Recent Advances in STRAUM (S_N Transport for Radiation Analysis with Unstructured Meshes) Code

MyeongHyeon Woo, Ser Gi Hong *
Department of Nuclear Engineering, Hanyang University, 222, Wangsimni-ro, Seongdong-gu. Seoul, Korea
**Corresponding author: hongsergi@hanyang.ac.kr*

1. Introduction

The STRAUM (S_N Transport for Radiation Analysis with Unstructured Meshes) code is a deterministic code which has been developed for radiation transport calculation with S_N method and unstructured tetrahedral meshes [1]. In this code, Discontinuous Galerkin (DFEM) and LDEM-SCB [2] spatial discretization methods were employed to solve the particle transport equation. The STRAUM code uses unstructured tetrahedral meshes generated by Gmsh [3] after importing a CAD file. Recently, we wrote an in-house program to automatically generate the mesh files for STRAUM by processing the meshes files generated by Gmsh. Previously, the transport calculation in STRAUM was performed using the scattering source iteration. In particular, the DSA method in which the discretized diffusion equations were derived by consistently discretizing the continuous diffusion equation to LDEM-SCB was used to accelerate the scattering source iteration of LDEM-SCB while the DFEM discretization was not accelerated. Since the 2000s, the Krylov method has become the standard method for S_N equation rather than the conventional scattering source iteration [4]. Consistently to this trend, the Krylov method has been implemented in STRAUM and the DSA can be optionally used as a preconditioner for LDEM-SCB. Also, we employed the Krylov method to perform the multigroup coupling iteration involved to the up-scattering rather than the Gauss-Seidel iteration.

This paper describes these recent advances in the STRAUM code and the preliminary test results.

2. Description of Transport Calculation Method

The steady-state neutron transport equation is given by

$$\hat{\Omega}\cdot\nabla\psi(\vec{r},E,\hat{\Omega})+\sigma_t(\vec{r},E)\psi=q(\vec{r},E,\hat{\Omega}), \qquad (1)$$

where ψ is angular flux, σ_t is total macroscopic cross section, and q is source term. In a fixed source problem, the source term is defined as the sum of scattering and external source terms as follows.

$$q=q_s+q_{ex}, \qquad (2)$$

where q_{ex} is external source term, and q_s is scattering source term is given by.

$$q_s=\int_0^\infty dE'\int_{4\pi}d\hat{\Omega}'\sigma_s(\vec{r},E'\to E,\hat{\Omega}'\cdot\hat{\Omega})\psi(\vec{r},E',\hat{\Omega}). \qquad (3)$$

Applying multigroup approximation to Eq. (1) and considering only discrete angles, the Eq. (4) can be obtained, which is called S_N method.

$$\hat{\Omega}_n\cdot\nabla\psi_n^{\mathrm{g}}(\vec{r})+\sigma_t^{\mathrm{g}}(\vec{r})\psi_n^{\mathrm{g}}(\vec{r}) = \sum_{\mathrm{g}'=1}^{G}\sum_{\ell=0}^{L}\sum_{m=-\ell}^{\ell}(2\ell+1)\sigma_{s,\ell}^{\mathrm{g}'\to\mathrm{g}}R_{\ell m}(\hat{\Omega})\phi_{\ell m}(\vec{r})+q_{ex,n}^{\mathrm{g}}(\vec{r}), \qquad (4)$$

where

$\psi_n^{\mathrm{g}}\equiv\psi^{\mathrm{g}}(\hat{\Omega}_n)$

$\sigma_t^{\mathrm{g}}=$ macroscopic total cross section for group g at $\vec{r}$

$\sigma_{s,\ell}^{\mathrm{g}'\to\mathrm{g}}=$ moment ℓ of the scattering cross section from g′ to g.

$R_{\ell m}=$ tesseral spherical harmonics function,

$\phi_{\ell m}=$ flux moment.

The flux moments are calculated by spherical integration using the quadrature rules as shown in Eq. (5).

$$\phi_{\ell\mathrm{m}}=\int_{4\pi}R_{\ell m}(\hat{\Omega})\psi(\hat{\Omega})d\Omega\cong\sum_{n=1}^{N}w_n\psi_nR_{\ell m}(\hat{\Omega}_n), \qquad (5)$$

where $\hat{\Omega}_{\mathrm{n}}$ is a discrete ordinate and w_n is the weight associated with this ordinate. In the STRAUM code, the Gauss-Chebyshev quadratures are used with weight normalization to 1.0 as follows :

$$\sum_{n=1}^{N}w_n=1.0. \qquad (6)$$

The transport equation expressed in Eq. (4) can be written by an operator form as

$$\mathbf{L}\psi=\mathbf{MS}\phi+q_{ex}=\mathrm{Q}, \qquad (7)$$

where **L** is the transport operator comprised of streaming and total collision terms, **MS** is the moment-to-discrete operator. Explicitly defining operators as matrix is inefficient in terms of memory computational speed and so a matrix-free fashion is applied in this work. In particular, $\mathbf{L^{-1}}$ implicitly defined as a sweeping operator that updates outgoing fluxes using incoming fluxes while crossing cells ordered in the directions. The operator from angular flux to moments is defined as **D**

$$\phi=\mathbf{D}\psi. \qquad (8)$$

Using these operators, conventional source iteration can be written with the iteration index *k* as follows :

Algorithm. Source Iteration

Step 1. $\phi^{(k+1/2)} = \mathbf{DL}^{-1}\left(\mathbf{MS}\phi^{(k)} + q_{ex}\right)$,
Step 2. $\phi^{(k+1)} = \phi^{(k+1/2)}$.

When DSA is applied, the diffusion equation is solved using the difference between the new scalar flux calculated through sweeping and the flux of the previous iteration as the source term, which can be expressed as the following procedure.

Algorithm. Source Iteration with DSA

Step 1. $\phi^{(k+1/2)} = \mathbf{DL}^{-1}\left(\mathbf{MS}\phi^{(k)} + q_{ex}\right)$,
Step 2. $\delta\phi^{(k+1/2)} = \mathbf{C}^{-1}\mathbf{S}\left(\phi^{(k+1/2)} - \phi^{(k)}\right)$,
Step 3. $\phi^{(k+1)} = \phi^{(k+1/2)} + \delta\phi^{(k+1/2)}$.

Here, $\mathbf{C}$ is the diffusion operator is defined in Eq. (9). In the STRAUM, $\mathbf{C}$ is obtained by discretizing the continuous diffusion equation consistently with LDEM-SCB.

3. Implementation of Krylov Subspace Method

For a detailed description of applying the Krylov method for S_N equation is described in reference [5]. To apply the Krylov method, first, Eqs. (7) and (9) are transformed to following form :

$$(\mathbf{I} - \mathbf{DL}^{-1}\mathbf{MS})\phi = \mathbf{DL}^{-1}\mathrm{Q}\,, \tag{9}$$

With use of DSA as a left-preconditioner, then it can be expressed as follows :

$$(\mathbf{I} + \mathbf{PC}^{-1}\mathbf{RS})(\mathbf{I} - \mathbf{DL}^{-1}\mathbf{MS})\phi = (\mathbf{I} + \mathbf{PC}^{-1}\mathbf{RS})\mathbf{DL}^{-1}\mathrm{Q}\,, \tag{10}$$

where $\mathbf{R}$ and $\mathbf{P}$ are restriction and projection operators, respectively, which are defined to extract the scalar flux required for the diffusion equation from the higher moment and to correct it.

2.1. Data Structure for Moment Vectors & Linear Algebra Operations

From Eqs.(10) and (11), it can be seen that the two equations have a suitable form, $\mathrm{A}\phi = \mathrm{b}$, to apply the Krylov method. Flux moments are defined as one explicit vector and stored in a one-dimensional array. The moments vector for group g is defined in Eq. (11), and the data storage of these vector, i.e., indexing, is shown in Fig. 1.

$$\phi_{\mathrm{g}} = \left(\phi_{00}^{\mathrm{g}}\ \phi_{1-1}^{\mathrm{g}}\ \phi_{10}^{\mathrm{g}}\ \phi_{11}^{\mathrm{g}} \cdots\ \phi_{\ell m}^{\mathrm{g}}\ \cdots\ \phi_{LL}^{\mathrm{g}}\right)^{T}. \tag{11}$$

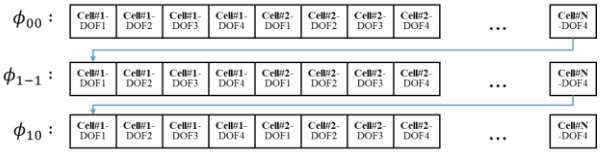

Fig. 1. Index Mapping of Moments into a 1-D Array

Linear algebra operations essential for the Krylov method were processed using the Eigen library [6]. The Eigen library was selected because it is written in C++, and the API design can be used intuitively and easily, and it can be used including only the header file without installation process.

Two Krylov methods Restarted GMRES(m) [7] and BiCGSTAB [8] were implemented in the STRAUM. As mentioned above, the Krylov methods were defined in a matrix-free manner using the basic algebra operations. Note that, Eigen provides the Krylov method for an explicitly defined matrix, and it can be used by wrapping the function corresponding to the matrix.

2.2. Multigroup Krylov Method for Upscattering Group

In the multigroup problem, the Gauss-Seidel (GS) method can be expressed as Eq. (12). This method has been used traditionally in transport calculations by constructing the source term from other groups for a specific group g and solving the within group equation.

$$\left(\mathbf{I} - \mathbf{DL}^{-1}\mathbf{MS}_{\mathrm{gg}}\right)\phi_{\mathrm{g}}^{(k+1)} = \mathbf{DL}^{-1}\mathbf{M}\left(\sum_{\mathrm{g}'=1}^{\mathrm{g}-1}\mathbf{S}_{\mathrm{gg}'}\phi_{\mathrm{g}'}^{(k+1)} + \sum_{\mathrm{g}'=\mathrm{g}+1}^{G}\mathbf{S}_{\mathrm{gg}'}\phi_{\mathrm{g}'}^{(k)} + q_{\mathrm{g}}^{ex}\right). \tag{12}$$

The GS method solves sequentially from high energy to low energy, and if there is upscattering, it should converge using extra-iterations additional to the single energy group sweeping. Since upscattering iteration takes a lot of time, the multigroup Krylov method as in Eq. (13) has been applied to solve the groups with upscattering with the following form [9] :

$$\left(\mathbf{I} - \mathbf{DL}^{-1}\mathbf{MS}_{GrpChunk}\right)\phi_{GrpChunk}^{(k+1)} = \mathbf{DL}^{-1}\mathbf{M}\left(\mathbf{S}_{incom}\phi_{incom}^{(k+1)} + q^{ex}\right). \tag{13}$$

Here, $GrpChunk$ means a cluster of groups, and $income$ means a cluster of all groups scattering into the energy groups in $GrpChunk$. In the STRAUM, the upper groups having only down-scattering are sequentially calculated with GS while the other groups with upscattering is treated as one chunk. For example, Fig. 2 illustrates the case where each group in the down scattering range is considered as a group chunk and the last three groups in the upscattering range are considered a single group chunk.

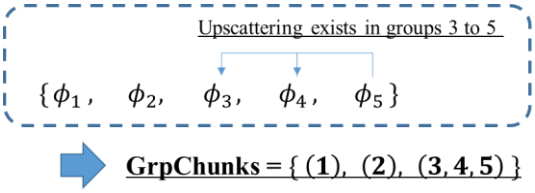

Fig. 2. Illustration of Group Chunks for Multigroup Krylov Method

4. Thread-based Parallelization for Sweep Operation

The sweep operation is performed based on the sorted order for a specific direction. As shown in Fig. 3, the relationship between visits in a specific direction can be expressed as a directed acyclic graph (DAG), and the sweeping order is derived through topological sorting for this graph.

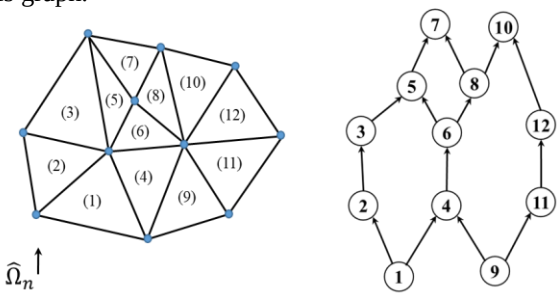

Fig. 3. A Spatial Mesh and Directed Acyclic Graph Aligned for a Specific Direction.

Generally, parallelism can be divided into three types: Message Passing Interface (MPI), thread, and Single Instruction, Multiple Data (SIMD) from a hardware point of view. Although domain decomposition using MPI and data level parallelism using SIMD should be finally applied to maximize computational efficiency, thread-based parallelization using shared memory was first implemented in this study.

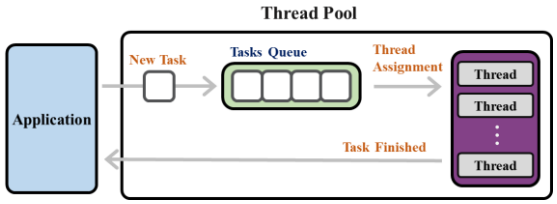

Fig. 4. Thread Pool Schematic Diagram

The method using openMP, which is widely known as shared memory parallelism, is specialized for a simple for loop, so it is not suitable for the sweep operation in which tasks have dependencies on each other. In this study, tasks were parallelized by creating a thread pool as shown in Fig. 4 using the Taskflow library [10].

5. Verifications

The verification and performance test for the Krylov methods and parallel computing implemented in the STRAUM were performed using following two shielding problems.

5.1. Kobayashi-like Problem

The first problem was developed by referring to Kobayashi's benchmark problem, which was designed for verifying transport code for the void region [11]. Fig. 5 shows the layout of the developed problem. The test problem is defined as a 30 cm x 50 cm x 30 cm hexahedron divided into 3 regions. The first region (region 1) has an isotropic and uniform neutron source of 1.0 neutrons/cm^{3}sec only for the first ten high energy groups. The second region (region 2) occupying most of the problem is filled with concrete. The third region (region 3) is defined as a 4 cm x 4 cm x 4 cm cube of SS304. The detailed problem parameters are described in Table 1.

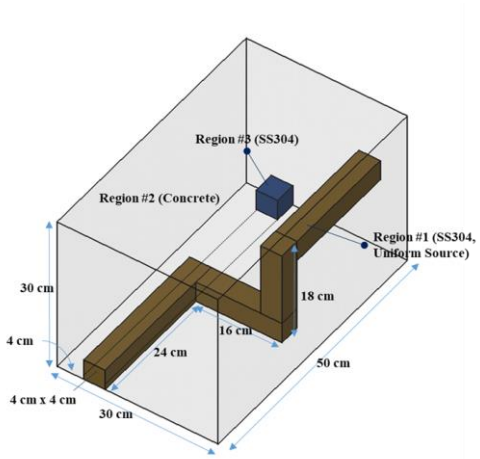

Fig. 5. Kobayashi-like Problem Layout

Table 1. Problem Parameters for Kobayashi-like Problem

# of Tetrahedrons	33,750
Discretization Method	LDEM-SCB(1)
Quadrature Set	2 polar x 2 azimuthal Gauss-Chebyshev set/octant
Cross Section	27 neutrons/19 gamma Coupled Cross Section
Anisotropic Order	P_3
Thermal Upscattering	Not considered

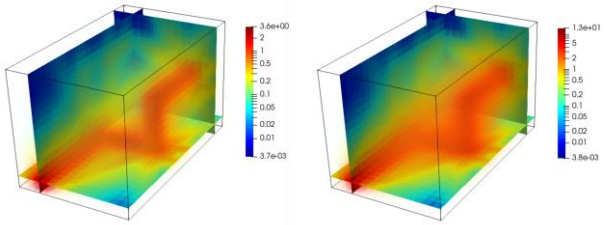

Fig. 6. Neutron Flux Distributions for Group 1 (left) and Group 10 (right)

Calculation time and the number of iterations were measured with different iteration methods. Fig. 6

compares the scalar fluxes for the groups 1 and 10 in the left and right sides, respectively. Fig. 7 compares the computing times for each energy group for several computing options implemented in STRAUM. As shown in the Fig. 7. Richardson with DSA and BiCGSTAB had the reduced elapsed time for convergence in comparison with the Richardson method (i.e., the original scattering source iteration). The GMRES(30) showed less performance improvement than expected, and it may be caused by the fact that the problem was easy and the solution converges quickly before the GMRES constructs the basis. Fig. 8 shows the change of the residuals for group 27, which takes the most computation time. As shown in the figure, DSA and GMRES(30) reduced residues smoothly, whereas BiCGSTAB shows some fluctuations. For this group, the Richardson with DSA showed the best performance in terms of both the number of iterations and computing time.

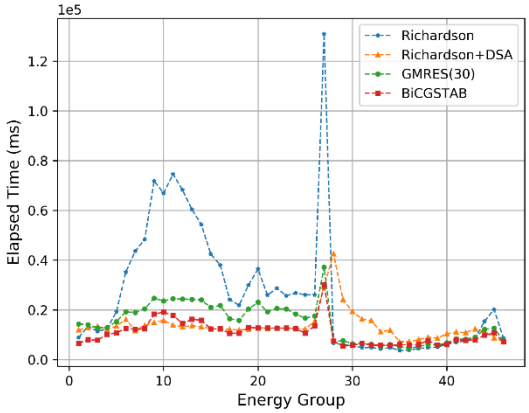

Fig 7. Groupwise Elapsed Time for Convergence

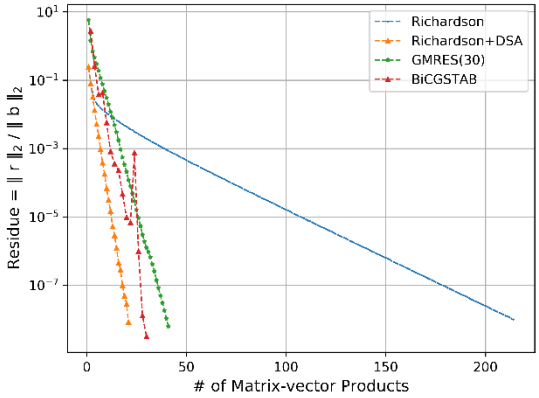

Fig 8. Comparison of Convergence Speed for Group-27

Table 2. Elapsed Time and Number of Sweep Operations for Group-27 Convergence

Iteration Method	# of Sweeps	Elapsed Time [ms]
Richardson	214	130,894
Richardson + DSA	21	29,404
GMRES(30)	41	37,163
BiCGSTAB	30	30,037

5.2. Reactor Pressure Vessel Problem

As the second problem, the Reactor Pressure Vessel (RPV) problem was designed to test the parallel computing and multigroup Krylov method in the STRAUM. Fig. 9 shows the tetrahedral mesh of the RPV problem. It has a total of 177,575 tetrahedral elements. A reflective condition was applied to three sides of the inner side of the core and vacuum condition was applied to the rest of the faces. The detailed problem parameters are described in Table 3.

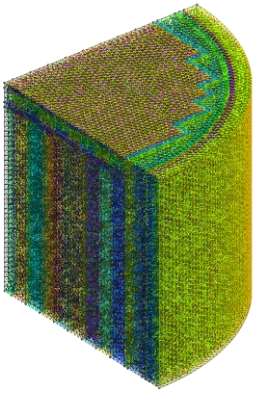

Fig. 9. Tetrahedron Mesh for RPV Problem Generated Using Gmsh

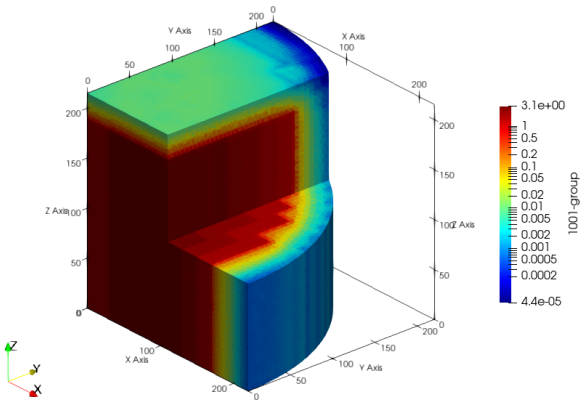

Fig. 10. Neutron Flux Distribution of Group-1 to RPV Problem

Table 3. Problem Parameters for RPV Problem

# of Tetrahedrons	177,575
Discretization Method	LDEM-SCB(1)
Quadrature Set (# of polar × azimuthal angles per octant)	2 × 2 & 3 × 3
Cross Section	47 neutrons with BUGLE-96 Structure
Anisotropic Order	P_3
Upscattering Groups	From 42 to 47

In order to measure the parallelism, the elapsed time of the 30 times consecutive sweeping operations in the problem was measured. The calculation was performed using an Intel i5-9600KF CPU (6 core, 6 threads, 3.70 GHz), and the measurement time is described in Tables 4 and 5. As can be seen from the tables, it was observed that the elapsed time was successfully reduced by increasing the number of threads. In addition, it was observed that parallel efficiency was not affected by the number of angles considered. But the parallel efficiency decreases up to 69% for 5 threads.

Table 4. Parallel Efficiency of Sweep Operation with 2x2 G-S Quadrature Set

# of Threads	Elapsed Time (x 30) [ms]	Parallel Efficiency
1	135,301	100 %
2	74,966	90 %
3	56,529	80 %
4	46,155	73 %
5	39,450	69 %

Table 5. Parallel Efficiency of Sweep Operation with 3x3 G-S Quadrature Set

# of Threads	Elapsed Time (x 30) [ms]	Parallel Efficiency
1	298,879	100 %
2	167,035	89 %
3	123,311	81 %
4	100,136	75 %
5	88,134	68 %

For thermal upscattering groups from 42 to 47, the conventional GS and multigroup Krylov method were applied and their convergences were inter-compared. In the GS method, the infinite norm of the scalar flux was set as the convergence checker, and in the multigroup Krylov method, the L_2 norm of the residue of the moments vector was set as the convergence checker. Since the infinite norm is a tighter convergence condition than L_2 norm, the convergence tolerance of the Krylov method was set to 10^{-9} for comparison. The detailed convergence criteria are listed in Table 6. As shown in Fig. 11, the GS method required a total of 549 iterations. On the contrary, the Krylov method converges quickly compared to the GS with a total of 33 iterations. The total computing time with BiCGSTAB and parallel computing using 4 threads for sweeping is about 0.9 hours for this problem.

Table 6. Comparison of Convergence Criteria for Thermal Upscattering Energy Groups

	GS Upscattering iteration	Multigroup Krylov
Convergence Criteria	$\max \left\| \frac{\phi_i^{(k+1)} - \phi_i^{(k)}}{\phi_i^{(k)}} \right\| < 10^{-3}$ for $i \in Cell$ with L_2 norm of residuals,	L_2 norm of residuals, $\frac{\|r\|_2}{\|b\|_2} < 10^{-9}$ for the GrpChunk Calc.
	$\frac{\|r\|_2}{\|b\|_2} < 10^{-5}$ for within-group Calc.	

Table 7. Computing Time for the RPV Problem

	GS Upscattering iteration	Multigroup Krylov
Downscatter Only Groups (from 1 to 41)	1932.751 sec (L_2 norm of residue < 10^{-9} with BiCGSTAB, and 4 Threads)	
Upscattering Groups (from 42 to 47)	26,648 sec	1346.7 sec (with BiCGSTAB, and 4 Threads)

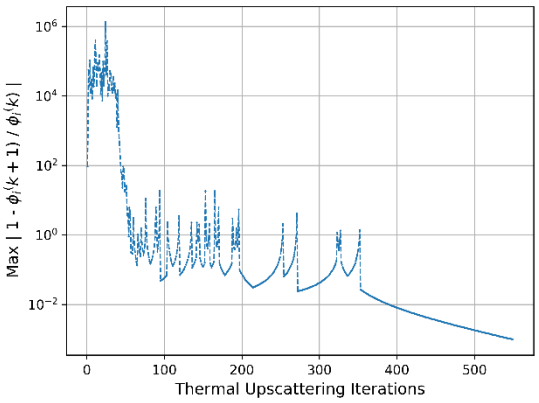

Fig. 11. Convergence of GS Thermal Upscattering

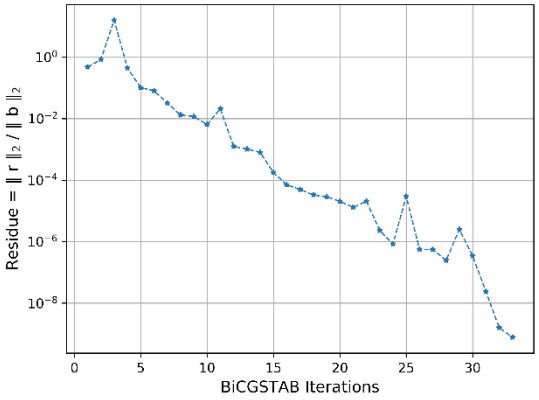

Fig. 12. Residuals of Group Chunk Calculation for Thermal Upscattering Groups

5. Conclusions

In this work, Krylov subspace methods and parallel computing for sweeping operation were applied in the STRAUM. Specifically, two Krylov methods, GMRES(m) and BiCGSTAB, were implemented. Additionally, The BiCGSTAB method has been expanded to multigroup problem to effectively solve the upscattering energy groups. From the application and analysis of these implementation, it is concluded that computation time was successfully reduced for the practical reactor shielding problems where the

computing time is about 0.9 hours in a single desktop PC having 4 threads. As future work, we plan to apply MPI-thread hybrid parallelization to maximize parallel efficiency and to employ some preconditioners for BiCGSTAB iterations.

Acknowledgments

This work was supported by the NRF (National Research Foundation of Korea) through Project No. NRF-2019M2D2A1A02057890.

REFERENCES

[1] S.G. Hong, J.W. Kim, and Y. Lee, "Development of MUST (Multi-group Unstructured Geometry S_N Transport) Code", Transaction of the Korean Nuclear Society Autumn Meeting, Gyeongju, Korea, 2009.
[2] S.G. Hong, "Two Subcell Balance Methods for Solving the Multigroup Discrete Ordinates Transport Equation with Tetrahedral Meshes", Nuclear Science and Engineering, Vol.173, p. 101-107, 2013.
[3] C. Geuzaine and J.-F. Remacle, Gmsh: a three-dimensional finite element mesh generator with built-in pre- and post-processing facilities. International Journal for Numerical Methods in Engineering, Vol.79, Issue 11, p. 1309-1331, 2009.
[4] Habib Muhammad, and S.G. Hong. "Diffusion synthetic acceleration with the fine mesh rebalance of the subcell balance method with tetrahedral meshes for SN transport calculations", Nuclear Engineering and Technology, Vol.52, Issue 3, p. 485-498, 2020.
[5] Azmy, Yousry, Sartori, Enrico, "Nuclear Computational Science: A Century in Review", Springer, pp. 1-84, 2010.
[6] Gaël Guennebaud, Benoıt Jacob and et al., "Eigen v3", http://eigen.tuxfamily.org/, 2010.
[7] Saad, Youcef and Schultz, Martin H, "GMRES: A Generalized Minimal Residual Algorithm for Solving Nonsymmetric Linear Systems", SIAM J. Sci. Stat. Comput., Vol.7, p. 856–869, 1986.
[8] Van der Vorst, H. A., "Bi-CGSTAB: A Fast and Smoothly Converging Variant of Bi-CG for the Solution of Nonsymmetric Linear Systems", SIAM J. Sci. and Stat. Comput., Vol.13(2), p. 631–644, 1992.
[9] R.N.Slaybaugh, et al., "Multigrid in Energy Preconditioner for Krylov Solvers", Journal of Computational Physics, Vol. 242, p. 405-419, 2013.
[10] T. -W. Huang, D. -L. Lin, C. -X. Lin and Y. Lin, "Taskflow: A Lightweight Parallel and Heterogeneous Task Graph Computing System," IEEE Transactions on Parallel and Distributed Systems, 2021.
[11] Keisuke Kobayashi, et al., "3-D Radiation Transport Benchmark Problems and Results for Simple Geometries with Void Regions", Progress in Nuclear Energy, Vol.39, p. 119-144, 2001.