AC032

The First International Conference on Supercomputing in Nuclear Applications (SNA '90) pp. 314-319.

# FAST NUMERICAL SOLUTION OF NEUTRON DIFFUSION EQUATION

Katsuyoshi Sotani
NEC Corporation
(KNES, LTD.)
4-24, Shiromi 1-chome, Chuo-ku, Osaka, Japan

## 1. Introduction

The neutron diffusion equation program has been studied by nuclear fission research organization in various countries and put to practical use. In the general solution, the process is divided into inner iteration (calculation of neutron flux) and outer iteration (calculation of neutron source), and the calculation is completed with the precision inherent in the solution based on the "SOR method system", which is a sound solution. We wish to report here that in NEC's high-speed version, the relative residual norm is defined in the neutron diffusion equation, and the minimum value of the truncation error with the smallest calculation time is sought through numerical experiments. To determine the calculating algorithm, the tridiagonal approximate factorization method capable of parallel processing suitable for supercomputers is used, and in the fundamental iteration, by the adoption of the general biconjugate gradient method, speeds three times greater than those obtained by general conventional solutions have been achieved.

## 2. Neutron Diffusion Equation

In designing a reactor, the most important condition is steady maintenance of the nuclear fission chain reaction, and the calculations are performed to this end, making changes for the composition of the fuel, the size of the reactor, etc. Here, the neutron diffusion equation is used for theoretical simulation. This means that solutions are obtained by numerical calculations of the eigenvalue problem of elliptic partial differential equations.

Eigenvalue Problem in Neutron Diffusion Equation:

$$\mathbf{B} \bullet \Phi = \frac{1}{K_e} \bullet \mathbf{F} \bullet \Phi \qquad (1)$$

$\mathbf{B}$ : Neutron transport scattering absorption coefficient
$\Phi$ : Neuron flux
$\mathbf{F}$ : Neutron source coefficient
$K_e$ : Eigenvalue of effective multiplication coefficient

$$\Phi = \begin{bmatrix} \phi 1 \\ \bullet \\ \bullet \\ \bullet \\ \phi n \end{bmatrix}$$

Expression by general equation:

$$-D^g(x,y)\nabla^2\phi^g(x,y) + \sum_t^g(x,y)\phi^g(x,y) = \frac{1}{K_e}\chi^g \sum_{g'=1}^{G} \nu \sum_1^{g'}(x,y)\Phi^{g'}(x,y) + \sum_{g>g'} \sum_r^{g-g'}(x,y)\Phi^{g'}(x,y) \qquad (2)$$

Where,

$$\sum_T^g(x,y) = \sum_a^g(x,y) + \sum_{g>g'} \sum_r^{g-g'}(x,y) + D^g B_k(x,y)$$

g : Group number ($g_1$ : fast neutron, $g_2$ : thermal neutron)

$\sum_f$ : Nuclear cross section

$\Phi$ : Neutron flux

D : Diffusion Coefficient

$B_k$ : Longitudinal buckling

$\sum_r^{g-g'}$ : Scattering removal cross section from group g to group g'

$\sum_a$ : Absorption cross section

$\nu$ : Mean number of neutron particles discharged in a single nuclear fission

$\chi$ : Neutron spectrum ($\sum_{g=1}^{G} \chi^g = 1$)

Equation (2) has a solution other than $\Phi(x,y) = 0$ for a specific $K_e$. This value is the eigenvalue for equation (2) (principle of neutron conservation)

When we change equation (2):

$$-D^g(x,y)\nabla^2\phi^g(x,y) + [\sum_T^g(x,y)\Phi^g(x,y) - \sum_{g>g'} \sum_r^{g-g'}(x,y)\Phi^{g'}(x,y)\,] = \frac{1}{K_e}\chi^g \sum_{g'=1}^{G} \nu \sum_f^{g'}(x,y)\Phi^{g'}(x,y) \qquad (3)$$

Equation (3) expresses the following relation :

Leak + absorption = generation

In case the maximum eigenvalue of equation (2) is < 1 , the following is realized:

Leak + absorption > generation

Thus, the nuclear fission reaction will not last.

When the maximum eigenvalue of equation (2) is > 1, the following is realized ;

Leak + absorption < generation

Thus, the reactor will run out of control.

Hear, the maximum eingenvalue is the effective multiplication coefficient. In this code, the calculations are performed by discretization of the two-dimensional multi-group neutron diffusion equations of other groups with the differential method by mesh division by energy group and space. Space discreteness is done by five-point differentiation.

## 3. Composition of Neutron Diffusion Equation Simulation Code

The simulation code is composed of outer iteration and inner iteration.

Outer iteration: Calculation of energy distribution of neutron source at various points in space

Inner iteration: Calculation of neutron flux per energy group in space

The solutions were sought by iteration calculation of simultaneous linear equations.

Equation (2) undergoes discretization by differential approximation, and becomes an eigenvalue problem with $\Phi_{ijk}$ in equation (1) as the eigenvector.

The maximum eigenvalue of equation (2) becomes the effective multiplication coefficient.

The maximum eigenvalue for equation (2) is obtained by the power method of following equations.

$$\Psi^{(t+1)} = \mathbf{B}^{-1}\mathbf{F}\phi^{(t)} \qquad (4)$$

$$K_e^{(t+1)} = \frac{\mathbf{F}\Psi^{(t+1)}}{\mathbf{F}\phi^{(t)}} \qquad (5)$$

$$\phi^{(t+1)} = \frac{1}{K_e^{(t+1)}}\Psi^{(t+1)} \qquad (6)$$

The process of repeatedly seeking $\phi^{(1)}$, $\phi^{(2)},\ldots,\phi^{(t)}$ by the iteration of (4),(5),and (6) is called "outer iteration".

In solving equation (4), the method in which this is achieved by repeating $\Psi^{(t+1)}$ while calculating $\phi^{(t+1)}$ given $\phi^{(0)}=1$, using iteration of (7) and (8) below, is called "inner iteration".

$$\mathbf{S}^{(i)} = \mathbf{F}\phi^{(i)} \qquad (7)$$

$$\mathbf{B}\Psi^{(i+1)} = \mathbf{S}^{(i)} \qquad (8)$$

## 4. Improving the Neutron Diffusion Method

In this section we present comparison criteria for high-speed technology.

### 4-1. Outline of calculations in general original codes

Convergence is achieved forcibly by performing inner iteration (calculation of neutron flux by paired SLOR in line direction and SLOR in row direction) once.

$\phi_i$ : Neutron flux

$K_e$ : Eigenvalue

Condition of convergence by outer iteration:

$$\left|\frac{\phi_i^{(t+1)} - \phi_i^{(t)}}{\phi_i^{(t+1)}}\right| < \varepsilon_1, \quad \left|\frac{K_e^{(t+1)} - K_e^{(t)}}{K_e^{(t+1)}}\right| < \varepsilon_2$$

$\phi_i$ which are the object of judgment are not all the $\phi_i$ being calculated by inner iteration, but they are restricted to the portion where the fuel rod exists in the domain targeted by the calculation.

There are no calculations for the residual norm, so it is a rough calculating method.

### 4-2. Summary of calculations of the high-speed version of the NEC method

True solution $\mathbf{x}$ and approximate solution $\tilde{\mathbf{x}}$ of the system of linear equations $\mathbf{A}\mathbf{x} = \mathbf{b}$

$$\mathbf{r} = \mathbf{b} - \mathbf{A}\tilde{\mathbf{x}}$$

Relative residual norm $n_r = \frac{\|\mathbf{r}\|}{\|\mathbf{b}\|}$ is defined.

Adding $n < \varepsilon_3$, the inner iteration calculations are performed. We seek the minimum value of this $\varepsilon_3$, which requires the least calculating time and is highly accurate, through numerical experiments.

The process of outer iteration is similar. As a result, a more exact solution can be realized. The sequence of operation is based on the TFBCG method.

## 5. Numerical Calculation for High Speed Technology

### 5-1. Comparison of SLOR and TFBCG methods

It is well known that the iteration method is of generally greater calculating efficiency than the direct method. The Jacobi method, Seidel method, Householder method, Power method, etc. that have been widely accepted as

the classic solutions among the iteration methods enable solutions to be obtained regardless of the symmetry or non-symmetry of the matrix, the calculations evolve quantitatively, and they necessarily converge, but the calculation steps are fixed and it takes time to reach the target.

In recent years, the SOR method system (SOR method, SLOR method, Point SOR method, Point Block SOR method, 2-line SOR method, Odd-Even SOR method, etc.) has improved this situation.

In this method, the acceleration parameter in the over-relaxation method is set at the best point, and a solution stable at high speed is sought. The speed is generally four to five times higher than that realized by the classic iteration method.

The SLOR method can be applied regardless of whether matrix is symmetric or non-symmetric, and regardless of the positive or non-positive value, besides even if the condition number is large, convergence is possible in a sure manner at relatively high speeds.

On the other hand, for the matrix, the TF method system is used to improve the number of conditions, leading to a crowd eigenvalue's form by implementing an appropriate precondition (scaling effect). In terms of geometry, it means that a hyper ellipsoid in the space of an eigenvalue of n dimensions approaches a spherical body, making numerical solutions easier. The fact that the gradient method is applied after the precondition means that convergence to the solution is achieved in the shortest distance.

Even among the gradient method systems, in the BCG method system (biconjugate gradient method), even when the number of conditions is relatively large for the matrix revealing no tendency to degenerate, stable solutions can be obtained.

The TFBCG method is an algorithm enabling the full operation of the vector operating pipeline of the supercomputer. It is a new solution that is rearranged into a parallel system by the use of tridiagonal approximate factorization method suitable for non-symmetric sparse matrixes, and mixed with the general biconjugate gradient method as the fundamental iteration method of the whole.

### 5-2. Basic concept of conjugate gradient method

When the coefficient matrixes in large scale simultaneous linear equations are sparse, the conjugate gradient method is excellent. The conjugate gradient method is not very effective when doing the calculations with general-purpose computers.

The major portion of the algorithm is composed of the product and inner product of matrixes and vectors and the operations are suited to computers which are capable of parallel calculation, so we believe it is a method ideal for supercomputers.

If matrix $\mathbf{A}$ of the following equation

$$\mathbf{A}\,\mathbf{x} = \mathbf{b} \tag{9}$$

is a positive symmetric matrix of $n \times n$, the following equation

$$F(\mathbf{x}) = \frac{1}{2}(\mathbf{x}, \mathbf{A}\mathbf{x}) - (\mathbf{x}, \mathbf{b}) \tag{10}$$

can be minimized by selecting a suitable vector $\mathbf{x}$.

Then, starting form an appropriate initial value $x_0$, we will make the $x_1, x_2, \ldots$ line of the adjustments, and finally reaching the minimum points $x_{min}$ of $F(\mathbf{x})$. Then a search will be conducted in the next direction. If the next approximate vector is $\mathbf{x}_{k+1}$, the equation is as follows:

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{p}_k \tag{11}$$

When $\mathbf{p}_k$ of equation (11), the search direction, is decided, the point where $F(\mathbf{x})$ of equations (10) is minimized in that direction is:

$$\frac{\partial F(\mathbf{x}_k + \alpha_k \mathbf{p}_k)}{\partial \alpha_k} = 0 \tag{12}$$

Where, $\alpha_k = \dfrac{(\mathbf{p}_k, \mathbf{r}_k)}{(\mathbf{p}_k, \mathbf{A}_k \mathbf{p}_k)}$

$\mathbf{r}_k$ is the residual difference of approximate vector $\mathbf{x}_k$.

$$\mathbf{r}_k = \mathbf{b} - \mathbf{A}\mathbf{x}_k$$

For selecting $\mathbf{p}_k$, correction of the searching direction of the preceding step is done in the direction of the most urgent descent $\mathbf{r}_k$.

$$\mathbf{p}_k = \mathbf{r}_k + \beta_{k-1} \mathbf{p}_{k-1}$$

$\mathbf{p}_k$ and $\mathbf{p}_{k-1}$ become linearly independent and are determined so that their relations with $\mathbf{A}$ will become conjugate, satisfying the following equation.

$$(\mathbf{p}_k, \mathbf{A}\mathbf{p}_{k-1}) = 0 \tag{13}$$

The following equation is now obtained:

$$\beta_k = -\frac{(\mathbf{r}_{k+1}, \mathbf{A}\mathbf{p}_k)}{(\mathbf{p}_k, \mathbf{A}\mathbf{p}_k)}$$

This $\mathbf{p}_k$ becomes conjugate with all $\mathbf{p}_{k-n}$, and the following equation is obtained:

$$(\mathbf{p}_i, \mathbf{A}\mathbf{p}_j) = 0 \quad (i \neq j) \qquad (14)$$

Similarly, $\mathbf{r}_k$ also satisfies the following equation.

$$(\mathbf{r}_i, \mathbf{r}_j) = 0 \quad (i \neq j) \qquad (15)$$

In the vector space of the n-dimension, the residual difference $r_n$ of $x_n$, obtained when searching is conducted n times, is zero, and the solution sought becomes $x_n$.

### 5-3. Tridiagonal approximate factorization method with preconditioned iterative method

**a.** Problem in incomplete LU factorization method

Equation (9) is defined as n sparse simultaneous linear equations having regular features, The preconditioned iterative method is for the following equation (16) equivalent to the equation (9).

$$(\mathbf{A}\mathbf{M}^{-1})(\mathbf{M}\mathbf{x}) = \mathbf{b} \qquad (16)$$

In order to achieve high speeds suitable to the vector computer, the equation (9) is changed into equation (16), and calculations are performed in combination with the fundamental iterative method.

As precondition, in the incomplete LU factorization method generally used, if it is performed only on non-symmetric elements of $\mathbf{A}$, then the approximate matrix of $\mathbf{A}$ will be obtained.

$$\mathbf{M}_{\mathrm{ILU}} = \mathbf{L}\mathbf{U} \qquad (17)$$

$\mathbf{M}^{-1}$ in equation (16) is calculated by advance and retreat substitutions. For instance, advance substitution $\mathbf{v} = \mathbf{L}^{-1}\mathrm{g}$ is developed, and becomes as follows:

$$v_{i,j} = (g_{i,j} - l_{i,j-1} \bullet v_{i,j-1} - l_{i-1,j} \bullet v_{i-1,j}) / l_{i,j}$$

When calculations are performed in the direction of i, no parallel proceeding is possible since there is a reference relation between $v_{i-1,j}$ and $v_{i,j}$. The commonly used incomplete LU factorization method like equation (17) is not fully suitable for vector computers.

**b.** Conditions necessary to take advantage of tridiagonal approximate factorization method

The precondition matrix is determined by a second order equation.

$$\mathbf{M}_{\mathrm{TF}} = (\mathbf{D}. + \mathbf{A}_x)\mathbf{D}.^{-1}(\mathbf{D}. + \mathbf{A}_y) \qquad (18)$$

1) Modification to simplify calculation of inverse matrix

LU factorization is performed for each factor prior to the iteration on equation (18), yielding a second order equation.

$$\mathbf{M}_{\mathrm{TF}} = \mathbf{L}_x\mathbf{U}_x\mathbf{D}.^{-1}\mathbf{L}_y\mathbf{U}_y \qquad (19)$$

By simple conversion the diagonals of $\mathbf{L}_x\mathbf{U}_x$ and $\mathbf{L}_y\mathbf{U}_y$ can be made into 1. In equation (19), the nearer the coefficient matrix is to the unit matrix, the faster it can converge.

2) Ease of parallelism

Inverse matrix calculation is performed for each factor, and if the factor in X direction:

$$(\mathbf{D}. + \mathbf{A}_x)\mathbf{v} = \mathbf{g} \qquad (20)$$

is solved, inverse matrix calculation becomes possible.

Equation (20) is simultaneous linear equation of $n_x$ independent equations simultaneously existing in the X direction of $n_y$ only. As for the advance and retreat substations, there are no mutual reference relations among the lines. Vectorization, therefore, is possible.

3) Convergence

For the conjugate gradient method, convergence is rapid if the eigenvalues of the coefficient matrix $\mathbf{A}$ are identical or closely clustered. This means that the precondition matrix should be an approximation of the original coefficient matrix.

**c.** TFBCG method

As for the preconditioning method, the tridiagonal approximate factorization method (TF method) is adopted, and with the fundamental iteration method, BCG (biconjugate gradient) method, usable in non-symmetric problems, is used. BCG method, a kind of conjugate gradient method, is a commonly used iteration method with high stability, so that a solution can be obtained by using biconjugate base even if the matrix $\mathbf{A}$ is non-symmetric.

## 6. Evaluation of Results

The SLOR method is derived form the classic SOR method system of the original version, so a large number of iterations is required to approach the solution. In a single inner iteration, the residual in the initial period is large.

The final number of outer iterations is determined through the precision inherent in the SLOR method. The results of the original, general version are nearing convergence toward the true solution.

By setting the convergence parameter $\varepsilon$ higher, a solution closer to the true solution can be obtained. However, a very long time will be required for that purpose. The solution in the original, general version was realized, taking into consideration the general purpose computers whose performance imposes limitations on the convergence parameter $\varepsilon$ during the inner iteration, so a compromise is made to a certain extent regarding the accuracy of the solution. Extrapolation in the following form is used to supplement the poor convergence.

$$\phi_{i,\infty} \sim \phi_{i,t} + Z_t(\phi_{i,t} - \phi_{i,t-1}) \qquad (21)$$

Where,

$\phi_{i,\infty}$ : True eigenvector for the maximum eigenvalue of the iterative matrix

$\phi_{i,t}$ : Eigenvector obtained by outer iteration performed t times

$Z_t$ : Extrapolation coefficient

In the NEC version, we are able to reduce the number of outer iterations by controlling $n_r = \frac{\|\mathbf{r}\|}{\|\mathbf{b}\|}$, the relative residual norm of the inner iteration, to increase the accuracy of the solution. By enhancing the accuracy of inner iteration, we have been able to reduce the outer iteration. The solution obtained by this method has a higher degree of accuracy, so it is closer to the true solution than the original, general version.

Since the neutron diffusion equation makes use of the multi-group diffusion theory, the neutron term is successively renewed accompanying the number outer iterations. Therefore, so long as the number of outer iterations is small, it is not necessary to treat convergence of the inner iteration as a problem. In the analysis of the delayed critical conditions, we must consider, in addition, that the physical value, change and adjustment in control material accompanying the changes of effective multiplication coefficient. When the changes of the physical value and adjustments in the control material accompanying the increse of outer iterations become small, it will be more effective to make into account the convergence in the inner iteration calculations to a certain extent, and the number of outer iterations becomes reduced as a result.

## 6-1. Comparison of calculated results

Neutron Diffusion Equation Code

[NEC SX2 Results of Measurement]

Table 1

| Present Level (Rough calculation) | | | Level Closer to True Solution (Exact calculation) | | |
|---|---|---|---|---|---|
| Condition | Time | Acutually measured ratio | Condition | Time | Acutually measured ratio |
| Original, general version<br><br>Tuning of grammar for super-computer | 194 sec (1) | | Original, general version<br><br>Tuning of grammar for super-computer <Increase in precision of solution> | 282 sec (3) | |
| NEC version (High-speed version)<br><br>Changing algorithmn (including tuning of grammar) | 65 sec (2) | 3 times (1)/(2) | NEC version (High-speed version)<br><br>Changing algorithmn (including tuning of grammar) <Increase in precision of solution> | 76 sec (4) | 3.7 times (3)/(4) |

Rough calculation:

Gives the allowable range of the neutron diffusion equation solution for practical use.

Exact calculation:

Gives a solution closer to the true solution than the rough calculation and resolves more precise physical phenomena.

In table 1, (1) and (2) are the actually measured rations by changing algorithm at a level of the present condition (rough calculation) by SX2.

(3) and (4) are the actually measured ratios by changing algorithm at a level close to the true solution (exact calculation) by SX2.

### 6-2. Progress of solution

Comparison of Execution:
Diagram of relationship between convergence of solution and calculating time of original, general solution, and improved NEC solution

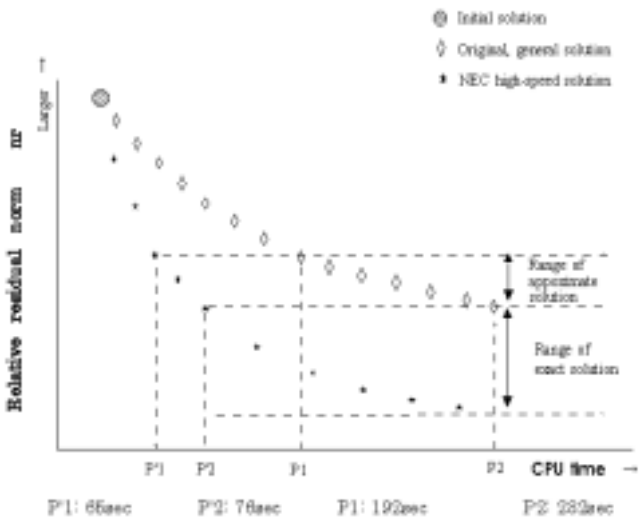


Figure 1

### 6-3. Convergence of original, general solution and of NEC solution

Difference in convergence form the initial solution to approximate solution and exact solution

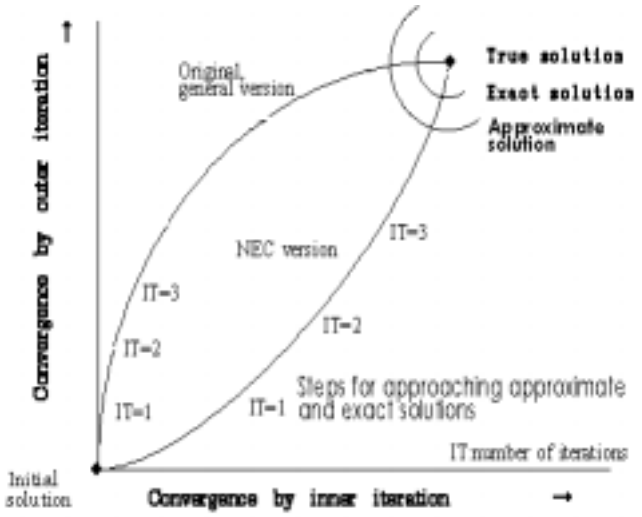


Figure 2

### 7. Conclusion

In solving the neutron diffusion equation eigenvalue problem(1), the SLOR method based only on the criterion of the accuracy of convergence of the original, general solution, has a small number of steps per iteration, and approaches the approximate and the exact solution slowly. (See Figures 1 and 2).

The NEC solution, on the other hand, by using the relative residual norm, improves the balance of the outer and inner iterations, then applies the TFBCG method. The number of steps per iteration is thus large, but convergence to the approximate and to the exact solution is very rapid.

### 8. Study

The difference in the above value tends to become greater as the problems become more complicated by mesh fractionalizations and by multiple group-related problems.

### References

1)"Vectorization of Nuclear Codes and Numerical Mthods "
Yasuo Tokunaga, Hiro Harada, and Misako Ishiguro
Computing Center, Tokai Research Establishment, JAERI M 85-143

2)"Tridiagonal Approximate Factorization Method: A Preconditioning Technique for Solving Nonsymmetric Linear Systems Suitable to Supercomputers"
Shun Doi and Norio Harada (NEC)
National Aero Space Laboratory,Special Data7

3)"A Fast Computing Technique for Diffusion-type Equations"
Sekiya and Sakai (OSAKA UNIV.)
Journal of Computational Physics, Vol 65. NO. 2 August 1986

4) "Nuclear Physics Lectureship 10"
Koji Fushimi
Kyoritsu Publication

5) "Super Computer"
Kenrou Murata, Tsutomu Oguni,and Yukihiko Karaki
Maruzen CO, LTD.