogeneity on the problem.

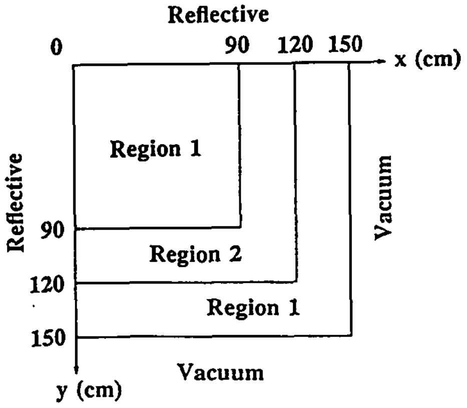

Figure 2: Calculation geometry for Problem 1.

Table 1: Group constants for Problem 1

Region	1	2
D (cm)	1.170	1000
Σ_a (cm^{-1})	0.023	0.0
$\nu\Sigma_f$(cm^{-1})	0.027	0.0

Since the convergence of the SOR method depends sensitively on the truncation number of inner iterations and the acceleration parameter, we performed parameter survey to determine an optimum truncation number of the inner iteration and an optimum acceleration parameter. Two types of convergence criterions were adopted. The first one is for relative residual:

$$\| \lambda_i B\phi_i - A\phi_i \|_2 / \| \lambda_i B\phi_i \|_2 < 10^{-8} , \tag{58}$$

and the second one is for flux change:

$$\max\left[\left|\max_{j} \frac{\phi_{i,j}}{\phi_{i-1,j}} - 1\right|, \left|\min_{j} \frac{\phi_{i,j}}{\phi_{i-1,j}} - 1\right|\right] < 10^{-8} , \qquad (59)$$

where i and j denote the i-th iteration and j-th spatial mesh, respectively. In case of the SOR and CITATION, i denotes the number of outer iteration.

The relative residual norm and flux change for Problem 1 are plotted along the number of iterations in Figs. 3 (a) and (b), respectively. Since the CITATION code dose not have an option for calculating the relative residual, the convergence curve of CITATION is not shown in Fig. 3 (a). In this problem, the CG method did not converge. On the other hand the preconditioned conjugate gradient method based on the MIC factorization (MICCG) required less iterations than the other methods. Table 2 shows the results of required CPU time for satisfying (59) on the scalar computer M-660K and the number of iterations for various methods. The CPU time and number of iterations for the MICCG method were less than the other methods.

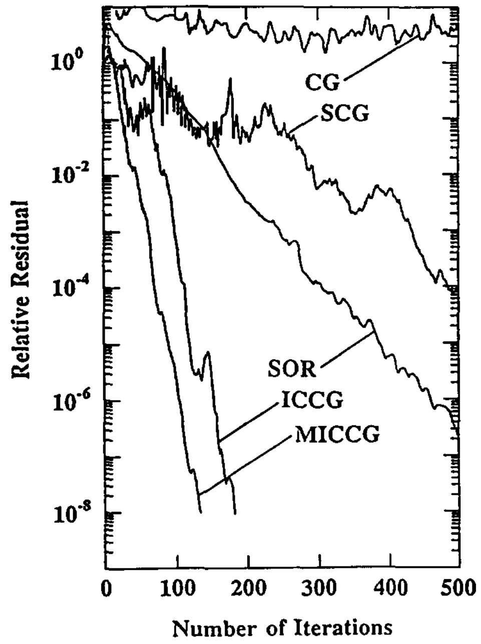

Figure 3(a): Relative residual norm for Problem 1 plotted along number of iterations.

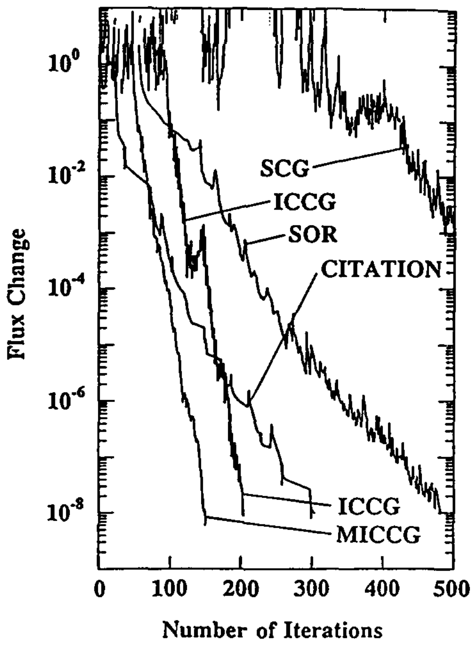

Figure 3(b): Flux change for Problem 1 plotted along number of iterations.

Table 2: CPU time and number of iterations for various algorithms†

MICCG	ICCG	SCG	SOR	CITATION
22.59 (150)	30.44 (203)	139.23 (1019)	39.36 (481)	92.76 (303)

† Unit of CPU time is second and values in parentheses represent number of iterations.

Table 3 shows the required CPU time for satisfying (58) for the various algorithms on the various computers. The hyperplane method was applied to the MICCG, ICCG and SOR method. Since the scalar computer can not treat fast indirect addressing, the hyperplane method was not adopted on the M-660K. The hyperplane method improved in the CPU time on both S-820/80 and ETA10-E.

Table 3: CPU time for Problem 1 on vector and scalar computers†

Computer	MICCG	ICCG	SCG	SOR
S-820/80	0.1516	0.2075	0.6110	0.7479
ETA10-E	1.4335	1.9620	5.6274	5.8060
M-660K††	18.672	25.553	118.10	40.393

† Unit of CPU time is second.

†† CPU times of the M-660K were measured for not hyperplane but original method.

IV.2. Few-group eigenvalue problem

In this section, we examine the following six methods:

MICOR (Scheme 1): OR method with MIC factorization (Scheme 1).
ICOR (Scheme 1): OR method with IC factorization (Scheme 1).
MICOR (Scheme 2): OR method with MIC factorization (Scheme 2).
ICOR (Scheme 2): OR method with IC factorization (Scheme 2).
SOR: Inner-outer iterative scheme (the successive over-relaxation method is used for the inner iteration and the power method without any acceleration for the outer iteration).
CITATION: Multi-group diffusion code.

The following eigenvalue problem for a thermal reactor was investigated.

Problem 2

A two-group eigenvalue problem of the two-dimensional homogeneous reactor shown in Fig. 4 was investigated. The boundary conditions at the top (y = 0 cm) and left boundaries (x = 0cm) are reflective, and bottom (y = 50cm) and right boundaries (x = 50cm) are vacuum. The mesh widths are $\Delta x = \Delta y = 1.0$ cm, so that the order of matrix eigenvalue problem is $N = 5{,}000$. The group constants are shown in Table 4.

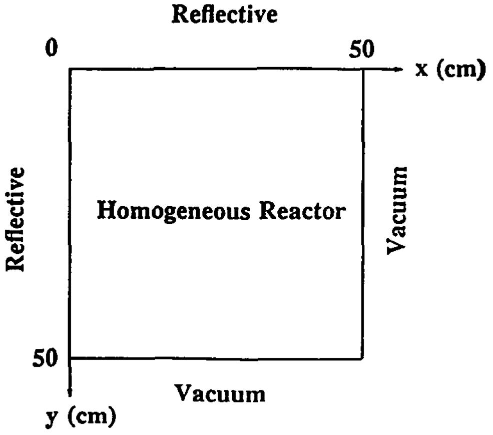

Figure 4: Calculation geometry for Problem 2.

Table 4: Group constants for Problem 2

Group	1	2
D (cm)	1.263×10^{0}	3.543×10^{-1}
Σ_a (cm^{-1})	1.207×10^{-2}	1.210×10^{-1}
Σ_R (cm^{-1})	2.619×10^{-2}	1.210×10^{-1}
$\nu\Sigma_f$(cm^{-1})	8.476×10^{-3}	1.851×10^{-1}

Table 5 shows the required numbers of iterations for Problems 2 for different types of the preconditioner. Both of the Schemes 1 and 2 with the MIC factorization worked well and the number of iterations reduced by half compared with the IC factorization.

Table 5: Number of iterations for Problem 2 for IC and MIC preconditionings

Algorithm	Scheme 1	Scheme 2
ICOR	126	120
MICOR	66	68

The performance of the various algorithms for Problem 2 is shown in Fig. 5. We took the Scheme 1 for the IC and MIC factorization. The CPU time and number of iterations for these algorithms are shown in Table 6, where the convergence criterion is the flux change less than 10^{-10}.

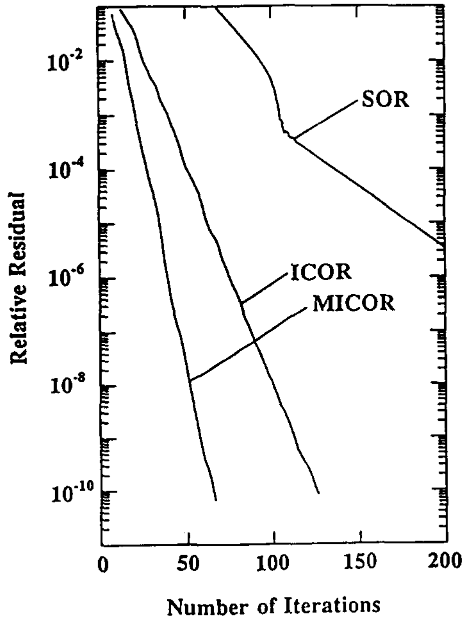

Figure 5(a): Relative residual norm for Problem 2 plotted along number of iterations.

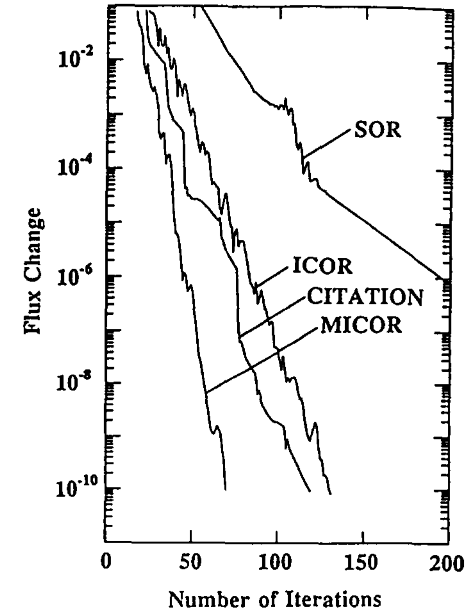

Figure 5(b): Flux change for Problem 2 plotted along number of iterations.

Table 6: CPU time and number of iterations for various algorithms†

MICOR	ICOR	SOR	CITATION
6.18 (70)	11.62 (131)	19.95 (371)	17.01 (119)

† Unit of CPU time is second and values in parentheses represent number of iterations.

Table 7 shows the required CPU time for the above mentioned algorithms on the various types of computer. The convergence criterion is the relative residual norm less than 10^{-10}. It is remarkable that the MICOR method with the hyperplane method overcame other methods.

Table 7: CPU time for Problem 2 on vector and scalar computers†

Computer	MICOR	ICOR	SOR
S-820/80	0.0439	0.0837	1.5436
ETA10-E	0.4210	0.7919	4.8004
M-660K††	5.6318	11.040	18.467

† Unit of CPU time is second.
†† CPU times of the M-660K were measured for not hyperplane but original method.

V. CONCLUSIONS

The CG like methods were applied to the eigenvalue problems for neutron diffusion. These methods do not need the inner-outer iterations and any acceleration parameter.

The CG method to minimize the Rayleigh quotient was proposed to the one-group eigenvalue problems. The convergence was enhanced by using the preconditioner based on the IC or MIC factorization of the coefficient matrix.

In order to calculate the nonsymmetric generalized eigenvalue problems arising from the few-group eigenvalue problems, we proposed the ORTHOMIN(1) method to minimize the residual norm. The ORTHOMIN(1) method was accelerated by the preconditioner based on the IC and MIC factorization.

These programs were fully vectorized using the hyperplane method and implemented on the vector computers: the HITAC S-820/80 and ETA10-E (one processor mode).

The numerical experiments for one- and two-group eigenvalue problems were performed. The MICCG method for the one-group eigenvalue problem and the MICOR method for the few-group eigenvalue problem required much smaller number of iterations and computational time compared with the inner-outer iterative scheme. The significant decrease of CPU time on the vector computers was observed by using the MICCG and MICOR method with the hyperplane method.

REFERENCES

Bradbury W.W. and Fletcher R (1966). *Numer. Math.*, 9, 259.
Eisenstat S.C., Elman H.C. and Schultz M.H. (1983) *SIAM J. Numer. Anal.*, 20, 345.
Fletcher R. and Reeves C.M. (1964) *Comput. J.*, 7,149.
Fowler T.B., Vondy D.R., Cunningham G.W. (1971) *ORNL-TM*-2496, Rev. 2.
Gambolati G., Pini G. and Sartoretto F. (1988) *J. Comput. Phys.*, 74, 41.
Geradin M. (1971) *J. Sound Vibration*, 19,319.
Gustafsson I. (1978) *BIT*, 18, 142.
Hestenes M.R. and Karush W. (1951) *J. Res. Nat. Bur. Standards*, 47,45 (1951).
Meijerink J.A. and Vorst H.A. van der (1977) *Math. Comput.*, 31, 148.
Suetomi E. and Sekimoto H. (1988) *J. Nucl. Sci. Technol.*, 25,100.
Suetomi E. and Sekimoto H. (1989) *Trans. IPS Japan*, (in Japanese), 30, 661.
Ushiro Y. (1984) *RIMS Kokyuroku (Trans. Res. Inst. Mathematical Sci., Kyoto Univ.)*, (in Japanese), 514, 110.
Vinsome P.K.W. (1976) *Proc. Fourth Symposium on Reservoir Simulation*, Society of Petroleum Engineers of AIME, 149.

APPENDIX

ORTHOMIN(1) Algorithm

The ORTHOMIN(1) (also called as conjugate residual) method for solving the systems of linear equations

$$A\phi = b \tag{A1}$$

can be written as follows:

$$Choose\ \phi_0 \ . \tag{A2a}$$

$$Compute\ r_0 = b - A\phi_0 \ . \tag{A2b}$$

$$Set\ p_0 = r_0 \ . \tag{A2c}$$

For $i=0$ Step 1 Until Convergence Do

$$a_i = \frac{(r_i, Ap_i)}{(Ap_i, Ap_i)} \ , \tag{A2d}$$

$$\phi_{i+1} = \phi_i + a_i p_i \ , \tag{A2e}$$

$$r_{i+1} = r_i - a_i Ap_i \ , \tag{A2f}$$

$$\beta_i = -\frac{(Ar_{i+1}, Ap_i)}{(Ap_i, Ap_i)} , \tag{A2g}$$

$$p_{i+1} = r_{i+1} + \beta_i p_i , \tag{A2h}$$

The choice of a_i in (A2d) minimizes $\| r_{i+1} \|_2 = \| b - A(\phi_i + a_i p_i) \|_2$. Therefore, the ORTHOMIN(1) iterates ϕ_i minimize

$$\| r \|_2 = \sqrt{(r, r)} , \tag{A3}$$