STUDY OF METHOD OF CHARACTERISTICS FOR SOLUTION OF NEUTRON TRANSPORT EQUATION

By

TANAY MAZUMDAR

PHYS01201304035

Bhabha Atomic Research Centre, Mumbai

A thesis submitted to the

Board of Studies in Physical Sciences

In partial fulfillment of requirements

for the Degree of

DOCTOR OF PHILOSOPHY

of

HOMI BHABHA NATIONAL INSTITUTE

December, 2019

Homi Bhabha National Institute

Recommendations of the Viva Voce Committee

As members of the Viva Voce Committee, we certify that we have read the dissertation prepared by TANAY MAZUMDAR entitled "STUDY OF METHOD OF CHARACTERISTICS FOR SOLUTION OF NEUTRON TRANSPORT EQUATION" and recommend that it may be accepted as fulfilling the thesis requirement for the award of Degree of Doctor of Philosophy.

Amar Sinha	19/12/2019
Chairman - Prof. Amar Sinha	**Date:**
A. Gupta	19/12/2019
Guide / Convener - Prof. Anurag Gupta	**Date:**
	19-12-2019
Examiner – Prof. K P N Murthy	**Date:**
Member 1- Prof. K. Devan	**Date:**
Member 2- Dr. K. P. Singh	**Date:** 19/12/2019
S. B. DEGWEKER Invitee	

Final approval and acceptance of this thesis is contingent upon the candidate's submission of the final copies of the thesis to HBNI.

I/We hereby certify that I/we have read this thesis prepared under my/our direction and recommend that it may be accepted as fulfilling the thesis requirement.

Date: 19/12/2019

Place: Mumbai

A. Gupta

Prof. Anurag Gupta

Guide

STATEMENT BY AUTHOR

This dissertation has been submitted in partial fulfillment of requirements for an advanced degree at Homi Bhabha National Institute (HBNI) and is deposited in the Library to be made available to borrowers under rules of the HBNI.

Brief quotations from this dissertation are allowable without special permission, provided that accurate acknowledgement of source is made. Requests for permission for extended quotation from or reproduction of this manuscript in whole or in part may be granted by the Competent Authority of HBNI when in his or her judgment the proposed use of the material is in the interests of scholarship. In all other instances, however, permission must be obtained from the author.

Tanay Mazumdar

Tanay Mazumdar

DECLARATION

I, hereby declare that the investigation presented in the thesis has been carried out by me. The work is original and has not been submitted earlier as a whole or in part for a degree / diploma at this or any other Institution / University.

Tanay Mazumdar

Tanay Mazumdar

List of Publications arising from the thesis

Journal

1. "Solution of Neutron Transport Equation by Method of Characteristics", **Mazumdar, T.**, Degweker, S. B., *Annals of Nuclear Energy*, **2015**, 77, 522 – 535.

2. "Solution of the Neutron Transport Equation by the Method of Characteristics Using a Linear Representation of the Source within a Mesh", **Mazumdar, T.**, Degweker, S. B., *Annals of Nuclear Energy*, **2017**, 108, 132 –150.

3. "Application of Krylov Acceleration Technique in Method of Characteristics based Neutron Transport Code", **Mazumdar, T.**, Gupta, A., *Nuclear Science and Engineering*, **2018**, 192(2), 153-188.

Conferences

1. "Solution of neutron transport equation by Method of Characteristics", **Mazumdar, T.**, Degweker, S. B., *Proceedings of DAE-BRNS symposium on Advances in Reactor Physics: Simulation technique and analysis methodologies (ARP 2013)*, Mumbai, October 23-25, **2013**.

2. "Acceleration technique in method of characteristics based neutron transport solver", **Mazumdar, T.**, Gupta, A., Singh, T., Varde, P. V., *Proceedings of DAE-BRNS symposium on Advances in Reactor Physics: Nuclear Fission Reactors: Now & Beyond (ARP 2017)*, Mumbai, December 06-09, **2017**.

Tanay Mazumdar

Tanay Mazumdar

DEDICATIONS

To

Smt. Anjusree Mazumdar (Mother),

Shri Tamasnath Mazumdar (Father)

&

Dr. Sreemoyee Sarkar (Wife)

ACKNOWLEDGEMENTS

The journey of my Phd days provided me a magnificent opportunity to learn how to overcome odds both in professional and in personal life to accomplish the research work. This would not have been possible without unconditional support from many individuals. In this section, I would like to grab the opportunity to render my ardent gratitude to them.

To begin with, I would like to express my deep and sincere gratitude to my former thesis supervisor Dr. S. B. Degweker, Head, MP&RThS (now retired) for his invaluable guidance, fruitful discussion, deep insight and constant encouragement in pursuing the work. I am thankful to him for allowing technical discussion over telephone due to my professional constraint. I am extremely indebted to my present supervisor Dr. Anurag Gupta, Head, RPAS, RRSD for considering me as his student after Dr. Degweker's retirement. I am grateful to him for introducing me advanced acceleration techniques.

I would like to thank the members of the doctoral committee - Prof. Amar Sinha (Chairman), Prof. K. Devan (Member) and Dr. K. P. Singh (Member) for their critical review and suggestions during the progress review, pre-synopsis seminar and in writing the synopsis and thesis.

I sincerely thank Dr. Kanchhi Singh, Head, RRSD (now retired), who has granted me permission to register in Phd programme of HBNI. Thanks are due to Dr. P. V. Varde (Associate Director, RG), Dr. Tej Singh (Head, RPNES), Dr. Y. S. Rana, Smt. Archana Sharma, Shri Jainendra Kumar, Shri Paritosh Pandey, Shri Avijit Das, Shri Abhishek Mishra and Shri Shantanab Banerjee for their support and encouragement.

The list will not be complete if I don't acknowledge my teachers of school, college and IIT without whom it would not have been possible to achieve whatever I have done so far. I would like to mention about late Prof. M. C. Valsakumar, who had not only mentored me to improve my skill in computational physics during OCES training in IGCAR, Kalpakkam, he had influenced me a lot with his impeccable ability to analyze scientific problem from first principle.

My sincere gratitude goes to my parents and elder brother for extending their continuous support and inspiration from the very beginning of my childhood. Ever since I joined BARC, my father used to remind me about registering in Phd and even after the registration, he was very much keen to know the progress of my work. I am grateful to my parents in laws as well for making my days joyful by preparing delicious dishes whenever we meet.

Last but not the least, my soulmate, my wife Dr. Sreemoyee Sarkar – what should I say about her? I feel short of words in expressing my appreciation towards her. Being an academician, she understands my psychology very well and helped me a lot with her own experience in this bumpy ride of Phd. I am thankful to her for unconditional love, support and encouragement.

CONTENTS

SYNOPSIS

Nuclear energy, since its inception in Chicago Pile experiment in 1942, has evolved as one of the alternatives of fossil fuel and leaders of low carbon power generation in today's world. This "clean" and "green" form of energy is extracted in safe manner from the controlled chain of nuclear fission reactions inside nuclear reactors. The process of reactor design involves finding the best possible configuration of nuclear material so as to produce the power safely and economically. This requires an accurate calculation of multiplication factor, power distribution, worth of reactivity devices, coefficient of reactivity etc.. A nuclear reactor is a heterogeneous ensemble of fuel, moderator, structural material, control absorbers, reflector etc. where each material has different interaction property with the neutron. Moreover, the interaction probability of neutrons with a single material is not a smooth function of neutron energy. Therefore, different approximate techniques are used to model a reactor, which lead to uncertainty in the design calculations. With better understanding of neutronic behavior and advanced computational resources available today, more exact modeling is being attempted to enhance the accuracy of the calculations.

In principle, the neutron transport equation, which is the analogue of Boltzmann equation of kinetic theory of gases, should be solved directly for the heterogeneous reactor core to calculate the parameters of interest. The traditional deterministic reactor physics calculations, however, follow a two-step process, mainly because of limitations of computational resources. The first step is the lattice calculation, in which two dimensional neutron transport equation is solved in multigroup framework (69 or 172

neutron energy groups) within a small and representative region of the reactor, referred to as the “lattice cell” (typically a fuel assembly), taking into account of all the details of the heterogeneous geometry. This provides fine mesh and fine energy group neutron flux distribution in the lattice cell, which is used to calculate few energy group cross sections of a homogeneous material, which replaces all the materials of the lattice cell. In the second step, these homogenized cross sections are used to solve the three dimensional, few energy group neutron diffusion equation for the full reactor core. Due to the phenomenal advancement of computing power in recent times, it has now become possible to obtain the solution of the transport equation directly for the entire core. Several such developments, both in 2D and 3D, have already been reported in literature [11][35]. 3D whole core transport calculation requires enormous computation effort. Hence, fusion methods [15] are being attempted, taking into account of the reactor core configuration, which is, in general, less heterogeneous in the axial direction than in the radial direction. It is the Method of Characteristics or MOC [17], which is common in all these references of full core transport work. MOC considers finite number of discrete directions similar to discrete ordinate or DSn method. However, both the methods calculate mesh average flux in different way. In DSn method, starting from a given mesh, fluxes are calculated in neighbouring meshes using principle of directional evaluation and imposing neutron balance over each mesh. This task is difficult for mesh of arbitrary shape. In MOC, a set of parallel characteristic lines or rays are traced over the problem domain (lattice or core) for each discrete direction and the intercepts of rays in different meshes are calculated. The neutron transport equation is solved analytically along these rays to obtain the mesh averaged ray angular flux and outgoing mesh edge ray angular flux. These mesh averaged ray angular fluxes are then summed

for all those rays passing through a mesh in a particular direction to calculate the average angular flux of the mesh for the direction. This formalism does not depend on the mesh geometry. Hence, MOC is able to treat complex geometries commonly encountered in the reactor core, which brings its applicability at par with the Monte Carlo method. Other advantages of MOC include (i) detailed flux and power distribution calculation over the region of interest, (ii) treatment of anisotropic scattering and (iii) ability to obtain solutions in neutronically large sized domains. Commonly used methods like collision probability or Monte Carlo method does not offer one or more of the abovementioned advantages. For these reasons the MOC is gaining popularity for not only lattice calculations but also for the whole core calculations without homogenization. Computer codes like DeCART [35], CRX [18], CASMO-4 [19], DRAGON [21], MOCUM [22], OpenMOC [23] etc. have so far been developed based on MOC. However, in MOC, solution of neutron transport equation is obtained by sweeping all the rays repetitively, which is time consuming and defeats the advantages of MOC. In order to circumvent the issue, it is required to adopt efficient solution technique, which will offer better convergence of MOC solution.

In the present thesis, we have tried to explore various efficient solution techniques of neutron transport equation within the framework of MOC.

1. We describe the ***development and verification of a 2D MOC based neutron transport code, which uses a mesh division technique based on Delaunay triangulation in conjunction with Bowyer-Watson algorithm*** and concepts of coordinate geometry for ray tracing. (reference a.[1])
2. ***Unlike most of the MOC codes, flat representation of neutron source within a mesh is extended to linear representation, which allows the method to use larger***

triangular mesh for solving the transport equation. In this connection, we have developed a method to correct the distortion in representing cylindrical geometry by the large triangular meshes. Finally, we have shown results of some benchmark problems to bring out the advantage of the linear source expansion method. (reference a.[2])

3. In MOC, the neutron transport equation is solved by repetitive sweeping of all the rays traced through the problem domain, which is time consuming. Therefore, ***MOC solution is accelerated using an advanced approach of Krylov subspace iteration method wherein explicit construction of matrix for the matrix-vector multiplication is not required.*** Advantage of this acceleration technique is demonstrated by solving several benchmark problems. (reference a.[3])

In the thesis, the work is presented in six chapters as elaborated below.

1. Introduction

The neutron transport equation, which forms the basis for the presented thesis work, is briefly introduced in Chapter 1, explaining its importance in the study of reactor core. Being the topic of our current research, there is special mention about MOC citing its advantages over the existing solution techniques of neutron transport equation. Many researchers have contributed to this field of MOC, which is acknowledged and the way the presented thesis work gives a value addition to this field, is clearly brought out. The work presented in the thesis is outlined at the end of the chapter.

2. Neutron Transport Equation and its solution

In order to understand the behavior of nuclear reactor, it is very essential to know the distribution of neutrons as a function of space, angle, energy and time. This can be found by solving the neutron transport equation. Chapter 2 discusses the assumptions

of neutron transport theory made in the context of nuclear reactor, derivation of integro-differential and integral forms of the neutron transport equation and the conditions (initial and boundary) required to solve the equation. It is impossible to obtain the exact solution of neutron transport equation unless we consider very simple cases such as isotropic scattering in a uniform, infinite medium containing plane or point source etc.. For all other cases, approximate solution techniques are used to solve the transport equation. In Chapter 2, few such approximate techniques, viz. Discrete ordinate, P_N, Multigroup, Collision probability and MOC are discussed. Numerous computer codes have been developed based on these approximate techniques to solve the transport equation in quick and efficient manner. Correctness of these codes have been established by solving a number of benchmark problems whose results are already known either by experiments or by theoretical calculations performed by some other benchmarked codes. Chapter 2 lists a number of such problems for various geometry and boundary conditions.

3. Solution of Neutron Transport Equation by Method of Characteristics

In Chapter 3, we describe the development of a 2D computer code based on MOC to solve the neutron transport equation for mainly assembly level lattice calculation with reflective and periodic boundary conditions and to some extent core level calculation with vacuum boundary condition. A prerequisite for solving the transport equation by the MOC is the division of the problem domain into small meshes. The traditional way of mesh division is to divide the assembly into square or hexagonal meshes conforming to the lattice structure and further divide these meshes into radial and azimuthal zones. While it gives the user the freedom to choose the manner of division, it makes the preparation of the input more cumbersome. In this respect, triangular meshes are found

to be useful and their applications are well appreciated for complicated unstructured geometry. Another prerequisite of MOC solution is ray tracing, where sufficiently large number of characteristic lines or rays, along which the transport equation is solved, are constructed in various discrete directions and their intersection with the meshes are found out. Usually the reflective boundary is discretized into small segments and the angular fluxes along a segment are either assumed to be uniform or calculated using linear interpolation scheme.

In the MOC code, which we have developed, the division of the geometry into meshes is incorporated through an automatic triangulation procedure based on the Delaunay triangulation technique along with the Bowyer-Watson algorithm, which is unique in a sense that the technique is applicable for any geometry and not many researchers have reported such application in the field of neutron transport theory. Presently, the solution domain can be any geometry consisting of a combination of circles, rectangles and hexagons subject to the outer boundary being square or hexagonal like CANDU, PWR, BWR etc.. Coordinate geometry based approach is adopted for ray tracing. A number of benchmark problems, described in Chapter 2, are analyzed to demonstrate the capability and validity of the code.

4. Linear Representation of Source in Method of Characteristics

In most of the MOC based transport codes, flat neutron flux or neutron source assumption is considered [18][21][22][23]. Under this assumption, the meshes must be small enough to achieve acceptable accuracy in results. Mesh size depends on mean free path of neutron in the respective material. This leads to the use of large number of meshes and consequently large computation time. A better and efficient way of calculation is to expand the neutron flux or source within a mesh in terms of polynomial

basis functions of spatial variables. This permits the use of larger mesh size so that fewer meshes are required. Inclusion of higher order basis functions in the expansion of flux enhances the allowable mesh size at the cost of evaluating additional expansion coefficients. Hence, the polynomial expansion is generally restricted up to linear terms, which, according to the literature [37], provides a balanced optimization between speed and memory requirement as compared to higher order expansions.

In Chapter 4, we propose a formalism in which the source within a triangular mesh is expanded up to linear terms. The expansion coefficients are determined by setting up equations for the average source and its moments in a mesh. A major problem with this approach is the inaccurate representation of circular geometries with coarse triangular meshes. Hence, a method is developed to modify the shape of the mesh for making the representation conformal. A number of benchmark problems are analyzed to emphasize the advantage of the source expansion method and the need to correct the shape of the coarse meshes used in the triangulation of the circular geometry.

5. Krylov Acceleration Technique in Method of Characteristics

In Chapter 5, we propose, to the best of our knowledge, the first ever matrix free approach in Krylov accelerated MOC wherein explicit construction of matrix for the matrix-vector multiplication is not required. In MOC, neutron transport equation is solved by sweeping all the rays repetitively, which is time consuming and defeats the advantages of MOC. Therefore, acceleration techniques have been developed for MOC. Krylov subspace methods [53] are useful in accelerating the solution of neutron transport equation. However, these methods, being matrix based, pose another difficulty in obtaining the solution quickly. For real problems, size of the matrix is huge, which motivates us to find out the effect of matrix “A” on vector “ψ”, instead of forming the

matrix explicitly. First, we have attempted the matrix based approach to understand how the matrix "A" operates on the vector "ψ". It has been found that the matrix is huge in size and difficult to handle even for simple problems. Using the angular flux continuity condition, the number of unknowns as well as the matrix size is reduced and the conventional MOC solution is converted into a practically solvable matrix equation $A\psi=q$ to conform to the framework of Krylov subspace methods. After obtaining the matrix equation, we work out the recipe to induce the similar effect of matrix equation without forming the matrix explicitly. This matrix free approach is used in combination with flat as well as linear source assumption to solve a number of benchmark problems. Results show significant improvement in terms of faster convergence of solution over the conventional iteration without compromising the accuracy.

6. Conclusion and Future Scope of Work

At present, the MOC based neutron transport code, developed during the course of this thesis work, can be used for mainly assembly level lattice calculation and to some extent core level calculation. Two types of flux representation, viz. flat and linear within a mesh are available with the code. The code is equipped with Krylov based matrix-free advanced acceleration technique, which offers faster convergence of solution. With this present status of our work, outline of the future work is presented in Chapter 6. Our next plan is to couple the developed MOC code to a fine-group cross section library and to include treatment of anisotropic scattering and burn-up to convert it into a complete lattice code. It has also been planned to apply acceleration technique for outer iteration to reduce the computation time. Further reduction in computation time is possible by simulating half or quarter core and permitting different boundary conditions at various surfaces instead of simulating the whole core and applying the same boundary condition

at each of the surfaces. Though efforts were made in this direction for one or two problems, there are scopes to implement this capability in a more general way. It is possible to solve neutron transport equation for full core calculation without doing any homogenization. For doing so, we would like to extend our transport code from lattice level to full core level using parallel computers, which will help us obtaining the solution in reasonable time for whole core.

References:

[1] Lee, G. S., Cho, N. Z., Hong, S. G., 2000. Acceleration and parallelization of the method of characteristics for lattice and whole-core heterogeneous calculations. Proc. of PHYSOR 2000, Session II-C, Pittsburgh, USA, May 7-11.

[2] Cho, J. Y., Joo, H. G., Kim, K. S., 2002. Cell based CMFD formulation for acceleration of whole-core method of characteristics calculations. Journal of the Korean Nuclear Society, 34(3), pp.250-258.

[3] Cho, N. Z., Lee, G. S., Park, C.J., 2002. A fusion technique of 2-D/1-D methods for three-dimensional whole-core transport calculations. Proc. of the Korean Nuclear Society Spring Meeting, Kwangju.

[4] Askew, J. R., 1972. A characteristics formulation of the neutron transport equation in complicated geometries. Report AEEW-M, 1108.

[5] Hong, S. G., Cho, N. Z., 1998. CRX: a code for rectangular and hexagonal lattices based on the method of characteristics. Ann. Nucl. Energy 25 (8), 547–565.

[6] Smith, K., Rhodes, J., 2000. CASMO-4 characteristics methods for two-dimensional PWR and BWR core calculations, ANS/ENS International Winter Meeting and Embedded Topical Meetings, Washington D.C., November.

[7] Marleau, G., Hébert, A., Roy, R., 2008. A user guide for DRAGON 3.06. Report IGE-174 Rev, 7.

[8] Yang, X., Satvat, N., 2012. MOCUM: A two-dimensional method of characteristics code based on constructive solid geometry and unstructured meshing for general geometries. Annals of Nuclear Energy, 46, 20-28.

[9] Boyd, W., Shaner, S., Li, L., Forget, B., Smith, K., 2014. The OpenMOC method of characteristics neutral particle transport code. Annals of Nuclear Energy, 68, pp.43-52.

[10] Masiello, E., Clemente, R., Santandrea, S., 2009, May. High Order Method of Characteristics for 2-D Unstructured Meshes. In Proc. Int. Conf. on Mathematics, Computational Methods & Reactor Physics (M&C 2009) (pp. 3-7).

[11] Saad, Y., 1996. Iterative Methods for Sparse Linear System. PWS, Boston, MA.

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

Introduction

In today's world, nuclear energy is being considered as one of the alternatives of fossil fuel for meeting our energy needs in a better and cleaner way. Nuclear energy is extracted in safe manner from the controlled chain of nuclear fission reactions inside nuclear reactors. A nuclear reactor is a heterogeneous ensemble of fuel, moderator, structural material, control absorbers, reflector etc. in a sense that every material has different interaction property with the neutron. In principle, neutron transport equation, which is the analogue of Boltzmann equation of kinetic theory of gases, should be solved for such a heterogeneous system to calculate the parameters of interest like multiplication factor, neutron flux, power distribution etc.. However, traditional deterministic reactor physics calculations follow a two-step process, mainly because of limitations of computational resources. The first step is the lattice calculation, in which one or two dimensional neutron transport equation is solved in multigroup framework (69 or 172 neutron energy groups in WIMS) within a small and representative region of the reactor. This region is known as "lattice cell" (typically a fuel assembly), which contains all the details of the heterogeneous geometry. This provides fine mesh and fine energy group neutron flux distribution in the lattice cell, which is used to calculate few energy group cross sections of a homogeneous material, which replaces all the materials of the lattice cell. In the second step, these homogenized cross sections are used to solve the three dimensional, few energy group neutron diffusion equation for the entire core.

In earlier days, the representative region used to be a ‘pin cell’ consisting of a single fuel rod and associated moderator with reflective boundary condition. The outer boundary was often converted from square or hexagonal to circular by Wigner-Seitz approximation to further simplify the geometry to that of a one dimensional problem. Due to the presence of heterogeneities such as control rods, water gaps or water rods, the lattice is not exactly periodic. Various corrections such as the use of a “white boundary” instead of reflective one and the use of a “supercell”, which includes a homogenized mixture of fuel and moderator to represent the spectrum external to the lattice, had to be included to get satisfactory results. An intermediate assembly calculation based on diffusion theory, using the homogenized and group condensed cross sections, was performed to obtain the homogenized cross sections for the fuel assembly. Examples of the use of such methods are MURLI [1][2], EXCEL [3] and the LEOPARD-PDQ combination [4]. Since seventies and early eighties, computer codes were developed for performing lattice calculations at the fuel assembly level directly. The integral transport theory, using either the collision probability, the interface current or a combination of these methods [5][6][7][8][9], was popular for treating the complex geometries involved. The method has the added advantage that it can be formulated in terms of only the scalar flux, since the scattering anisotropy plays only a minor role and can be corrected by the use of transport cross sections.

Due to the phenomenal advancement of computing power in recent time, we can now think of obtaining the direct solution of the transport equation for full reactor core i.e. without the need for a separate lattice calculation for the homogenized cross sections. Several such developments, both in 2D and 3D, have already been reported in literature [10][11][12][13][14]. It is easy to understand that 3D whole core transport calculation

requires enormous computation effort. Hence, fusion methods [15][16] are attempted, taking into the account of the reactor core design which is, in general, less heterogeneous in axial direction than in radial direction. It is the Method of Characteristics or MOC [17], which is common in all these references of full core transport work. MOC considers finite number of discrete directions similar to discrete ordinate or DSn method. However, both the methods calculate mesh average flux in different ways. In DSn method, starting from a given mesh, fluxes are calculated in neighbouring meshes using principle of directional evaluation and imposing neutron balance over each mesh. This task is difficult for mesh of arbitrary shape. In MOC, a set of parallel characteristic lines or rays are traced over the problem domain (lattice or core) for each discrete direction and the intercepts of rays in different meshes are calculated. Neutron transport equation is solved analytically along these rays to obtain mesh average ray angular flux and outgoing mesh edge ray angular flux. These mesh average ray angular fluxes are then summed for all the rays passing through a mesh in a discrete direction to calculate average angular flux of the mesh. This formalism does not depend on the mesh geometry. Hence, MOC is able to treat complex geometries commonly encountered in reactor core, which brings its applicability at par with Monte Carlo method. Other advantages of MOC include (i) detailed flux and power distribution calculation over the region of interest, (ii) treatment of anisotropic scattering and (iii) ability to obtain solution in neutronically large sized domains. Commonly used methods like collision probability or Monte Carlo method does not offer one or more of the abovementioned advantages. For these reasons the MOC is gaining popularity for not only lattice calculations but also for whole core calculations without homogenization. Computer codes like CRX [18], CASMO-4 [19], AutoMOC

[20], DRAGON [21], MOCUM [22], OpenMOC [23], BOXER [24] etc. have so far been developed based on MOC.

A prerequisite for solving the neutron transport equation by the MOC is the division of the problem domain into small meshes. The traditional way of mesh division is to divide the assembly into square [25][26][27] or hexagonal meshes [28][29] conforming to the lattice structure. Further sub division of these meshes into radial and azimuthal zones is also attempted [9][24]. While it gives the user the freedom to choose the manner of division, it makes the preparation of the input more cumbersome. Triangular meshes are found useful for complicated unstructured geometry, likely to be encountered in advanced reactors, and therefore, used in MOC codes like TPTRI [30], MOCUM etc.. With an aim to simulate complex geometry, AEGIS [31] and ANEMONA [32] have been developed based on R-function solid modeler, which has high flexibility and the power of the combinatorial geometry method. The powerful graphics capability of AutoCAD, a well-known computer aided design software package, is exploited by AutoMOC for efficient handling of unstructured geometry. Another prerequisite of MOC solution is ray tracing, where sufficiently large number of characteristic lines or rays are constructed in various discrete directions. Usually ray tracing algorithm requires each ray, after multiple reflection, should end at its starting point on reflective boundary [33]. However, this constraint restricts the distribution of rays and shape of external boundary. In order to remove the constraint, the reflective boundary is discretized into small segments and the angular fluxes along a segment are either assumed to be uniform [18][32] or calculated using linear interpolation scheme [34]. In AutoMOC, the VBA (Visual Basic for Applications) language is used to customize AutoCAD for carrying out the ray tracing procedure irrespective of

geometric shape. ANEMONA offers R-function based general geometry ray tracing. For whole core calculation, enormous computational effort is required to store all the ray segments intercepted by the mesh boundary. Therefore, DeCART [35] and CHAPLET [10] use cell based modular ray tracing where ray segments are generated only for few types of cells and each ray segment of a cell is linked to a corresponding ray segment of the neighboring cell by adjusting ray spacing and azimuthal angle. In the MOC code we have developed and reported [36], the division of the geometry into meshes is through an automatic triangulation procedure based on the Delaunay triangulation technique along with the Bowyer-Watson algorithm, which is unique in a sense that the technique is applicable for any geometry and not many researchers [22] have reported such application in the field of neutron transport theory. Presently the solution domain can be any geometry consisting of a combination of circles, rectangles and hexagons subject to the outer boundary being square or hexagonal like CANDU, PWR, BWR etc.. A very basic approach, based on elementary coordinate geometry, is adopted for ray tracing.

Many MOC based transport codes consider flat neutron flux or neutron source (i.e. fission and scattering source). Under this assumption, the meshes must be kept small enough to achieve acceptable accuracy in results. Extent of smallness depends on the mean free path of neutron in the respective material. This leads to use of large number of meshes and consequently large computation time. A better and efficient way is to employ a higher order representation of the flux or source within a mesh. In other words, the neutron flux or source within a mesh is expanded in terms of polynomial basis functions of spatial variables, multiplied by suitable expansion coefficients. This permits the use of larger mesh size. Inclusion of higher order basis functions in the

expansion of flux increases the allowable mesh size at the cost of evaluating additional expansion coefficients. Hence, the polynomial expansion is generally restricted up to linear terms, which, according to literature [37][38], provides a balanced optimization between speed and memory requirement as compared to higher order expansions. With an aim to perform core calculation in reasonable time with our MOC based transport code, we propose a formalism in which the source in a triangular mesh is expanded in terms of polynomials, retaining up to the linear terms [39]. The expansion coefficients are determined by setting up equations for the average source and its moments in a mesh. A major problem with this approach, for geometries involving curved surfaces (e.g. cylindrical rods), is that a coarse mesh division by triangulation does not represent the circular boundary correctly. To overcome this, a method has been developed to use the coarse meshes in circular geometry by modifying the shape of the meshes so that the geometry representation is exact.

In MOC, the neutron transport equation is solved along a set of parallel characteristic lines or rays in a number of discrete directions. Sweeping on all these rays again and again has to be carried out in order to obtain results with acceptable accuracy. This may be time consuming and may defeat the advantages of MOC. Thus, for larger practical problems, acceleration techniques need to be incorporated for MOC simulations. A common method, the Coarse Mesh Finite Difference (CMFD), used by many transport codes [40][41][42], converts the fine mesh heterogeneous problem into a coarse mesh homogeneous one, which is easier to solve with few energy groups, diffusion like finite difference scheme and the solution provides much faster convergence of fission and scattering source distribution to MOC kernel. This method has also been extended for general geometry in Generalized Coarse Mesh Rebalancing

(GCMR) [43] and Generalized Coarse Mesh Finite Difference (GCMFD) [44]. CACTUS, the cyclic characteristics solver of WIMS, implements energy group rebalancing algorithm, which homogenizes the original problem using conventional flux and volume weighting method and uses the homogeneous solution, obtained at the end of each iteration, to rescale the characteristics flux [45]. In DRAGON, Track Merging Technique (TMT) and Self Collision Rebalancing (SCR) techniques are prescribed for accelerating the characteristic solution in large domain [46]. Transport Synthetic Acceleration (TSA) scheme has been used by Zika and Adams [47] in their MOC solver to accelerate the high order approximated problem by solving a lower order approximated problem. Sanchez and Chetaine introduced an Asymptotic Synthetic Acceleration (ASA) scheme in their MOC code with unstructured meshes [48], which was generalized later by incorporating a general DP_N angular approximation [49]. Angular Dependent Rebalance (ADR) iteration method has been applied in equilateral triangular meshes in the step characteristic based transport code [50].

Applications of Krylov subspace iteration methods have been found to be useful in many fields including neutron transport theory [51][52] since its inception for solving linear systems with matrices [53]. In order to take advantage of the methods, it is attempted to convert the problem into a matrix equation, which is then solved by different variants of Krylov subspace method, namely Generalized Minimal RESidual (GMRES) method [54][55][56], Lanczos' iteration [49] etc.. To apply the method in our MOC setup [57], we formulated the problem by constructing a coefficient matrix, which is non-symmetric in nature and therefore, solved by a Krylov variant, called BiConjugate Gradient STABilized (BiCGSTAB) method. However, the matrix-based approaches suffer from speed and storage issues in case of large problem despite

utilizing the sparsity of the matrix. In order to circumvent these issues, various remedies like partitioning the problem domain into several sub-domains in domain decomposition method [55], finding an appropriate preconditioner of GMRES to transform the original linear system into an easier one, keeping the solution intact [56] etc. are in effect. In this connection, we propose first time, in our knowledge, a matrix free approach in Krylov accelerated MOC wherein explicit construction of matrix for the matrix-vector multiplication is not needed [57].

In the present thesis work, we have tried to explore various efficient solution techniques of neutron transport equation within the framework of MOC. The thesis is organized as follows: Chapter 2 describes the fundamental neutron transport theory and its underlying assumptions, which is the backbone of the present study. This is followed by a mention of few existing solution techniques of neutron transport equation. In order to test the performance of these solution techniques, benchmark problems are required. A list of such problems with details of input parameters like geometry and material description, cross section of neutron interaction with the material etc. are given in Chapter 2. Chapter 3 describes the development and verification of a MOC based neutron transport code, which uses Delaunay triangulation technique in conjunction with Bowyer-Watson algorithm for mesh division and concepts of coordinate geometry for ray tracing. In Chapter 4, the mathematical formulation of the linear representation of the source within a mesh and the geometry correction needed for the expansion in the presence of cylindrical bodies are described. This chapter further includes analysis of some benchmark problems to emphasize the advantage of the source expansion method. In Chapter 5, acceleration of the MOC solution using Krylov subspace based iteration method is discussed. Advantage of the acceleration technique using a novel

matrix-free formalism, proposed in this chapter, is demonstrated by solving few benchmark problems. In the final chapter, we present our conclusions and future scope of the work.

CHAPTER 2

Neutron Transport Equation and its solution

2.1 Introduction

In order to understand the behavior of a nuclear reactor, it is essential to know the distribution of neutrons as a function of space, angle, energy and time. This can be found out by solving the linear form of Boltzmann neutron transport equation. The sections in the present chapter include the discussion on the assumptions of neutron transport theory made in the context of nuclear reactor, derivation of integro-differential and integral form of the neutron transport equation and the conditions (initial and boundary) required to solve the equation. It is very difficult to obtain the exact solution of the neutron transport equation. There are two approaches, viz. deterministic and stochastic, to obtain the approximate solution. In present chapter, we restrict to deterministic approach in which the independent variables - space, angle, energy and time, are discretized to solve the transport equation. Few deterministic methods viz. Discrete ordinate, P_N, Multigroup, Collision probability and MOC are discussed here. Numerous computer codes have been developed based on these approximate techniques to solve the transport equation in quick and efficient manner. Correctness of these codes have been established by solving a number of benchmark problems whose results are already known either by experiments or by theoretical calculations performed by some other benchmarked codes. Number of such problems are provided in this chapter for various geometry and boundary conditions.

2.2 Assumptions of neutron transport theory

Following assumptions are made before deriving the neutron transport equation [58][59]:

(i) *Neutrons are assumed to be point particles*. According to the wave-particle duality in quantum mechanics, the De Broglie wavelength (λ) is associated with neutron ($\lambda \sim h/p$; h is Planck constant and p is momentum of neutron). The wavelength for neutrons in reactors is typically found to be small (~ 0.1 A^o for 0.01 eV neutron) as compared to the distance travelled by the neutrons between successive collisions (~ few cm). Therefore, position and momentum of the neutrons can be determined simultaneously with adequate precision. This classical treatment does not hold for neutrons with very low energy. Since, such neutrons are very few in number, this violation does not cause any significant alteration in neutron transport theory.

(ii) *Neutron-neutron interactions are ignored*. Neutron density in reactors is much lower as compared to target nucleus density in the rector materials. For example, if thermal neutron flux (Φ) is about ~ 10^{15} n/cm^2/sec in a typical reactor and v is the average velocity of thermal neutron ~ 10^5 cm/sec, then the corresponding neutron density Φ/v will be ~ 10^{10} n/cm^3 whereas density of nucleus or atom in fuel, clad etc. is $\rho N_A/A$ (ρ is density of material, A is mass number and N_A is Avogadro number) which is of the order of 10^{22}/cm^3. Hence, neutron-nucleus interaction is much more favored than the neutron-neutron interaction. This assumption makes the neutron transport equation linear. It is worthy to mention here that in kinetics theory of gases, Boltzmann transport equation is non-linear due to the interaction between gas molecules.

(iii) *Neutrons travel in straight lines in between successive collisions*. Since, neutron does not carry any charge, long range electric or magnetic force does not alter its straight line trajectory. Deflection takes place only when short range force acts on the neutron during neutron-nucleus interaction.

(iv) *Collisions are regarded instantaneous*. Neutrons are emitted immediately (~ 10^{-14} s) after every collision except in the case of nuclear fission where a small fraction of neutrons, called delayed neutrons, are emitted after their precursors decay (~ msec to minute). Hence, the assumption of instantaneous collision will not result the correct time dependent behavior of neutron kinetics problem. However, the neutron transport equation with no delayed neutrons is found suitable for steady state problems either with fixed neutron sources or the reactor criticality calculations in which one is interested only in the critical state and flux distribution in the reactor and not the details of the time-dependent behavior.

(v) *Only the expected or mean value of the neutron density distribution is considered*. As a general rule in nuclear reactors, fluctuations about the mean value of neutron density is small as compared to the mean value. There are some practical situations e.g. reactor startup in which the departure from the average behavior is relatively large and cannot be overlooked. For dealing with such behavior, stochastic theories of neutron transport and multiplication have been developed in which the probabilities of various exceptional events are considered along with more normal situations.

(vi) *The material properties are assumed to be isotropic*. However, if the energy of neutron is very low and hence, its wavelength becomes long, then the diffraction

pattern arisen from material having crystal structure preferably aligned along certain directions, will be anisotropic.

(vii) *The properties of nuclei and the compositions of materials under consideration are assumed to be time-independent*. It should be noted that the thermal energy produced in nuclear fission rises the temperature of the fuel material and therefore, nuclear density of the fuel changes. Moreover, this temperature rise causes Doppler broadening of cross section (specially at resonance peak), which changes the interaction cross section of nuclei with neutron. Apart from this, new isotopes are also formed in fission, (n, γ), (n, α) and various other reactions, which modify the neutronic characteristics of material. Since, the time scale for these changes to take place is very long as compared to neutron transport time, they are treated by solving other equations which take care of these changes in composition.

2.3 Derivation of Neutron Transport equation

2.3.1 Definitions of some relevant physical quantities

(i) *Neutron Angular Density*

It is defined as the expected (or average) number of neutrons at position $\vec{r}$ with direction Ω and energy E at time t per unit volume per unit solid angle per unit energy interval (n/cm^3/steradian/MeV). It is represented as $N\left(\vec{r},\Omega,E,t\right)$.

If a volume element dV=dxdydz about $\vec{r}$ and a direction element $d\Omega = \sin\theta d\theta d\varphi$ about Ω are considered as shown in Fig. 2.1, then the expected number of neutrons in the volume element, having directions within

$d\Omega$ about Ω and energies within dE about E at time t, will be $N\left(\vec{r},\Omega,E,t\right)dVd\Omega dE$.

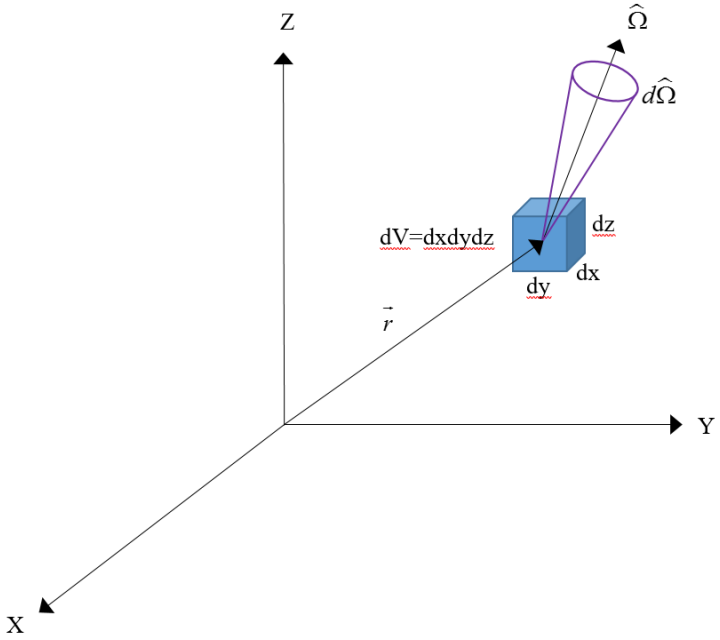

Fig. 2.1: Volume element dV and direction element $d\Omega$ considered in neutron transport theory

(ii) *Neutron Density*

Integration of the neutron angular density over all the directions is defined as the neutron density and represented as $n\left(\vec{r},E,t\right)$. Mathematically, we can write

$$n\left(\vec{r},E,t\right)=\int_{4\pi} N\left(\vec{r},\Omega,E,t\right)d\Omega$$

It is the expected number of neutrons at position $\vec{r}$ with energy E at time t per unit volume per unit energy interval (n/cm^3/MeV).

(iii) *Angular and scalar flux*

The product of neutron speed v(E) and neutron angular density $N\left(\vec{r},\Omega,E,t\right)$ is called angular flux, which is denoted by $\Psi\left(\vec{r},\Omega,E,t\right)$ [$= vN\left(\vec{r},\Omega,E,t\right)$]. This is a measure of rate of neutrons passing through per unit area per unit solid angle per unit energy. Unit of angular flux is n/cm^2/steradian/Mev/sec. If the angular flux is integrated over all the directions, then we will get scalar flux $\phi\left(\vec{r},E,t\right) = \int_{4\pi} \Psi\left(\vec{r},\Omega,E,t\right) d\Omega$.

(iv) *Cross section*

Cross section, in general, is defined as the probability of interaction of a projectile (neutron, gamma etc.) with a single nucleus of a target material. *Microscopic cross section*, denoted by σ, is the effective area presented by the target to the neutron. Its unit is given in cm^2 or barn (=10^{-24} cm^2). If microscopic cross section is multiplied with the nucleus density of target material, then we will get *Macroscopic cross section* or Σ, which characterizes the probability of neutron interaction for a volume of material and has the dimension of inverse of length (cm^{-1}).

(v) *Reaction rate*

Product of macroscopic cross section $\Sigma\left(\vec{r},\Omega,E\right)$ of any neutron-nucleus interaction and corresponding neutron angular flux $\Psi\left(\vec{r},\Omega,E,t\right)$ is defined as the reaction rate i.e. the number of interactions taking place per unit volume per unit solid angle per unit energy interval. Unit of the reaction rate is given as n/cm^3/steradian/Mev/sec.

(vi) *Neutron source*

The neutron source (strength) or $Q\left(\vec{r},\Omega,E,t\right)$ is commonly defined as the number of neutrons emitted per unit time per unit volume per unit solid angle per unit energy interval. This includes any source like fission, scattering, (n, 2n), (α, n), (γ, n) or some external neutron source. This has the dimension same as that of the reaction rate.

2.3.2 Integro differential form of transport equation

$N\left(\vec{r},\Omega,E,t\right)dVd\Omega dE$ is the expected number of neutrons in the volume element dV, having directions within the direction element $d\Omega$ about Ω and energies within dE about E at time t, as shown in Fig. 2.1. If v(E) is the speed of the neutron of energy E, then the neutron will travel v(E)Δt distance in time Δt. According to the definition of macroscopic cross section (total) $\Sigma^t\left(\vec{r},E\right)$, the probability that $N\left(\vec{r},\Omega,E,t\right)dVd\Omega dE$ number of neutrons will undergo interaction is $\Sigma^t\left(\vec{r},E\right)v\left(E\right)\Delta t$. If we assume that a neutron is lost once it undergoes an interaction, then the number of neutrons remains after time Δt will be $N\left(\vec{r},\Omega,E,t\right)\left[1-\Sigma^t\left(\vec{r},E\right)v\left(E\right)\Delta t\right]dVd\Omega dE$. These neutrons have now reached a new position coordinate $\vec{r}+\Omega v\Delta t$ at time t+Δt. There are neutrons, which are moving in some other direction Ω' with energy E′, may reach $\vec{r}+\Omega v\Delta t$ at time t+Δt due to scattering. In order to calculate the number of such neutrons, scattering reaction rate is integrated over Ω' and E′ (which results in neutrons of energy E moving

in the direction Ω), followed by multiplication with the volume element dV, direction element $d\Omega$, energy element dE and time interval Δt. Thus, we get

$$\left[\int_{\Omega'}\int_{E'}\Sigma_s\left(\vec{r};\Omega',E'\to\Omega,E\right)\Psi\left(\vec{r},\Omega',E',t\right)d\Omega'\,dE'\right]dVd\Omega dE\Delta t$$

where Σ_s is scattering cross section. In addition to this, there are neutrons from fission, external neutron sources etc., together represented as $Q\left(\vec{r},\Omega,E,t\right)$, reaching $\vec{r}+\Omega v\Delta t$ at time t+Δt. Hence, we can write

$$\begin{aligned}
&N\left(\vec{r}+\Omega v\Delta t,\Omega,E,t+\Delta t\right)dVd\Omega dE = N\left(\vec{r},\Omega,E,t\right)\left(1-\Sigma^t\left(\vec{r},E\right)v\Delta t\right)dVd\Omega dE\\
&+\left[\int_{\Omega'}\int_{E'}\Sigma_s\left(\vec{r};\Omega',E'\to\Omega,E\right)\Psi\left(\vec{r},\Omega',E',t\right)d\Omega'\,dE'\right]dVd\Omega dE\Delta t + Q\left(\vec{r},\Omega,E,t\right)dVd\Omega dE\Delta t\\
&\Rightarrow N\left(\vec{r}+\Omega v\Delta t,\Omega,E,t+\Delta t\right) = N\left(\vec{r},\Omega,E,t\right)\left(1-\Sigma^t\left(\vec{r},E\right)v\Delta t\right)\\
&+\left[\int_{\Omega'}\int_{E'}\Sigma_s\left(\vec{r};\Omega',E'\to\Omega,E\right)\Psi\left(\vec{r},\Omega',E',t\right)d\Omega'\,dE'\right]\Delta t + Q\left(\vec{r},\Omega,E,t\right)\Delta t
\end{aligned}\tag{2.1}$$

We can expand the LHS of Eq.2.1 as the following Taylor series, keeping terms up to 1st order.

$$N\left(\vec{r}+\Omega v\Delta t,\Omega,E,t+\Delta t\right)\simeq N\left(\vec{r},\Omega,E,t\right)+\vec{\nabla}N\left(\vec{r},\Omega,E,t\right).\Omega v\Delta t+\frac{\partial}{\partial t}N\left(\vec{r},\Omega,E,t\right)\Delta t \quad (2.2)$$

Using Eq.2.2, we can modify Eq.2.1 as

$$\begin{aligned}
&\frac{1}{v(E)}\frac{\partial}{\partial t}\Psi\left(\vec{r},\Omega,E,t\right)+\Omega.\vec{\nabla}\Psi\left(\vec{r},\Omega,E,t\right)+\Sigma^t\left(\vec{r},E\right)\Psi\left(\vec{r},\Omega,E,t\right)\\
&=\left[\int_{\Omega'}\int_{E'}\Sigma_s\left(\vec{r};\Omega',E'\to\Omega,E\right)\Psi\left(\vec{r},\Omega',E',t\right)d\Omega'\,dE'\right]+Q\left(\vec{r},\Omega,E,t\right)
\end{aligned}\tag{2.3}$$

This is the integro differential form of neutron transport equation [58][59]. The steady state neutron transport equation can be obtained by setting $\frac{\partial\Psi}{\partial t}=0$.

$$\Omega.\vec{\nabla}\Psi\left(\vec{r},\Omega,E\right)+\Sigma^{t}\left(\vec{r},E\right)\Psi\left(\vec{r},\Omega,E\right)$$
$$=\left[\int_{\Omega'}\int_{E'}\Sigma_{s}\left(\vec{r};\Omega',E'\rightarrow\Omega,E\right)\Psi\left(\vec{r},\Omega',E'\right)d\Omega'\,dE'\right]+Q\left(\vec{r},\Omega,E\right) \quad (2.4)$$

In 3D geometry, Ω has components sinθcosφ, sinθsinφ and cosφ along X, Y and Z axes respectively where θ is polar angle and φ is azimuthal angle. In plane geometry where the medium is finite along X axis and infinite along Y and Z axes, Ω has the only component μ = cosθ along X axis. Hence, time independent neutron transport equation in plane geometry can be written as

$$\mu\frac{\partial\Psi\left(x,\mu,E\right)}{\partial x}+\Sigma^{t}\left(x,E\right)\Psi\left(x,\mu,E\right)$$
$$=\left[2\pi\int_{-1}^{1}\int_{E'}\Sigma_{s}\left(x;\mu',E'\rightarrow\mu,E\right)\Psi\left(x,\mu',E'\right)d\mu'\,dE'\right]+Q\left(x,\mu,E\right) \quad (2.5)$$

where μ is incident direction and μ′ is scattered direction of neutron and $\int_{\Omega'}d\Omega'=\int_{-1}^{1}d\mu'\int_{0}^{2\pi}d\varphi'=2\pi\int_{-1}^{1}d\mu'$.

2.3.3 Integral form of transport equation

The integral form of neutron transport equation can be derived from the integro differential form. To start with, scattering source term on RHS of Eq.2.3 can be included in $Q\left(\vec{r},\Omega,E,t\right)$. Hence,

$$\frac{1}{v\left(E\right)}\frac{\partial}{\partial t}\Psi\left(\vec{r},\Omega,E,t\right)+\Omega.\vec{\nabla}\Psi\left(\vec{r},\Omega,E,t\right)+\Sigma^{t}\left(\vec{r},E\right)\Psi\left(\vec{r},\Omega,E,t\right)=Q\left(\vec{r},\Omega,E,t\right) \quad (2.6)$$

Using the expanded form of gradient operator ($\vec{\nabla}$) in Cartesian coordinate system, first two terms on LHS of Eq.2.6 can be written as

$$\left(\frac{1}{v(E)}\frac{\partial}{\partial t}+\Omega.\vec{\nabla}\right)\Psi\left(\vec{r},\Omega,E,t\right)=\left(\frac{1}{v(E)}\frac{\partial}{\partial t}+\Omega_x\frac{\partial}{\partial x}+\Omega_y\frac{\partial}{\partial y}+\Omega_z\frac{\partial}{\partial z}\right)\Psi\left(\vec{r},\Omega,E,t\right) \quad (2.7)$$

where Ω_x, Ω_y and Ω_z are the direction cosines of the line along which neutron travels at position $\vec{r}$. Eq.2.7 can alternatively be represented by the following total derivative of angular flux with respect to the distance "s" travelled by the neutron along Ω.

$$\frac{d}{ds}\Psi\left(\vec{r},\Omega,E,t\right)=\left(\frac{dt}{ds}\frac{\partial}{\partial t}+\frac{dx}{ds}\frac{\partial}{\partial x}+\frac{dy}{ds}\frac{\partial}{\partial y}+\frac{dz}{ds}\frac{\partial}{\partial z}\right)\Psi\left(\vec{r},\Omega,E,t\right) \quad (2.8)$$

Comparing Eq.2.7 and 2.8, we get

$$\frac{dt}{ds}=\frac{1}{v}\Rightarrow t=t_0+\frac{s}{v} \quad (2.9)$$

and

$$\left.\begin{aligned}\frac{dx}{ds}=\Omega_x\Rightarrow x=x_0+\Omega_x s\\ \frac{dy}{ds}=\Omega_y\Rightarrow y=y_0+\Omega_y s\\ \frac{dz}{ds}=\Omega_x\Rightarrow z=z_0+\Omega_z s\end{aligned}\right\}\rightarrow\vec{r}=\vec{r_0}+\Omega s \quad (2.10)$$

Using the equivalence of Eq.2.7 and 2.8, $\left(\frac{1}{v(E)}\frac{\partial}{\partial t}+\Omega.\vec{\nabla}\right)\Psi$ is replaced by the total derivative $\left(\frac{d\Psi}{ds}\right)$. Therefore, Eq.2.6 can be rewritten as

$$\left[\frac{d}{ds}+\Sigma^t\left(\vec{r_0}+\Omega s,E\right)\right]\Psi\left(\vec{r_0}+\Omega s,\Omega,E,t_0+\frac{s}{v}\right)=Q\left(\vec{r_0}+\Omega s,\Omega,E,t_0+\frac{s}{v}\right) \quad (2.11)$$

This is a first order linear ordinary differential equation, which has a solution of following form.

$$\Psi\left(\vec{r}_0+\Omega s,\Omega,E,t_0+\frac{s}{v}\right)=\Psi\left(\vec{r}_0,\Omega,E,t_0\right)e^{-\int_0^s\Sigma^t\left(\vec{r}_0+\Omega s',E\right)ds'} + \int_0^s Q\left(\vec{r}_0+\Omega s',\Omega,E,t_0+\frac{s'}{v}\right)e^{-\int_{s'}^s\Sigma^t\left(\vec{r}_0+\Omega s'',E\right)ds''}\,ds' \tag{2.12}$$

This means the angular flux at $\vec{r}$ is contributed by (i) the incoming angular flux, attenuated while travelling from $\vec{r}_0$ to $\vec{r}$ and (ii) neutron sources present between $\vec{r}_0$ and $\vec{r}$ on the line along Ω, attenuated before reaching $\vec{r}$ (Fig. 2.2). If the boundary is set at vacuum, then $\Psi\left(\vec{r}_0,\Omega,E,t_0\right)$ will be zero and hence,

$$\Psi\left(\vec{r},\Omega,E,t\right)=\int_0^s Q\left(\vec{r}_0+\Omega s',\Omega,E,t_0+\frac{s'}{v}\right)e^{-\int_{s'}^s\Sigma^t\left(\vec{r}_0+\Omega s'',E\right)ds''}\,ds' \tag{2.13}$$

This is the integral form of neutron transport equation [58][59].

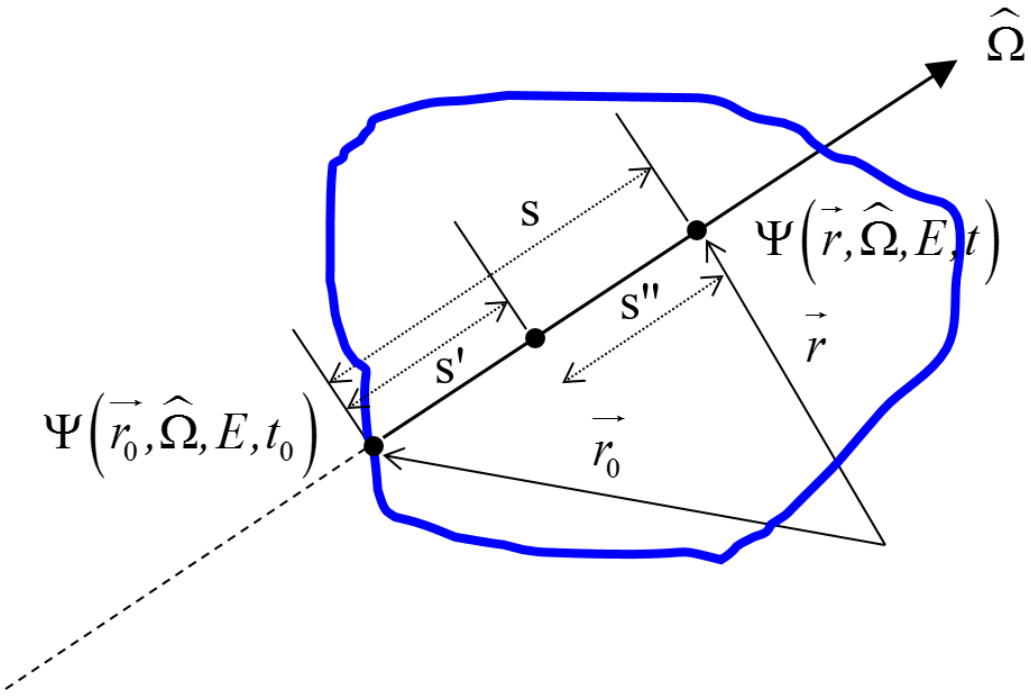

Fig. 2.2: Illustration of symbols used in the derivation of integral transport equation

2.3.4 Initial and boundary conditions

The time dependent neutron transport equation (Eq.2.3) is first order in time as well as space. So, one initial and one boundary condition is sufficient to solve the equation. As initial condition, angular flux $\Psi\left(\vec{r},\Omega,E,t\right)$ at t=0 sec must be known. For fixing the boundary condition, let us consider a domain of volume V surrounded by an arbitrary surface S within which the transport equation is to be solved. We can set the angular flux, incoming to the domain through S, to zero i.e.

$$\Psi\left(\vec{r},\Omega,E,t\right)=0 \qquad \forall\ \Omega.n<0 \tag{2.14}$$

where n is the outward normal to S. This is known as the *vacuum boundary condition* since it implies that a neutron once leaks out from the domain will not return. This condition remains valid if the surface is non reentrant in nature (Fig. 2.3).

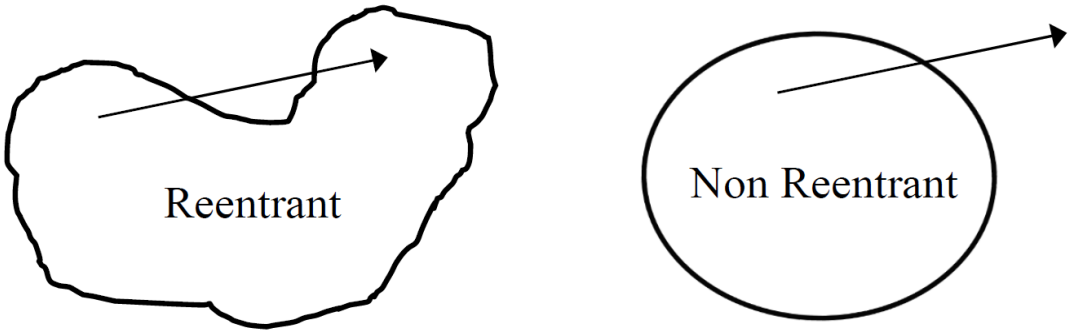

Fig. 2.3: Illustration of reentrant and non-reentrant surface

For full core calculation, vacuum boundary condition can be used whereas for lattice level calculation, we have a choice of following boundary conditions depending on the symmetry of the problem. Incoming angular flux in a direction Ω at boundary S

is set equal to the fraction of outgoing angular flux in a direction Ω′ on the same boundary i.e.

$$\Psi\left(\vec{r},\Omega,E,t\right)=\alpha\left(E\right)\Psi\left(\vec{r},\Omega',E,t\right) \quad \forall\ \Omega.n<0 \tag{2.15}$$

where Ω′ and Ω are the incident and reflected direction of neutron at boundary S respectively and α(E) is the isotropic albedo. This condition is known as *albedo boundary condition*. If the albedo is 1.0 i.e. incoming angular flux is set equal to the outgoing angular flux, then we will get *reflective boundary condition*.

$$\Psi\left(\vec{r},\Omega,E,t\right)=\Psi\left(\vec{r},\Omega',E,t\right) \quad \forall\ \Omega.n<0 \tag{2.16}$$

This condition is used for those boundaries which are having reflection symmetry.

Under Wigner-Seitz approximation, square boundary, keeping the area within the boundary conserved, can be converted into circular one in order to simplify the geometry to that of a one dimensional problem. However, this approximation gives contradictory results in case of lattices having fuel pin surrounded by thin moderator region. In order to resolve this problem, we use *white boundary condition*, which assumes that the neutrons passing out of the domain return with an isotropic distribution.

$$\Psi\left(\vec{r},\Omega,E,t\right)=\frac{1}{4}\int_{\Omega'.n>0}\Psi\left(\vec{r},\Omega',E,t\right)\Omega'.n\,d\Omega' \quad \forall\ \Omega.n<0 \tag{2.17}$$

In case of a periodic lattice as shown in Fig. 2.4, the angular flux distribution on one boundary “a” is equal to that in another boundary “b”, known as *periodic boundary condition*.

$$\Psi\left(\vec{r_a},\Omega,E,t\right)=\Psi\left(\vec{r_b},\Omega,E,t\right) \quad \forall\ \Omega.n<0 \tag{2.18}$$

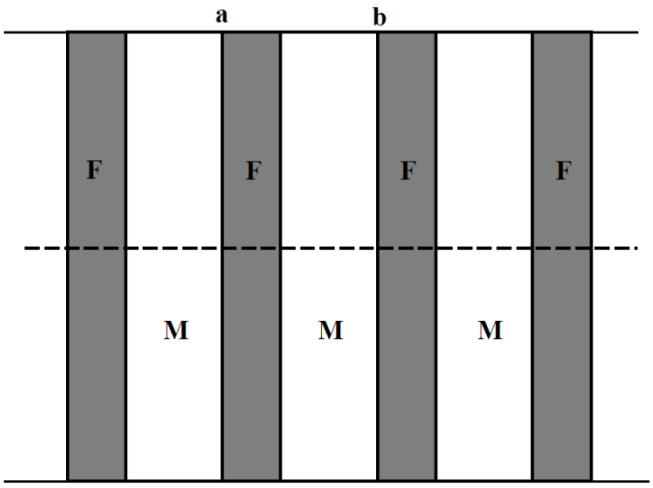

Fig. 2.4: Periodic array of fuel (F) and moderator (M) slabs

2.4 Solution of Neutron Transport Equation

2.4.1 The Multigroup method

In multigroup method, neutron energy E, which is a continuous variable within the range of interest ($E_{min} \leq E \leq E_{max}$), is divided into a finite number (G) of intervals separated by energy $\Delta E_g = (E_g - E_{g-1})$, where g =1, 2, 3, …., G (energy increases as g decreases), as shown in Fig. 2.5. This division is made, wherever possible, in order to keep the variation of cross section reasonably small within an interval. For some cases, the intervals are also chosen such that (E_g/E_{g+1}) is roughly constant i.e. lethargy intervals are equal.

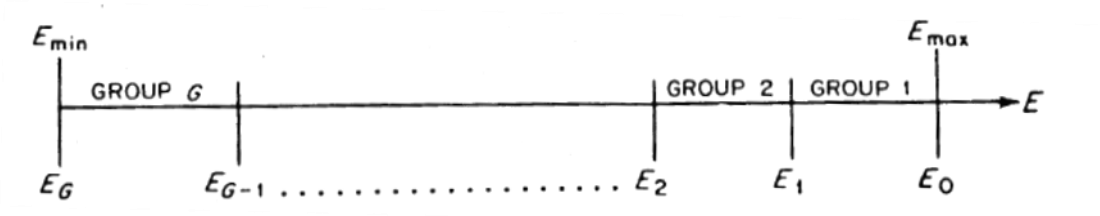

Fig. 2.5: Division of continuously varying neutron energy into discrete intervals

The angular dependence of the angular flux Ψ, scattering cross section Σ_s and neutron source Q in Eq.2.5 is represented by P_N or S_N method, which will be discussed in the subsequent sections. The integration over E′ in Eq.2.5 is expressed as the sum of integrations over all energy group intervals i.e.

$$\int_{E'} dE' = \sum_{g'=1}^{G} \int_{E_{g'}}^{E_{g'-1}} dE' = \sum_{g'=1}^{G} \int_{g'} dE' \tag{2.19}$$

Finally, Eq.2.5 is integrated over the g-th energy interval ($E_g \leq E \leq E_{g-1}$) to obtain the following group constants.

$$\phi_g(x) = \int_{E_g}^{E_{g-1}} \phi(x,E)\, dE \tag{2.20}$$

$$\Sigma_g^t(x) = \frac{\int_{E_g}^{E_{g-1}} \Sigma^t(x,E)\phi(x,E)\, dE}{\phi_g(x)} \tag{2.21}$$

$$\Sigma_{sg'\to g}(x) = \frac{\int_{E_g}^{E_{g-1}} \int_{g'} \Sigma_s(x; E' \to E)\phi(x,E')dE'\, dE}{\phi_{g'}(x)} \tag{2.22}$$

$$\text{and } Q_g(x) = \int_{E_g}^{E_{g-1}} Q(x,E)\, dE \tag{2.23}$$

While deriving the above constants, it is assumed that the angular dependence of Ψ, Σ_s and Q are decoupled from the spatial dependence. The group constants are independent of energy within g-th interval. Thus, neutron transport equation is converted into G number of coupled ordinary differential equations with abovementioned group constants. Values of these group constants depend on the evaluation of ϕ(x, E).

2.4.2 P_N method

In neutron transport theory, angular flux (Ψ), scattering cross section (Σ_s) and neutron source (Q) depend on the angular variables θ and φ. In P_N method, this angular dependence is expressed in terms of Spherical harmonics [$Y^l_m(\theta, \varphi)$] in case of general geometry where no azimuthal symmetry exists. For plane geometry, the angular dependence is expressed in terms of Legendre polynomial as indicated below.

$$\Psi(x,\mu,E)=\sum_{m=0}^{\infty}\left(\frac{2m+1}{4\pi}\right)\phi_m(x,E)P_m(\mu) \tag{2.24}$$

$$\Sigma_s(x;\mu',E'\to\mu,E)=\sum_{l=0}^{\infty}\left(\frac{2l+1}{4\pi}\right)\Sigma_{sl}(x;E'\to E)P_l(\mu')P_l(\mu) \tag{2.25}$$

$$Q(x,\mu,E)=\sum_{m=0}^{\infty}\left(\frac{2m+1}{4\pi}\right)Q_m(x,E)P_m(\mu) \tag{2.26}$$

Putting these expansions into Eq.2.5 and writing the transport equation in multigroup form, as discussed in the previous section, we get

$$\begin{aligned}&\mu\frac{\partial}{\partial x}\left[\sum_{m=0}^{\infty}\left(\frac{2m+1}{4\pi}\right)\phi_{mg}(x)P_m(\mu)\right]+\Sigma_g^t(x)\sum_{m=0}^{\infty}\left(\frac{2m+1}{4\pi}\right)\phi_{mg}(x)P_m(\mu)\\&=\left[2\pi\int_{-1}^{1}\int_{E'}\left[\begin{array}{l}\left\{\sum_{l=0}^{\infty}\left(\frac{2l+1}{4\pi}\right)\Sigma_{slg'\to g}(x)P_l(\mu')P_l(\mu)\right\}\\\left\{\sum_{m=0}^{\infty}\left(\frac{2m+1}{4\pi}\right)\phi_{mg'}(x)P_m(\mu')\right\}\end{array}\right]d\mu'\,dE'\right]\\&+\sum_{m=0}^{\infty}\left(\frac{2m+1}{4\pi}\right)Q_{mg}(x)P_m(\mu)\end{aligned} \tag{2.27}$$

Using recurrence relation and orthogonality property of Legendre polynomial, Eq.2.29 is converted into a set of equations.

$$n\frac{d\phi_{n-1}(x)}{dx}+(n+1)\frac{d\phi_{n+1}(x)}{dx}+(2n+1)\Sigma_n(x)\phi_n(x)=(2n+1)Q_n(x) \tag{2.28}$$

where n = 0, 1, 2, 3,….. This infinite set of equations is truncated by setting

$$\frac{d\phi_{N+1}(x)}{dx} = 0 \tag{2.29}$$

This is known as P_N approximation, which makes total number of equations limited to (N+1). In order to obtain the solution of transport equation, the problem domain is divided into a number of meshes within which the P_N equations are solved.

P_N method is based on the integro-differential form of neutron transport equation (Eq.2.3). Therefore, this method is suitable for regular geometry.

2.4.3 Discrete Ordinate method

The Discrete Ordinate method, also known as the S_N method, is used extensively in reactor physics calculations to solve the neutron transport equation numerically. The basic assumption of this method lies in the treatment of angular variable on which the neutron flux depends. Unlike P_N method, where the flux is treated as a continuous function of the angular variable, a number of discrete directions are chosen in the S_N method to evaluate the flux. The choice of the set of finite number of directions is an important aspect to be looked into. Gauss quadrature set is usually chosen for this purpose since it offers accurate result for the integration of a polynomial of order $n \leq (2N-1)$ where N is number of discrete directions considered in the quadrature set. The angular variable is discretized as

$$\int_{-1}^{1} F(\mu)\, d\mu = \sum_{i=1}^{N} w_i F(\mu_i) \tag{2.30}$$

where $F(\mu) \sim \mu^n$ and w_i is the weight of μ_i, which is the i-th discrete value of μ in the quadrature set. These weights are always positive and symmetric in nature.

Starting with the neutron transport equation for the slab geometry as given in Eq.2.5 and expanding the scattering cross section in terms of Legendre polynomial, as shown in Eq.2.27, we get

$$\mu\frac{\partial\Psi(x,\mu,E)}{\partial x}+\Sigma^{t}(x,E)\Psi(x,\mu,E)$$
$$=\left[2\pi\int_{-1}^{1}\int_{E'}\sum_{l=0}^{\infty}\left(\frac{2l+1}{4\pi}\right)\Sigma_{sl}(x;E'\to E)P_l(\mu')P_l(\mu)\Psi(x,\mu',E')\,d\mu'\,dE'\right] \tag{2.31}$$
$$+Q(x,\mu,E)$$

In multi-group format, above equation can be written as

$$\mu\frac{\partial\Psi_g(x,\mu)}{\partial x}+\Sigma_g^{t}(x)\Psi_g(x,\mu)$$
$$=\left[2\pi\int_{-1}^{1}\sum_{g'=1}^{G}\sum_{l=0}^{\infty}\left(\frac{2l+1}{4\pi}\right)\Sigma_{sl,g'\to g}(x)P_l(\mu')P_l(\mu)\Psi_{g'}(x,\mu')\,d\mu'\right]+Q_g(x,\mu) \tag{2.32}$$

In the S_N method, discrete values of μ, which are based on some suitable quadrature set, are chosen. For a discrete value μ_i, Eq.2.34 can be written as

$$\mu_i\frac{\partial\Psi_g(x,\mu_i)}{\partial x}+\Sigma_g^{t}(x)\Psi_g(x,\mu_i)$$
$$=\left[\sum_{l=0}^{\infty}\left(\frac{2l+1}{4\pi}\right)P_l(\mu_i)\sum_{g'=1}^{G}\Sigma_{sl,g'\to g}(x)\phi_{lg'}(x)\right]+Q_g(x,\mu_i) \tag{2.33}$$

where $\phi_{lg'}(x)=2\pi\int_{-1}^{1}P_l(\mu')\Psi_{g'}(x,\mu')\,d\mu'$. Once angular flux in the direction μ_i is obtained using Eq.2.35, the scalar flux is calculated by integrating the angular fluxes for all the discrete directions chosen.

Though S_N method is quite popular in reactor physics community, it has number of drawbacks. First, the method is difficult to apply for irregular geometries. Second, the iterative solution technique, commonly adopted in S_N method, is extremely slow for scattering dominant problem and third, the method encounters negative scalar flux due to Diamond differencing in case of absorption dominant problem with isolated source.

2.4.4 Collison probability method

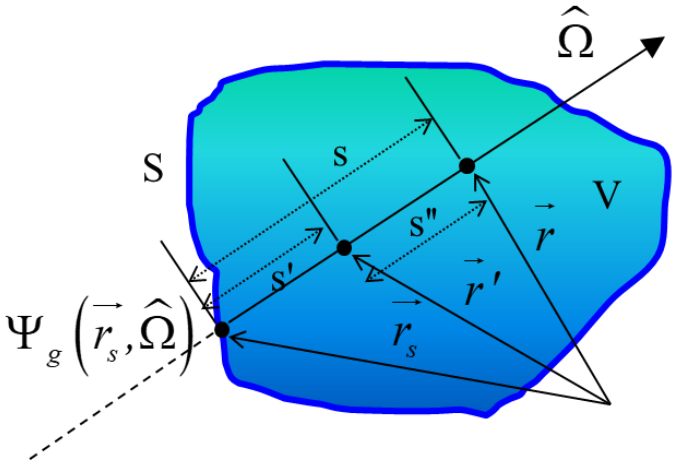

Fig. 2.6: Illustration of collision probability method

Let us consider a volume V enclosed by a surface S, as shown in Fig. 2.6. We are interested in calculating the angular flux $\Psi_g\left(\vec{r},\Omega\right)$ at $\vec{r}$ $(=\vec{r}_s+\Omega s)$ inside V. The angular flux is contributed by (i) those neutrons which are travelling from other points $\vec{r}\,'$ $(=\vec{r}_s+\Omega s')$ inside the volume V to the point of interest $\vec{r}$ along Ω and (ii) those neutrons which have reached the surface S from elsewhere and now they are travelling towards the point $\vec{r}$ along Ω. At $\vec{r}\,'$, there is a neutron source $Q_g\left(\vec{r}\,',\Omega\right)$ which includes both scattering as well as fission source. The neutrons, contributed by this source, interact (or collide) with the medium while travelling from $\vec{r}\,'$ to $\vec{r}$ and hence they are attenuated by a factor $e^{-\int_{s'}^{s}\Sigma_g^t\left(\vec{r}_s+\Omega s''\right)ds''}$. Contribution of these neutrons to the angular flux $\Psi_g\left(\vec{r},\Omega\right)$ can be written as $\int_0^s Q_g\left(\vec{r}_s+\Omega s',\Omega\right)e^{-\int_{s'}^{s}\Sigma_g^t\left(\vec{r}_s+\Omega s''\right)ds''}ds'$ [similar to Eq.2.13]. If the angular flux at surface S along Ω is $\Psi_g\left(\vec{r}_s,\Omega\right)$, then angular flux reaching $\vec{r}$ from

$\vec{r}_s$ after encountering collision with the medium will be $\Psi_g\left(\vec{r}_s,\Omega\right)e^{-\int_0^s \Sigma_g^t\left(\vec{r}_s+\Omega s'\right)ds'}$.

Therefore, angular flux at $\vec{r}$ in the direction Ω is

$$\Psi_g\left(\vec{r},\Omega\right)=\int_0^s Q_g\left(\vec{r}_s+\Omega s',\Omega\right)e^{-\int_{s'}^s \Sigma_g^t\left(\vec{r}_s+\Omega s''\right)ds''}ds'+\Psi_g\left(\vec{r}_s,\Omega\right)e^{-\int_0^s \Sigma_g^t\left(\vec{r}_s+\Omega s'\right)ds'}$$
$$=\int_0^s Q_g\left(\vec{r}',\Omega\right)e^{-\tau_{rr'}^g}ds'+\Psi_g\left(\vec{r}_s,\Omega\right)e^{-\tau_{rr_s}^g}$$

(2.34)

where $\tau_{rr'}^g=\int_{r'}^r \Sigma_g^t\left(R\right)dR$ and $\tau_{rr_s}^g=\int_{r_s}^r \Sigma_g^t\left(R\right)dR$. Integrating Eq.2.36 over Ω, we get the scalar flux $\phi_g(r)$.

$$\phi_g\left(r\right)=\int\Psi_g\left(\vec{r},\Omega\right)d\Omega$$
$$=\int\int_0^s Q_g\left(\vec{r}',\Omega\right)e^{-\tau_{rr'}^g}ds'\frac{dA'}{|\vec{r}-\vec{r}'|^2}+\int\Psi_g\left(\vec{r}_s,\Omega\right)e^{-\tau_{rr_s}^g}\frac{\overrightarrow{dA_s}\cdot\Omega}{|\vec{r}-\vec{r}_s|^2} \qquad (2.35)$$
$$=\int_V\frac{Q_g\left(\vec{r}',\Omega\right)e^{-\tau_{rr'}^g}}{|\vec{r}-\vec{r}'|^2}dV'+\int_S\frac{J_g^{in}\left(\vec{r}_s,\Omega\right)e^{-\tau_{rr_s}^g}}{|\vec{r}-\vec{r}_s|^2}dA_s$$

Eq.2.37 leads to forming a nodal balance equation for the k-th node shown in Fig. 2.7.

$$\Sigma_{tg}^k\phi_g^kV^k=\sum_i Q_g^iV^iP_g^{ik}+\sum_n J_{in,g}^{S_n}P_g^{S_nk} \qquad (2.36)$$

where the collision probabilities are given as

$$P_g^{ik}=\frac{\int_{V^k}\left(\int_{V^i}\frac{Q_g\left(r',\Omega\right)e^{-\tau_{r'r}^g}}{|r-r'|^2}dV'\right)\Sigma_{tg}\left(r\right)dV}{\int_{V^i}\int_\Omega Q_g\left(r',\Omega\right)d\Omega dV'}$$

$$P_g^{S_nk}=\frac{\int_{V^k}\left(\int_{S_n}\frac{J_g^{in}\left(r_s,\Omega\right)e^{-\tau_{r_sr}^g}}{|r-r_s|^2}dS\right)\Sigma_{tg}\left(r\right)dV}{\int_{S_n}\int_\Omega J_g^{in}\left(r_s,\Omega\right)d\Omega dS}$$

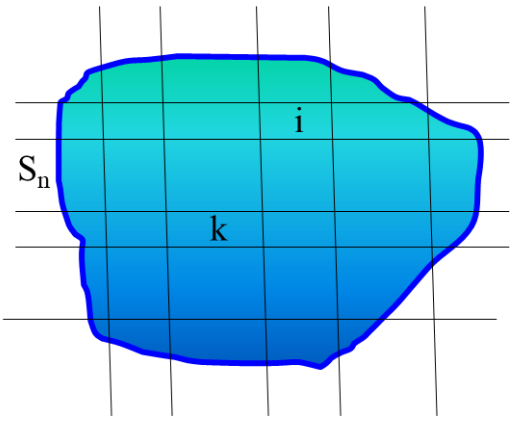

Fig. 2.7: Nodalization of a problem domain

Since, the collision probability method solves the integral form of neutron transport equation (Eq.2.13), it can be applied for any geometry. However, this method suffers from the issue of storing huge data (P^{ik}-s') during computation. For example, if there are 100 meshes in X and Y directions each, then there will be 10^8 number of P^{ik} values to be computed for 1 energy group. Multiplying this with total number of energy groups, say 100, total number of P^{ik} values to be stored would be 10^{10}, which is quite huge.

2.4.5 Method of characteristics

In Method of Characteristics or MOC, angular variable is discretized into a number of directions (like S_N method) and spatial variable is discretized into a number of meshes. For each direction, a set of parallel characteristic lines or rays are traced over the problem domain, which has already been divided into meshes. The intercepts of rays in different meshes are then calculated. Applying MOC, the neutron transport equation, which is a partial differential equation, is converted into a set of ordinary differential equations, which are solved analytically along the rays to obtain the mesh averaged ray angular flux and outgoing mesh edge ray angular flux. These mesh averaged ray angular fluxes are then summed for all those rays passing through a mesh in a particular direction to calculate the average angular flux of the mesh for the direction. Finally, the average angular flux is integrated over all the directions to get the scalar flux. This formalism does not depend on the mesh geometry. Hence, it is able to treat complex geometries commonly encountered in the reactor core, which brings its applicability at par with the Monte Carlo methods. Details of this method is given in Chapter 3.

2.5 Benchmark problems with reflective boundary condition

2.5.1 Analytical benchmark test set

The reference paper [60] describes a set of benchmark problems with analytical solution of neutron transport equation for infinite medium. Purpose of the test set is to verify the correctness of algorithm implemented in neutron transport codes. The authors of the paper believe that the transport code, before acceptance, must be able to produce solution accurate to at least five decimal places of the results reported in the reference. The test set includes 18 benchmark problems for infinite medium with one material; one, two, three and six energy groups and isotropic scattering. For convenience in referring the problems, unique identifier is used. First problem of the test set can be identified as Pua-1-0-IN, which means the fissile material is Plutonium-a (Pua), the number of energy group is one (1), scattering is isotropic (0) and the problem is for an infinite medium (IN). In similar fashion, identifiers are used for rest of the problems. Cross sections of materials of all the problems along with fission spectrum are given in Table- 2.1, Table- 2.2 and Table- 2.3.

.

Table- 2.1: One group cross sections (cm^{-1}) of analytical benchmark test set

Benchmark problem: Pua-1-0-IN							
Material	g	χ	Σ_f	Σ_a	$\nu\Sigma_f$	Σ_t	$\Sigma_{sg\rightarrow1}$
Pua	1	1.0	0.0816	0.101184	0.264384	0.3264	0.225216
Benchmark problem: Pub-1-0-IN							
Pub	1	1.0	0.0816	0.101184	0.231744	0.3264	0.225216
Benchmark problem: Ua-1-0-IN							
Ua	1	1.0	0.065280	0.078336	0.176256	0.3264	0.248064
Benchmark problem: Ub-1-0-IN							
Ub	1	1.0	0.065280	0.078336	0.182594753	0.3264	0.248064
Benchmark problem: Uc-1-0-IN							
Uc	1	1.0	0.065280	0.078336	0.176733066	0.3264	0.248064
Benchmark problem: Ud-1-0-IN							
Ud	1	1.0	0.065280	0.078336	0.174898045	0.3264	0.248064
Benchmark problem: UD2O-1-0-IN							
UD2O	1	1.0	0.054628	0.081942	0.0928676	0.54628	0.464338
Benchmark problem: Ue-1-0-IN							
Ue	1	1.0	0.06922744	0.079365	0.1730686	0.407407	0.328042

Table- 2.2: Two group cross sections (cm^{-1}) of analytical benchmark test set

Benchmark problem: Pu-2-0-IN								
Material	g	X	Σ_f	Σ_a	$\nu\Sigma_f$	Σ_t	$\Sigma_{sg\to1}$	$\Sigma_{sg\to2}$
Pu	1	0.575	0.0936	0.0984	0.29016	0.2208	0.0792	0.0432
	2	0.425	0.08544	0.09984	0.2503392	0.3360	0.0	0.23616
Benchmark problem: U-2-0-IN								
U	1	0.575	0.06192	0.06576	0.167184	0.2160	0.078240	0.0720
	2	0.425	0.06912	0.08256	0.1728	0.3456	0.0	0.26304
Benchmark problem: UAL-2-0-IN								
UAL	1	1.0	0.0	0.000222	0.0	0.26817	0.247516	0.020432
	2	0.0	0.06070636	0.06385	0.171799	1.27698	0.0	1.21313
Benchmark problem: URRa-2-0-IN								
URRa	1	1.0	0.0010484	0.002053	0.002621	0.65696	0.62568	0.029227
	2	0.0	0.050632	0.07642	0.12658	2.52025	0.0	2.44383
Benchmark problem: URRb-2-0-IN								
URRb	1	1.0	0.000836	0.00194	0.00209	0.88721	0.83892	0.04635
	2	0.0	0.029564	0.053633	0.07391	2.9727	0.000767	2.9183
Benchmark problem: URRc-2-0-IN								
URRc	1	1.0	0.001648	0.00312	0.00412	0.88655	0.83807	0.04536
	2	0.0	0.057296	0.08654	0.14324	2.9628	0.00116	2.8751
Benchmark problem: URRd-2-0-IN								
URRd	1	1.0	0.61475	0.6167162	0.617209	0.650917	0.0	0.0342008
	2	0.0	0.045704	0.0692	0.11426	2.13800	0.0	2.06880
Benchmark problem: UD2O-2-0-IN								
UD2O	1	1.0	0.002817	0.0115248	0.0070425	0.33588	0.31980	0.0045552
	2	0.0	0.097	0.12218	0.2425	0.54628	0.0	0.42410

Table- 2.3: Multigroup cross sections (cm^{-1}) of analytical benchmark test set

Benchmark problem: URR-3-0-IN											
Material	g	X	Σ_f	Σ_a	$\nu\Sigma_f$	Σ_t	$\Sigma_{sg\to1}$	$\Sigma_{sg\to2}$	$\Sigma_{sg\to3}$		
URR	1	0.96	0.006	0.012	0.018	0.240	0.024	0.171	0.033		
	2	0.04	0.060	0.100	0.150	0.975	0.000	0.600	0.275		
	3	0.0	0.900	1.100	1.800	3.100	0.000	0.000	2.000		
Benchmark problem: URR-6-0-IN											
Material	G	X	Σ_f	Σ_a	$\nu\Sigma_f$	Σ_t	$\Sigma_{sg\to1}$	$\Sigma_{sg\to2}$	$\Sigma_{sg\to3}$	$\Sigma_{sg\to4}$	$\Sigma_{sg\to5}$	$\Sigma_{sg\to6}$
URR	1	0.48	0.006	0.012	0.018	0.240	0.024	0.171	0.033	0.000	0.000	0.000
	2	0.02	0.060	0.100	0.150	0.975	0.000	0.600	0.275	0.000	0.000	0.000
	3	0.0	0.900	1.100	1.800	3.100	0.000	0.000	2.000	0.000	0.000	0.000
	4	0.0	0.900	1.100	1.800	3.100	0.000	0.000	0.000	2.000	0.000	0.000
	5	0.02	0.060	0.100	0.150	0.975	0.000	0.000	0.000	0.275	0.600	0.000
	6	0.48	0.006	0.012	0.018	0.240	0.000	0.000	0.000	0.033	0.171	0.024

2.5.2 BWR benchmark problem

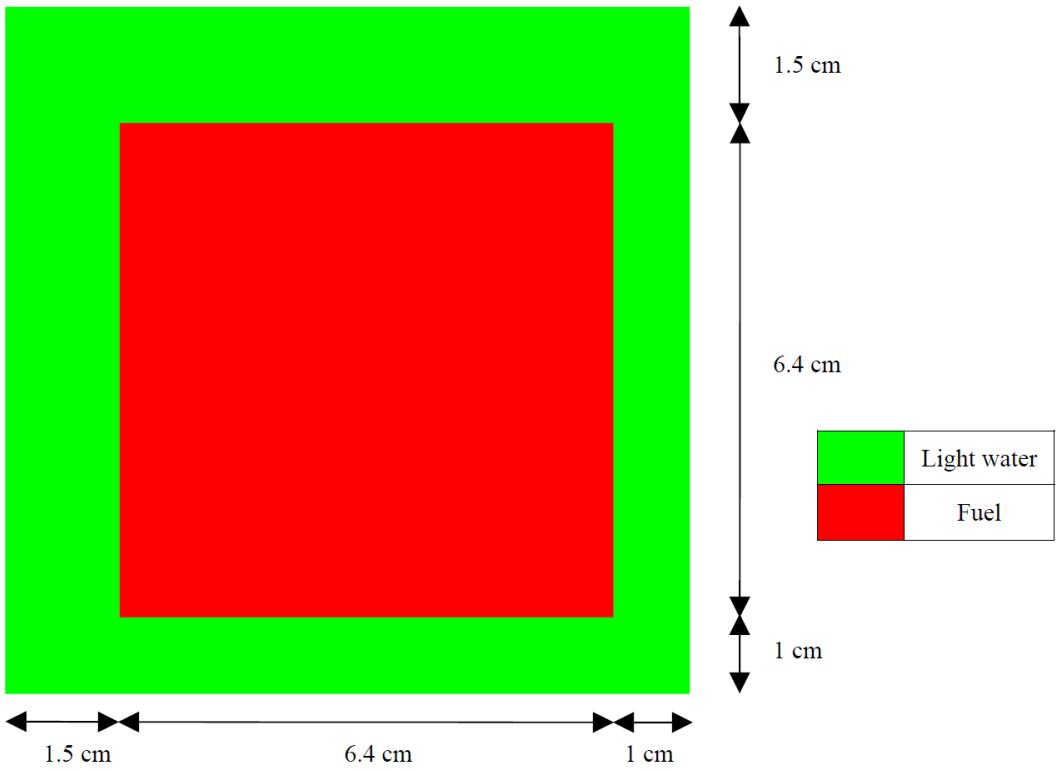

Fig. 2.8: BWR benchmark problem. Dimensions are shown in the figure. Representation of different materials with different colours is indicated in the legend table.

This problem is taken from [30]. It is a two energy group, isotropic scattering problem for lattice of a boiling water reactor (BWR). As shown in Fig. 2.8, the lattice consists of two regions; the outer region is a square of size 8.9 cm × 8.9 cm and the inner region, which is a square of size 6.4 cm × 6.4 cm, is off centre by 0.25 cm towards positive X axis as well as negative Y axis with respect to the centre of the outer region. The inner and outer regions are filled with homogenized fuel and light water respectively. Reflective boundary condition is applied at the outermost surface of the lattice. Cross sections of both the materials along with fission spectrum are given in Table- 2.4.

Table- 2.4: Cross sections (cm^{-1}) of BWR benchmark problem (χ: 1.0, 0.0)

Material	g	Σ_a	$\nu\Sigma_f$	Σ_t	$\Sigma_{sg\to1}$	$\Sigma_{sg\to2}$
Fuel	1	8.627E-3	6.203E-3	1.96647E-1	1.78E-1	1.002E-2
	2	6.957E-2	1.101E-1	5.96159E-1	1.089E-3	5.255E-1
Light water	1	6.84E-4	0.0	2.22064E-1	1.995E-1	2.188E-2
	2	8.016E-3	0.0	8.87874E-1	1.558E-3	8.783E-1

2.5.3 LWR benchmark problem with burnable poison

This problem is also taken from [30]. It is a two energy group, isotropic scattering problem designed for a LWR lattice. Unlike the earlier problem, the lattice contains burnable poison. In the lattice of size 8.9 cm × 9.5 cm, the poison element is present only inside a square region of size 1.6 cm × 1.6 cm, which is surrounded by another square region of size 6.4 cm × 6.4 cm and filled with uranium. Rest of the lattice is filled with light water. Fig. 2.9 shows the lattice. Reflective boundary condition is applied at the outermost boundary. Fission spectrum and cross sections of uranium, light water and burnable poison are given in Table- 2.5.

Table- 2.5: Cross sections (cm^{-1}) of LWR benchmark problem with burnable poison (χ: 1.0, 0.0)

Material	g	Σ_a	$\nu\Sigma_f$	Σ_t	$\Sigma_{sg\to1}$	$\Sigma_{sg\to2}$
Fuel	1	9.0E-3	6.2E-3	1.99E-1	1.8E-1	1.00E-2
	2	7.0E-2	1.1E-1	6.01E-1	1.0E-3	5.3E-1
Light water	1	7.0E-4	0.0	2.227E-1	2.0E-1	2.2E-2
	2	8.0E-3	0.0	8.9E-1	2.0E-3	8.8E-1
Burnable poison	1	9.0E-3	6.2E-3	1.99E-1	1.8E-1	1.00E-2
	2	3.0	1.1E-1	3.531	1.0E-3	5.3E-1

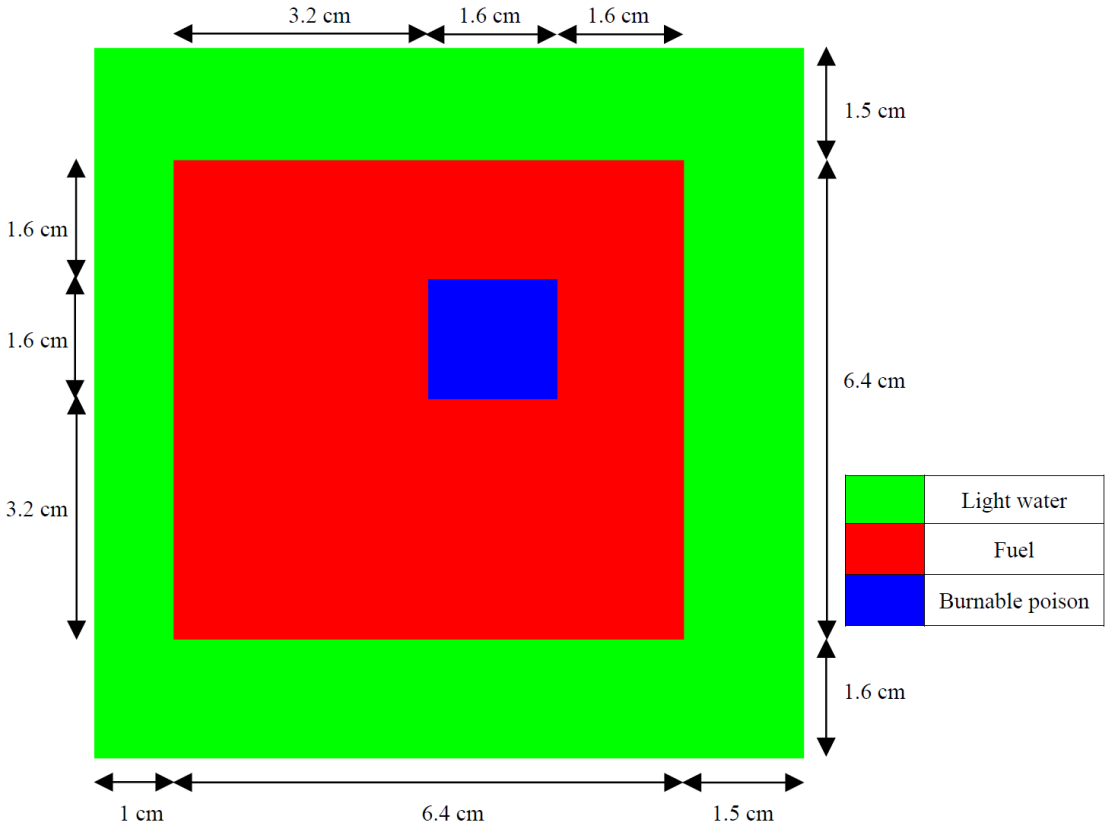

Fig. 2.9: LWR benchmark problem with burnable poison. Dimensions are shown in the figure. Representation of different materials with different colours is indicated in the legend table.

2.5.4 BWR benchmark problem with Gd pins

This two energy group, isotropic scattering benchmark problem is designed for a 4 × 4 BWR lattice which contains 14 fuel pins and 2 Gadolinium pins [18]. The fuel pins of radius 0.5 cm are made of 3 wt% UO_2 while the Gadolinium pins of size same with fuel are made of 3 wt% UO_2 and 3 wt% Gd_2O_3. All the pins are cladded with 0.1 cm thick Zircaloy-2. Light water is used as moderator. The lattice is shown in Fig. 2.10. Reflective boundary condition is applied on all four sides of the lattice. Fission spectrum and cross sections of UO_2, UO_2+Gd_2O_3, Zircaloy-2 and light water are given in Table-2.6.

.

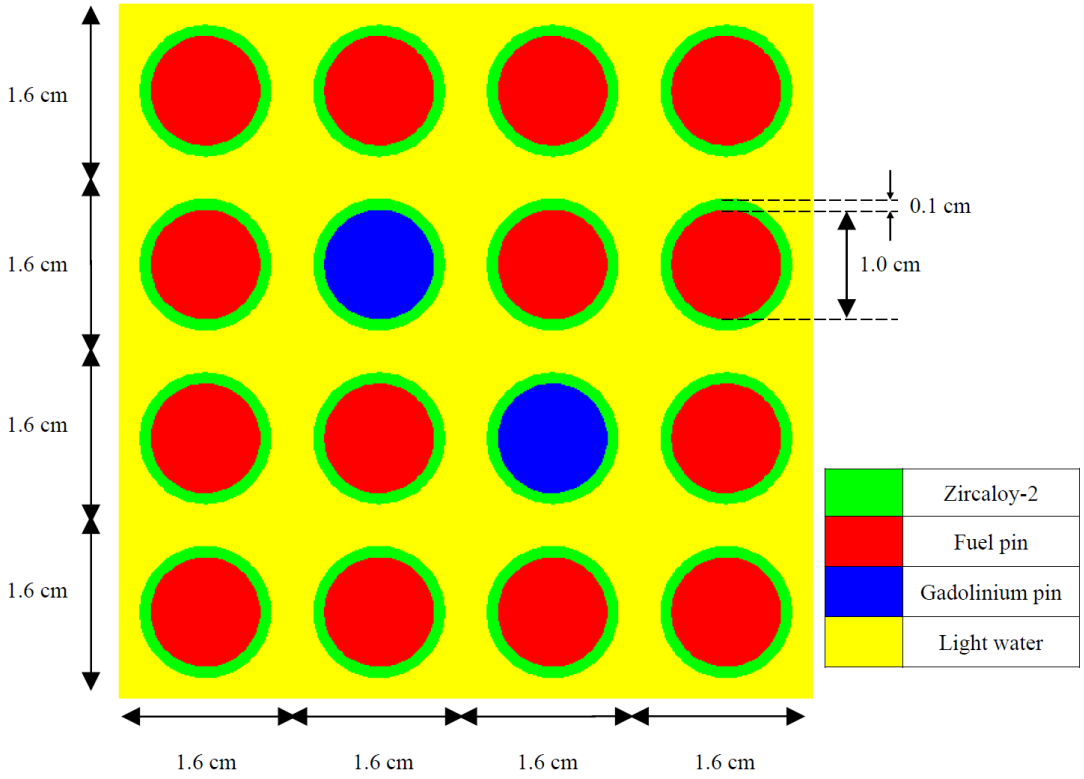

Fig. 2.10: BWR benchmark problem with Gd pins. Dimensions are shown in the figure. Representation of different materials with different colours is indicated in the legend table.

Table- 2.6: Cross sections (cm^{-1}) of BWR benchmark problem with Gd pins (χ: 1.0, 0.0)

Material	g	Σ_f	$\nu\Sigma_f$	Σ_t	$\Sigma_{sg\to1}$	$\Sigma_{sg\to2}$
Fuel pin	1	7.22964E-3	1.86278E-2	3.62022E-1	3.33748E-1	6.64881E-4
	2	1.41126E-1	3.44137E-1	5.72155E-1	0.0	3.80898E-1
Zircaloy-2	1	0.0	0.0	2.74144E-1	2.72377E-1	1.90838E-4
	2	0.0	0.0	2.80890E-1	0.0	2.77230E-1
Gd pin	1	6.97904E-3	1.79336E-2	3.71785E-1	3.38096E-1	6.92807E-4
	2	6.47524E-2	1.57929E-1	1.75	0.0	3.83204E-1
Light water	1	0.0	0.0	6.40711E-1	6.07382E-1	3.31316E-2
	2	0.0	0.0	1.69131	0.0	1.68428

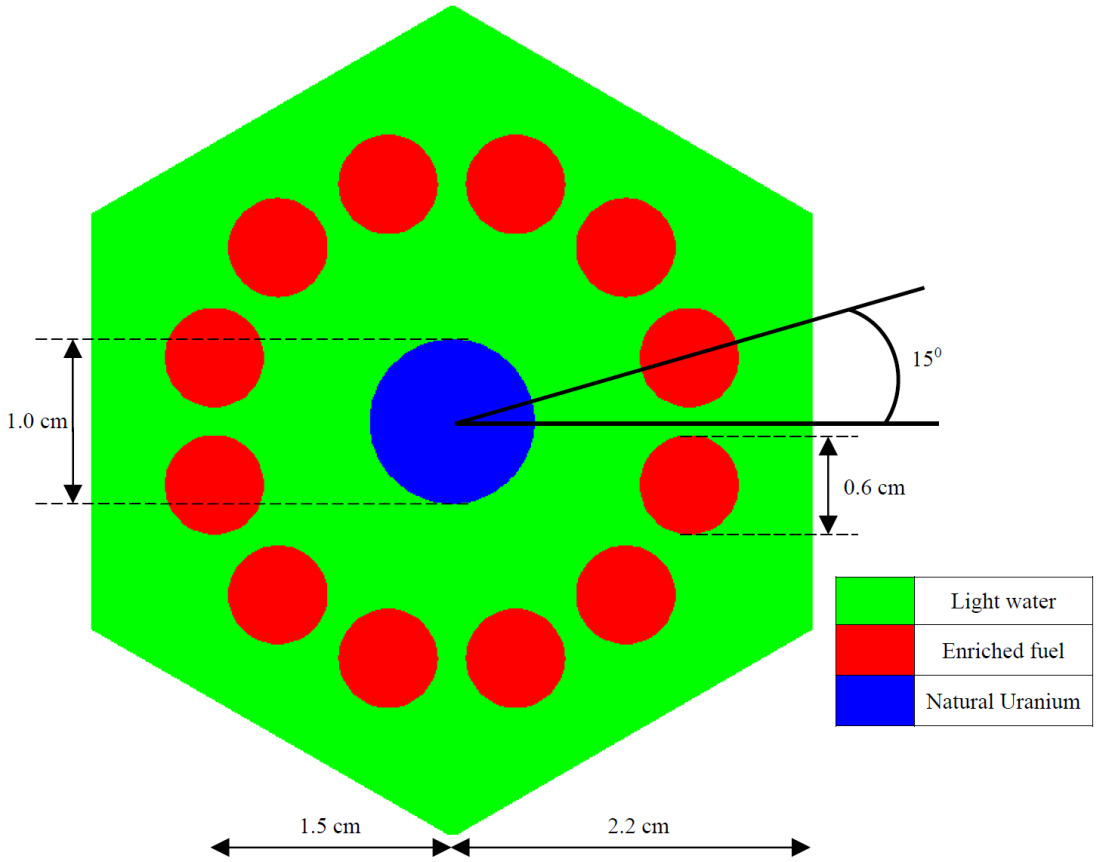

Fig. 2.11: Hexagonal cell problem with central breeding pin. Dimensions are shown in the figure. Representation of different materials with different colours is indicated in the legend table.

2.5.5 Hexagonal cell problem with central breeding pin

This problem is derived from a fixed source problem for a hexagonal cell with central breeding pin [61] by deleting the source term. Inside the hexagonal cell of pitch 4.4 cm filled with light water, there is a central natural uranium pin of radius 0.5 cm which is surrounded by 12 enriched fuel pins each having radius of 0.3 cm. These 12 enriched pins are placed symmetrically on a circle of radius 1.5 cm. The hexagonal cell is shown in Fig. 2.11. The scattering is isotropic and the cell boundary is reflective. Two group material cross sections and fission spectrum are given in Table- 2.7.

Table- 2.7: Cross sections (cm^{-1}) of Hexagonal cell problem with central breeding pin (χ: 1.0, 0.0)

Material	g	Σ_a	$\nu\Sigma_f$	Σ_t	$\Sigma_{sg\rightarrow1}$	$\Sigma_{sg\rightarrow2}$
Natural	1	8.627E-3	6.203E-3	1.96647E-1	1.78E-1	1.002E-2
Uranium	2	6.957E-2	1.101E-1	5.96159E-1	1.089E-3	5.255E-1
Enriched	1	8.627E-3	6.203E-3	1.96647E-1	1.78E-1	1.002E-2
fuel	2	6.957E-2	1.101E-1	5.96159E-1	1.089E-3	5.255E-1
Light	1	6.84E-4	0.0	2.22064E-1	1.995E-1	2.188E-2
water	2	8.016E-3	0.0	8.87874E-1	1.558E-3	8.783E-1

2.5.6 CANDU-6 annular cell benchmark problem

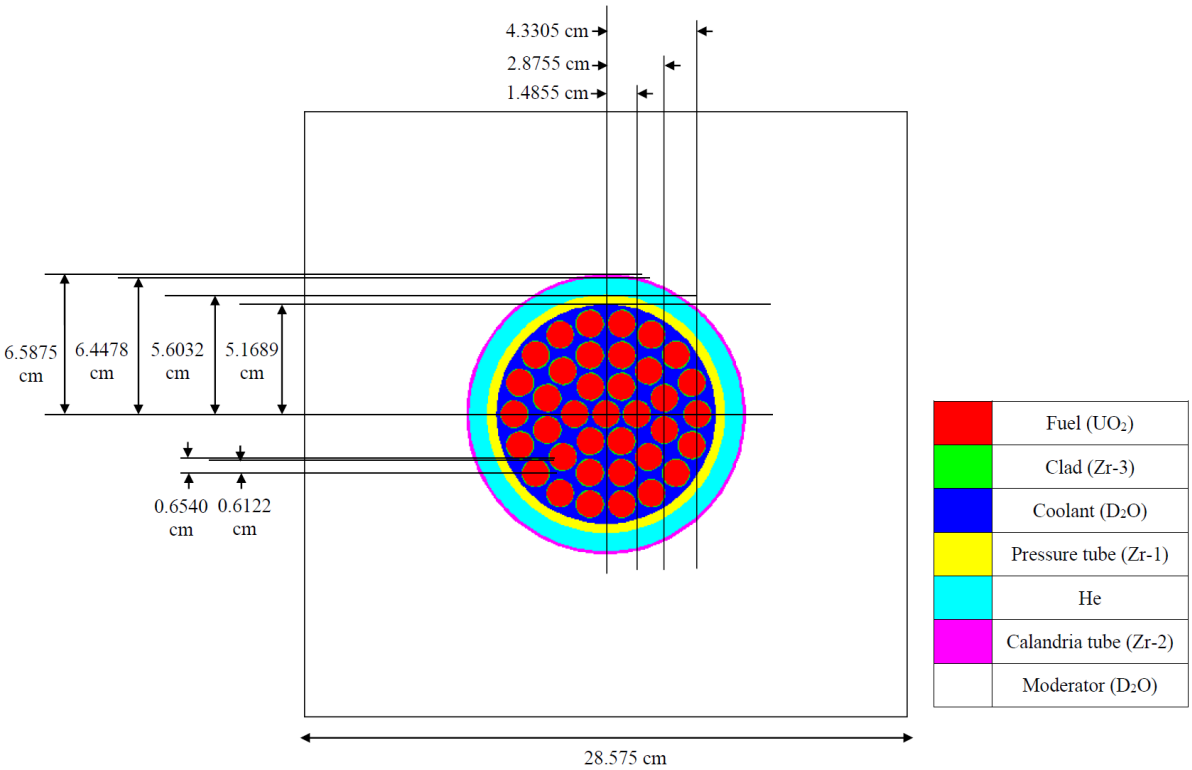

Fig. 2.12: CANDU-6 annular cell benchmark problem. Dimensions are shown in the figure. Representation of different materials with different colours is indicated in the legend table.

This benchmark problem is taken from the user manual of DRAGON 3.06 [21]. It is listed in the manual as “TCWU06” which describes a three dimensional supercell of a CANDU type reactor containing a horizontal fuel channel as well as a vertical stainless steel adjuster rod. Alternatively, the cell can be represented as a square lattice of side length 28.575 cm having 37 fuel pins arranged in three concentric rings with one

central pin. The radii of these UO_2 based fuel pins are 0.6122 cm which are clad with 0.0418 cm thick Zr-3. Fuel pins are cooled by D_2O. Apart from the central pin, the remaining pins are arranged in three rings of radii 1.4855 cm, 2.8755 cm and 4.3305 cm consist of 6, 12 and 18 pins respectively. All 37 pins along with the coolant are contained in a pressure tube made from Zr-1 and having an inner radius of 5.1689 cm and an outer radius of 5.6032 cm. This tube is surrounded by a Zr-2 calandria tube having an inner radius of 6.4478 cm and an outer radius of 6.5875 cm. For the purpose of insulation, the space between these two tubes is filled with helium. D_2O also acts as moderator and fills rest of the cell outside the calandria tube. The CANDU-6 cell is shown in Fig. 2.12. The scattering is isotropic and the boundary is reflective. Two energy group cross sections of all the above mentioned materials and the fission spectrum are given in Table- 2.8.

Table- 2.8: Cross sections (cm^{-1}) of CANDU-6 annular cell benchmark problem (χ: 1.0, 0.0)

Material	g	Σ_a	$\nu\Sigma_f$	Σ_t	$\Sigma_{sg\to1}$	$\Sigma_{sg\to2}$
Fuel (UO_2)	1	3.73334E-01	9.45470E-03	3.73334E-01	3.55414E-01	5.65522E-04
	2	5.05007E-01	1.36271E-01	5.05007E-01	0.00000E+00	4.00902E-01
Clad (Zr-3)	1	2.83817E-01	0.00000E+00	2.83817E-01	2.82283E-01	1.58650E-04
	2	2.80823E-01	0.00000E+00	2.80823E-01	0.00000E+00	2.75008E-01
Coolant (D_2O)	1	2.36057E-01	0.00000E+00	2.36057E-01	2.32571E-01	3.47825E-03
	2	3.46547E-01	0.00000E+00	3.46547E-01	0.00000E+00	3.46449E-01
Pressure tube (Zr-1)	1	3.02340E-01	0.00000E+00	3.02340E-01	2.99881E-01	2.33735E-04
	2	2.85758E-01	0.00000E+00	2.85758E-01	0.00000E+00	2.79187E-01
He	1	3.05524E-04	0.00000E+00	3.05524E-04	3.02747E-04	2.77636E-06
	2	1.71235E-04	0.00000E+00	1.71235E-04	0.00000E+00	1.71235E-04
Calandria tube (Zr-2)	1	3.04423E-01	0.00000E+00	3.04423E-01	3.02281E-01	2.62590E-04
	2	2.81233E-01	0.00000E+00	2.81233E-01	0.00000E+00	2.74710E-01
Moderator (D_2O)	1	3.33045E-01	0.00000E+00	3.33045E-01	3.20134E-01	1.29081E-02
	2	4.55438E-01	0.00000E+00	4.55438E-01	0.00000E+00	4.55395E-01

2.5.7 HTTR benchmark problem

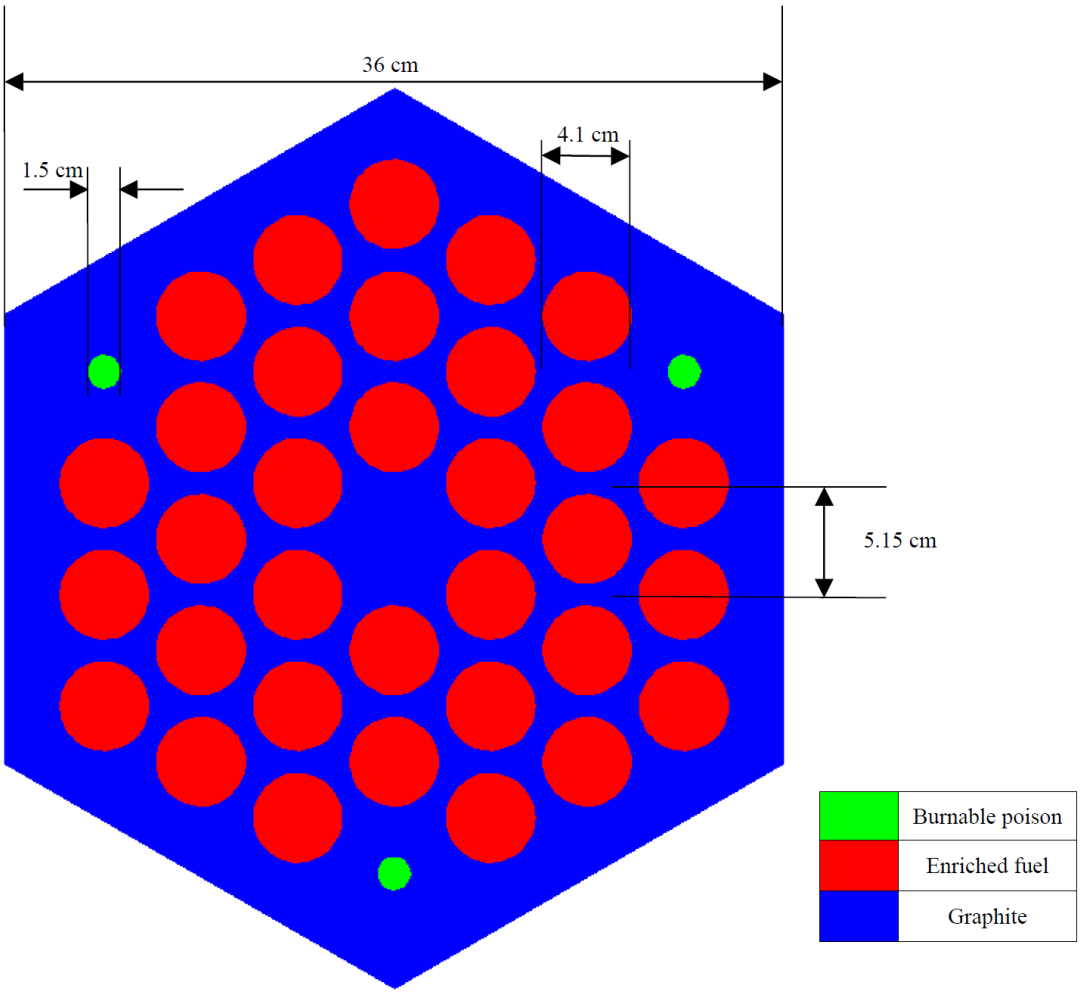

Fig. 2.13: HTTR benchmark problem. The dimensions are as shown in the figure. Representation of various materials with different colors is as indicated in the legend table.

In order to test the accuracy of transport method based codes in the calculation of Gen-IV reactors, number of heterogeneous prismatic benchmark problems, which are based on an experimental High Temperature Engineering Test Reactor (HTTR) built in Japan, were created by Zhang et al. [62]. This is mainly a six energy group whole core problem available in two and three dimension for various control rod position. Apart from this, a two dimensional benchmark problem, based on a single fuel block, is also available. The fuel block is a hexagon with 36 cm flat (edge)-to-flat (edge)

distance. It consists of 33 identical fuel pins and 3 burnable poison (BP) rods. Rest of the block is made of graphite. Diametre of each fuel pin is 4.1 cm and that of each BP rod is 1.5 cm. All the pins are arranged in a triangular lattice of pitch 5.15 cm. The geometry of the problem is shown in Fig. 2.13. There are 7 different fuel enrichments available though, for a single fuel block, the enrichment is same for all 33 pins. Case of 3.4 wt% enriched fuel is selected for the verification of MOC code since both multiplication factor and pin fission density distributions are available for the case. Scattering is isotropic and reflective boundary condition is applied at the outer boundary of the fuel block. The cross sections of the fuel, burnable poison, graphite and the fission spectrum are given in Table- 2.9 and Table- 2.10.

Table- 2.9: Fission, absorption and total cross sections (cm^{-1}) of HTTR benchmark problem
(χ: 0.969128, 0.0308724, 0.0, 0.0, 0.0, 0.0)

Material	g	Σ_f	Σ_a	$\nu\Sigma_f$	Σ_t
Fuel	1	1.45771E-04	2.15911E-04	3.99224E-04	1.40025E-01
	2	7.39241E-05	5.66989E-04	1.79819E-04	2.39531E-01
	3	6.41822E-04	7.52341E-03	1.56207E-03	2.55101E-01
	4	9.40389E-04	1.54287E-03	2.28872E-03	2.44455E-01
	5	3.76654E-03	5.17541E-03	9.16700E-03	2.51443E-01
	6	8.20805E-03	1.09730E-02	1.99765E-02	2.73904E-01
Burnable poison	1	0.00000E+00	6.47461E-05	0.00000E+00	7.08728E-02
	2	0.00000E+00	7.06662E-04	0.00000E+00	1.21259E-01
	3	0.00000E+00	1.19547E-02	0.00000E+00	1.36067E-01
	4	0.00000E+00	5.86182E-02	0.00000E+00	1.82689E-01
	5	0.00000E+00	1.31071E-01	0.00000E+00	2.56451E-01
	6	0.00000E+00	2.65951E-01	0.00000E+00	3.98509E-01
Graphite	1	0.00000E+00	4.61652E-07	0.00000E+00	2.24774E-01
	2	0.00000E+00	5.57902E-07	0.00000E+00	3.85038E-01
	3	0.00000E+00	9.84346E-06	0.00000E+00	3.97037E-01
	4	0.00000E+00	4.86718E-05	0.00000E+00	3.97831E-01
	5	0.00000E+00	1.09377E-04	0.00000E+00	4.03642E-01
	6	0.00000E+00	2.27789E-04	0.00000E+00	4.32667E-01

Table- 2.10: Scattering cross sections (cm^{-1}) of HTTR benchmark problem
(χ: 0.969128, 0.0308724, 0.0, 0.0, 0.0, 0.0)

Material	G	$\Sigma_{sg\to1}$	$\Sigma_{sg\to2}$	$\Sigma_{sg\to3}$	$\Sigma_{sg\to4}$	$\Sigma_{sg\to5}$	$\Sigma_{sg\to6}$
Fuel	1	1.30001E-01	9.80803E-03	6.76860E-09	0.00000E+00	0.00000E+00	0.00000E+00
	2	0.00000E+00	2.31833E-01	7.13067E-03	0.00000E+00	0.00000E+00	0.00000E+00
	3	0.00000E+00	0.00000E+00	2.42203E-01	5.37504E-03	0.00000E+00	0.00000E+00
	4	0.00000E+00	0.00000E+00	1.10955E-04	2.04442E-01	3.83572E-02	2.37433E-06
	5	0.00000E+00	0.00000E+00	0.00000E+00	2.39272E-03	2.15746E-01	2.81294E-02
	6	0.00000E+00	0.00000E+00	0.00000E+00	5.19254E-07	6.60101E-02	1.96920E-01
Burnable poison	1	6.51728E-02	5.63527E-03	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00
	2	0.00000E+00	1.16700E-01	3.85260E-03	0.00000E+00	0.00000E+00	0.00000E+00
	3	0.00000E+00	0.00000E+00	1.21376E-01	2.73639E-03	0.00000E+00	0.00000E+00
	4	0.00000E+00	0.00000E+00	3.48357E-05	1.05613E-01	1.84226E-02	6.94932E-07
	5	0.00000E+00	0.00000E+00	0.00000E+00	1.43604E-03	1.10302E-01	1.36420E-02
	6	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00	3.41636E-02	9.83944E-02
Graphite	1	2.07502E-01	1.72713E-02	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00
	2	0.00000E+00	3.72960E-01	1.20777E-02	0.00000E+00	0.00000E+00	0.00000E+00
	3	0.00000E+00	0.00000E+00	3.88199E-01	8.82826E-03	0.00000E+00	0.00000E+00
	4	0.00000E+00	0.00000E+00	8.61626E-05	3.37082E-01	6.06121E-02	2.27262E-06
	5	0.00000E+00	0.00000E+00	0.00000E+00	4.40537E-03	3.53948E-01	4.51795E-02
	6	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00	1.07248E-01	3.25191E-01

2.6 Benchmark problems with vacuum boundary condition

2.6.1 One energy group eigen-value problem

This is a one group, isotropic scattering problem given in [63]. Its outer boundary is a square of size 20 cm × 20 cm, inside of which five rectangles each having size of 1 cm × 18 cm are placed symmetrically with respect to the centre of the square as shown in Fig. 2.14. All five rectangular regions are filled with a material “F” and the remaining region of the square is filled with another material “M”. The cross sections of materials “F” and “M” are given in Table- 2.11. Vacuum boundary condition is applied at the outermost boundary.

Table- 2.11: Cross sections (cm^{-1}) of Mono energy group eigen-value problem
(χ: 1.0)

Material	Σ_f	$\nu\Sigma_f$	Σ_t	$\Sigma_{s1\to1}$
M	0.1	0.24	1.5	1.35
F	0.0	0.0	1.0	0.93

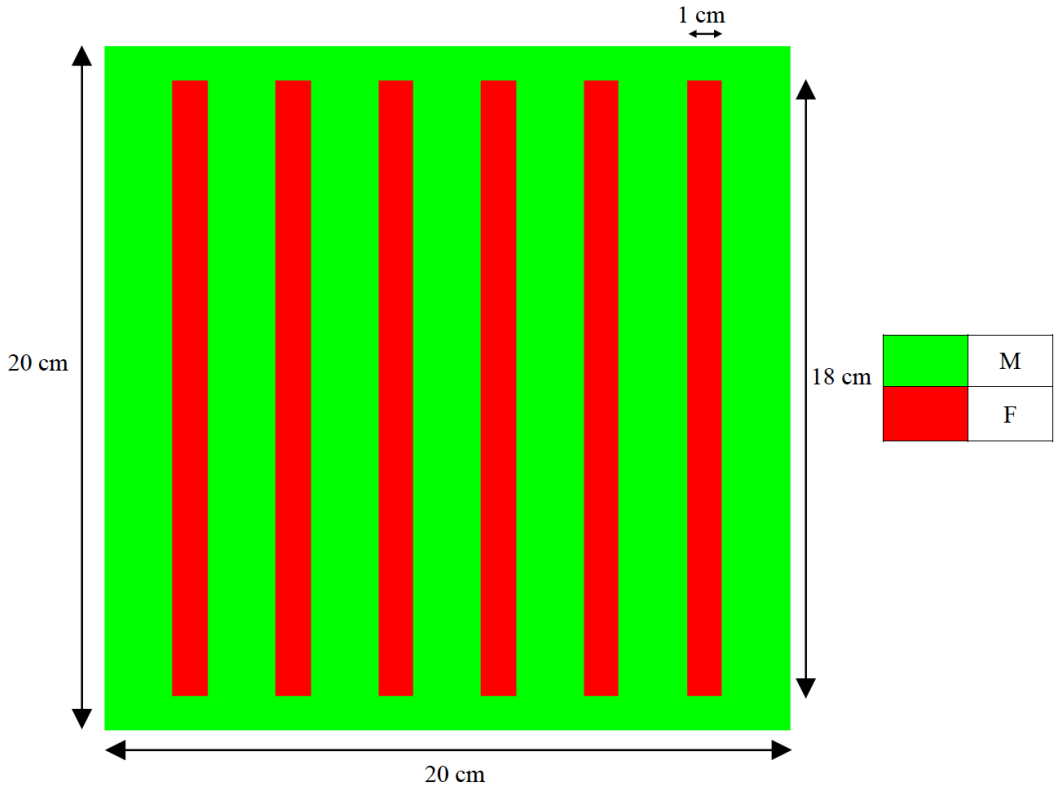

Fig. 2.14: Mono energy group eigen-value problem. Dimensions are shown in the figure. Representation of different materials with different colors is indicated in the legend table.

2.6.2 IAEA benchmark problem

The outer boundary of this problem is a rectangle of size 96 cm× 86 cm whose central region is occupied by 4 rectangular regions each having size of 30 cm × 25 cm. 4 materials are put into these 4 central rectangles and the region outside these rectangles is filled with another material. It is a one energy group, isotropic scattering, five-region problem of a swimming pool type reactor, defined by Stepanek et al. [25] for IAEA research program on transport theory and advanced reactor calculations. The geometry of the problem is shown in Fig. 2.15. The cross sections of five materials are given in Table- 2.12. Vacuum boundary condition is applied at the outer boundary.

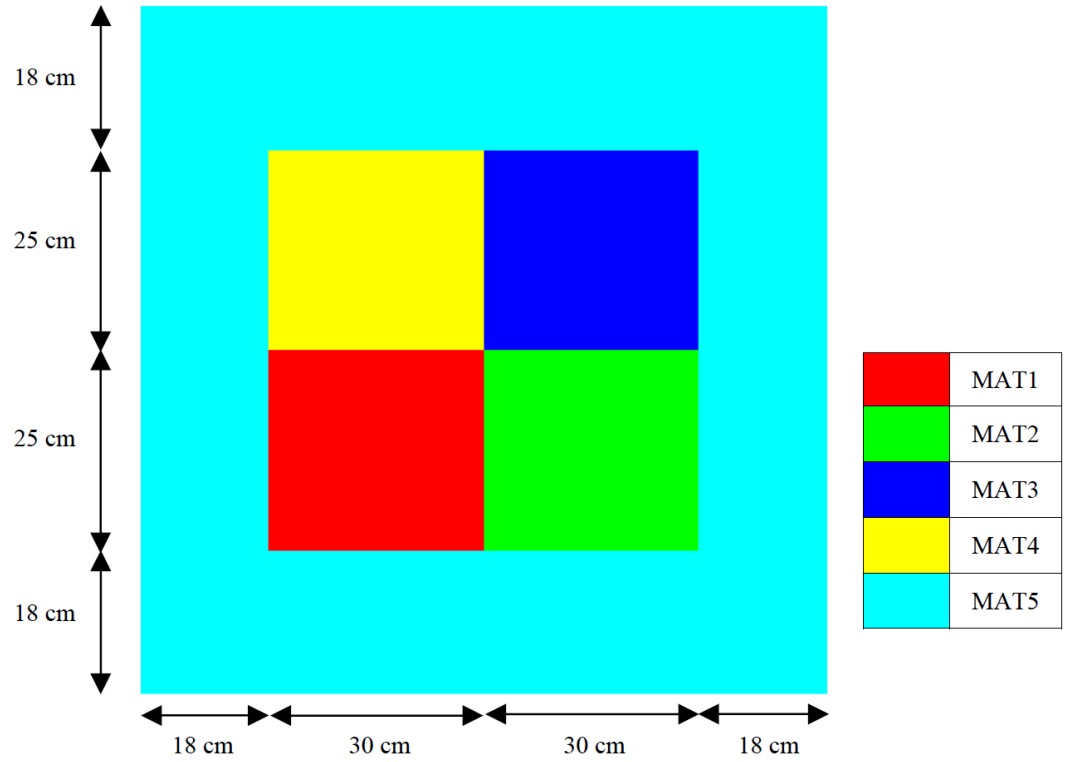

Fig. 2.15: IAEA benchmark problem. Dimensions are shown in the figure. Representation of different materials with different colors is indicated in the legend table.

Table- 2.12: Cross sections (cm^{-1}) of IAEA benchmark problem (χ: 1.0)

Material	Σ_a	$\nu\Sigma_f$	Σ_t	$\Sigma_{s1\rightarrow1}$
MAT1	0.07	0.079	0.60	0.53
MAT2	0.28	0.0	0.48	0.20
MAT3	0.04	0.043	0.70	0.66
MAT4	0.15	0.0	0.65	0.50
MAT5	0.01	0.0	0.90	0.89

2.6.3 Hexagonal assembly problem

It is a one energy group, isotropic scattering problem in hexagonal geometry [18]. Light water, fuel and control rod are three materials used in the problem. Fuel is present in a homogeneous hexagonal cell of side 3.2 cm excluding a central hexagon of side 0.4 cm where control rod is present and both the hexagons are placed concentrically in another hexagon of side 3.6 cm which is filled with light water as represented in Fig.

2.16. Cross sections of all three materials are given in Table- 2.13. Vacuum boundary condition is applied at the outer boundary.

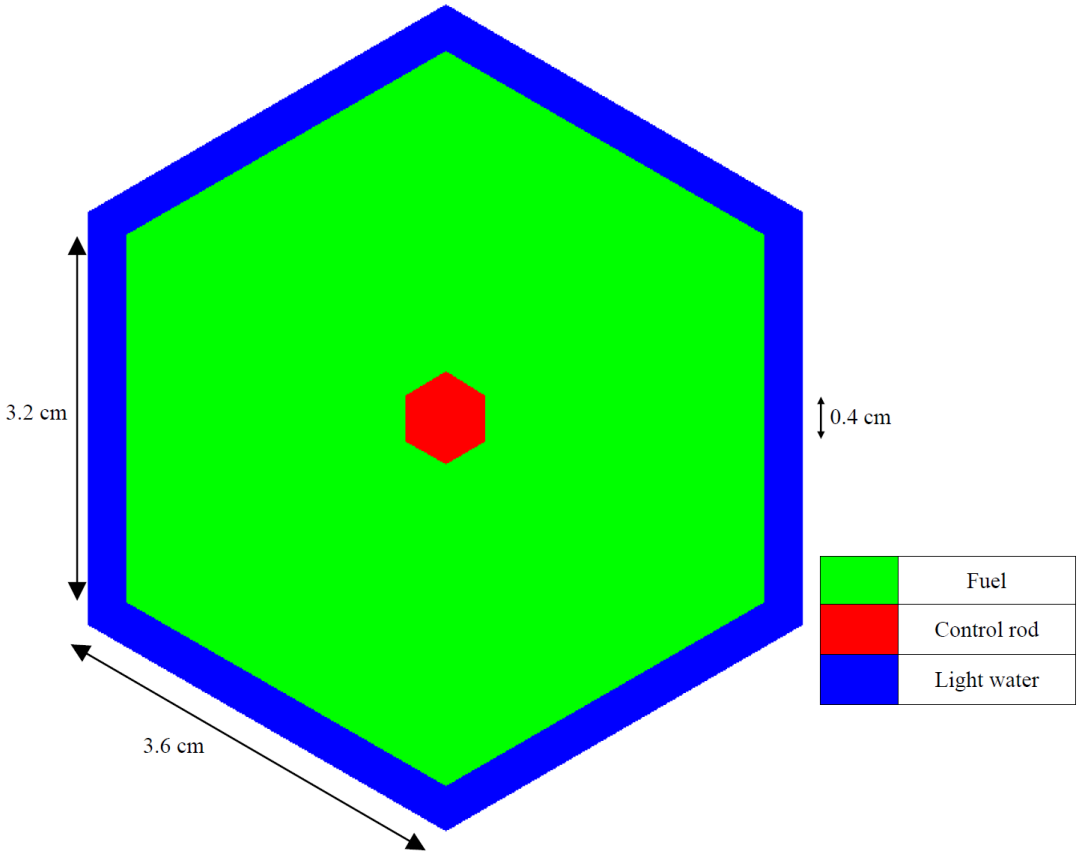

Fig. 2.16: Hexagonal assembly problem. Dimensions are shown in the figure. Representation of different materials with different colors is indicated in the legend table.

Table- 2.13: Cross sections (cm^{-1}) of Hexagonal assembly problem (χ: 1.0)

Material	$\nu\Sigma_f$	Σ_t	$\Sigma_{s1\rightarrow1}$
Fuel	0.17970	0.93480	0.83220
Control rod	0.0	1.2498	0.4658
Light water	0.0	1.32956	1.02093

2.6.4 MZA fast reactor benchmark problem

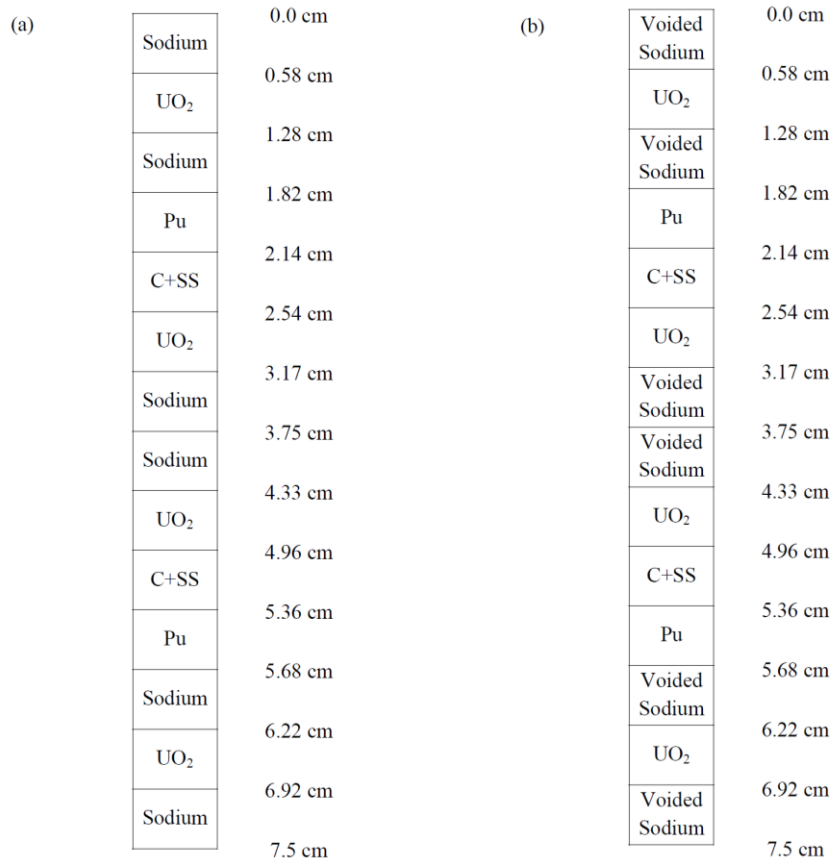

Fig. 2.17: Material distribution along the vertical axis of (a) unit fuel lattice and (b) unit voided fuel lattice in the MZA fast reactor benchmark problem.

This benchmark problem is based on a simplified description of the MZA fast reactor core [64][65]. The central region of the core is filled with fuel. It is surrounded by a blanket region and there is a reflector in the outermost region. The problem is defined in both two and three dimensions. Except for the boundaries of the respective regions, the problem description is identical in both cases. Four different core configurations are considered here. The first is the reference configuration (i.e. fuel-blanket-reflector, as described above) while the other configurations deal with void formation in the sodium coolant near the core centre and near the boundaries between the fuel and blanket regions. The unit cell of the fuel lattice [of size 5.4 cm × 5.4 cm] is

made up of six sodium plates, four uranium-dioxide plates, two plutonium plates and two carbon and steel plates, which are piled up axially up to a height of 7.5 cm (Fig. 2.17(a)). For representing the voided fuel lattice, sodium plates of unit fuel lattice are replaced by voided sodium plates, as shown in Fig. 2.17(b). Unlike the fuel region, the blanket and reflector lattices are treated homogeneously. It is only possible for three dimensional transport codes to simulate this core configuration exactly. For two dimensional transport codes, the homogenized cross sections of axially heterogeneous fuel unit cell (voided and without void) are used. The reference core consists of three concentric squares of size 90 cm × 90 cm (fuel), 165 cm × 165 cm (reflector) and 240 cm × 240 cm (blanket). In another core configuration, void formation takes place in a central region of size 60 cm × 60 cm. The other two cases are related with the cores where void is formed within an area of size 15 cm × 90 cm located at the horizontal and vertical boundaries between the fuel and blanket regions. Core configurations in all these four cases are shown in Fig. 2.18(a)-(d). Four energy group cross section data of fuel, voided fuel, blanket and reflector materials along with the fission spectrum is given in Table- 2.14. Anisotropic scattering is taken into account by the transport approximation.

Table- 2.14: Cross sections (cm^{-1}) of MZA fast reactor benchmark problem (χ: 0.88529, 0.11329 0.00142 0.0)

Material	g	Σ_a	$\nu\Sigma_f$	Σ_t	
Fuel	1	5.03700E-03	1.27400E-02	1.31000E-01	
	2	4.02600E-03	6.61700E-03	2.21000E-01	
	3	1.10700E-02	1.09600E-02	3.44300E-01	
	4	3.77400E-02	4.24200E-02	4.00700E-01	
Blanket	1	2.70700E-03	4.37200E-03	1.51300E-01	
	2	2.70300E-03	2.92500E-04	2.69000E-01	
	3	8.85400E-03	9.35000E-04	3.94500E-01	
	4	2.09000E-02	3.98400E-03	4.23100E-01	
Reflector	1	4.06000E-04	0.00000E+00	1.02000E-01	
	2	4.02000E-04	0.00000E+00	1.00000E-01	
	3	1.03100E-03	0.00000E+00	1.37800E-01	
	4	5.74700E-03	0.00000E+00	7.80700E-01	
Voided fuel	1	4.98600E-03	1.26100E-02	1.06400E-01	
	2	4.02300E-03	6.62400E-03	1.85100E-01	
	3	1.10300E-02	1.10000E-02	2.82900E-01	
	4	3.80000E-02	4.28500E-02	3.66900E-01	
Material	g	$\Sigma_{sg\to1}$	$\Sigma_{sg\to2}$	$\Sigma_{sg\to3}$	$\Sigma_{sg\to4}$
Fuel	1	1.09459E-01	1.64000E-02	1.04000E-04	0
	2	0.00000E+00	2.13668E-01	3.30600E-03	0
	3	0.00000E+00	0.00000E+00	3.32457E-01	7.73E-04
	4	0.00000E+00	0.00000E+00	0.00000E+00	3.63E-01
Blanket	1	1.23467E-01	2.49900E-02	1.36000E-04	0
	2	0.00000E+00	2.57705E-01	8.59200E-03	0
	3	0.00000E+00	0.00000E+00	3.81671E-01	3.98E-03
	4	0.00000E+00	0.00000E+00	0.00000E+00	4.02E-01
Reflector	1	8.82680E-02	1.31000E-02	2.26000E-04	0
	2	0.00000E+00	9.70540E-02	2.54400E-03	0
	3	0.00000E+00	0.00000E+00	1.33814E-01	2.96E-03
	4	0.00000E+00	0.00000E+00	0.00000E+00	7.75E-01
Voided fuel	1	8.76810E-02	1.36300E-02	1.03000E-04	0
	2	0.00000E+00	1.78424E-01	2.65300E-03	0
	3	0.00000E+00	0.00000E+00	2.71238E-01	6.32E-04
	4	0.00000E+00	0.00000E+00	0.00000E+00	3.29E-01

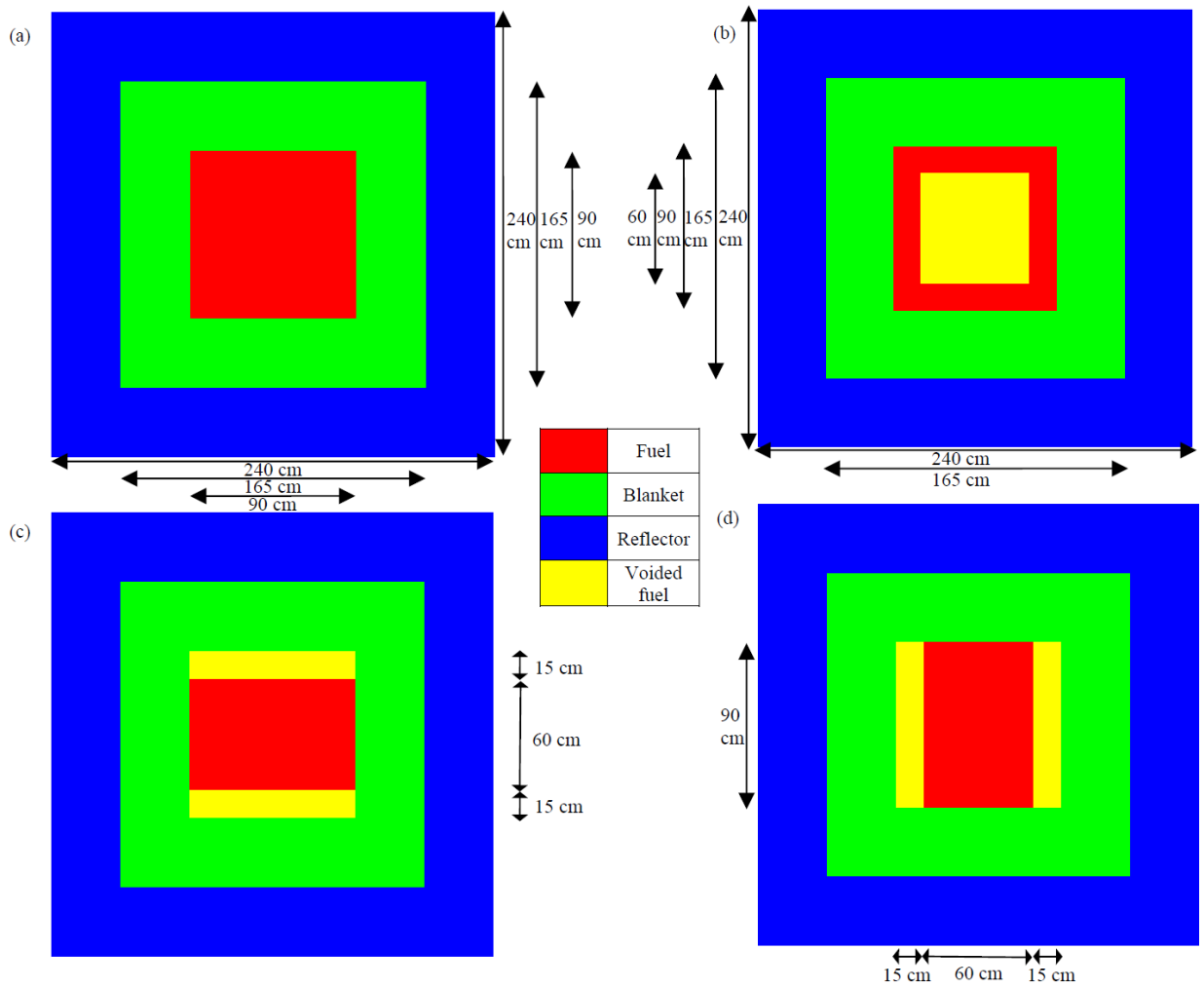

Fig. 2.18: MZA fast reactor benchmark problem in four cases: (a) Reference case; (b) void is formed at the centre of fuel region; (c) void is formed at the horizontal boundaries between the fuel and blanket regions, (d) void is formed at the vertical boundaries between the fuel and blanket regions. The dimensions are as shown in the figure. Representation of various materials with different colors is as indicated in the legend table.

2.6.5 KNK-II benchmark problem with non-reentrant boundary

The reactor core of this benchmark problem is that of a small Fast Breeder Reactor (FBR) with a hexagonal lattice which is a model of the KNK-II core [66]. It is originally a three dimensional problem. The core is composed of 169 homogeneous assemblies arranged in eight concentric hexagonal rings in such a way that each ring contains only one material. The length of each side of the hexagon is 7.5 cm. In order

to validate two dimensional transport codes, the problem was later redefined in two dimensions by taking a horizontal cross section of the core at the mid plane [22][67]. Vacuum boundary condition is applied at the outermost boundary which is "re-entrant" in nature (Fig. 2.19). Such type of surface allows the escaped neutrons to re-enter into the core. Transport codes like TWOHEX [68] and SPANDOM [67] include some portion of vacuum region in the problem domain to make the surface "non-reentrant". For codes like DIAMANT2 [69] etc., which are not developed to solve the transport equation in the presence of reentrant surface, the two dimensional problem is further modified by taking seven materials in seven concentric hexagons with parallel edges. As shown in Fig. 2.20, the perpendicular distance between the centre and the edge is 19.5 cm for the central hexagon and is 13 cm between the parallel edges of two adjacent hexagons. Four energy group cross sections of the materials and the fission spectrum are given in Table- 2.15 and Table- 2.16. The scattering is isotropic and vacuum boundary condition is applied at the outermost "non-reentrant" boundary.

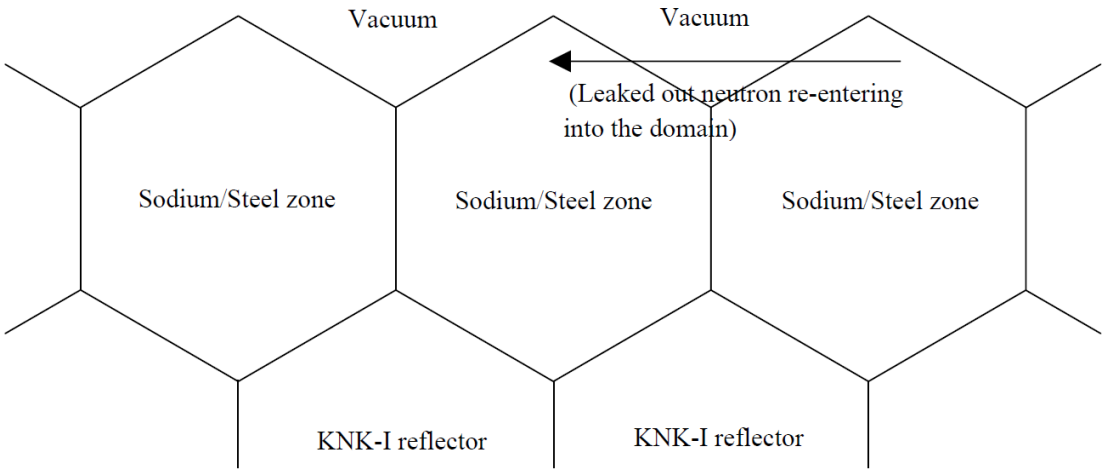

Fig. 2.19: Reentrant boundary of original KNK-II benchmark problem which allows leaked out neutrons to reenter the core.

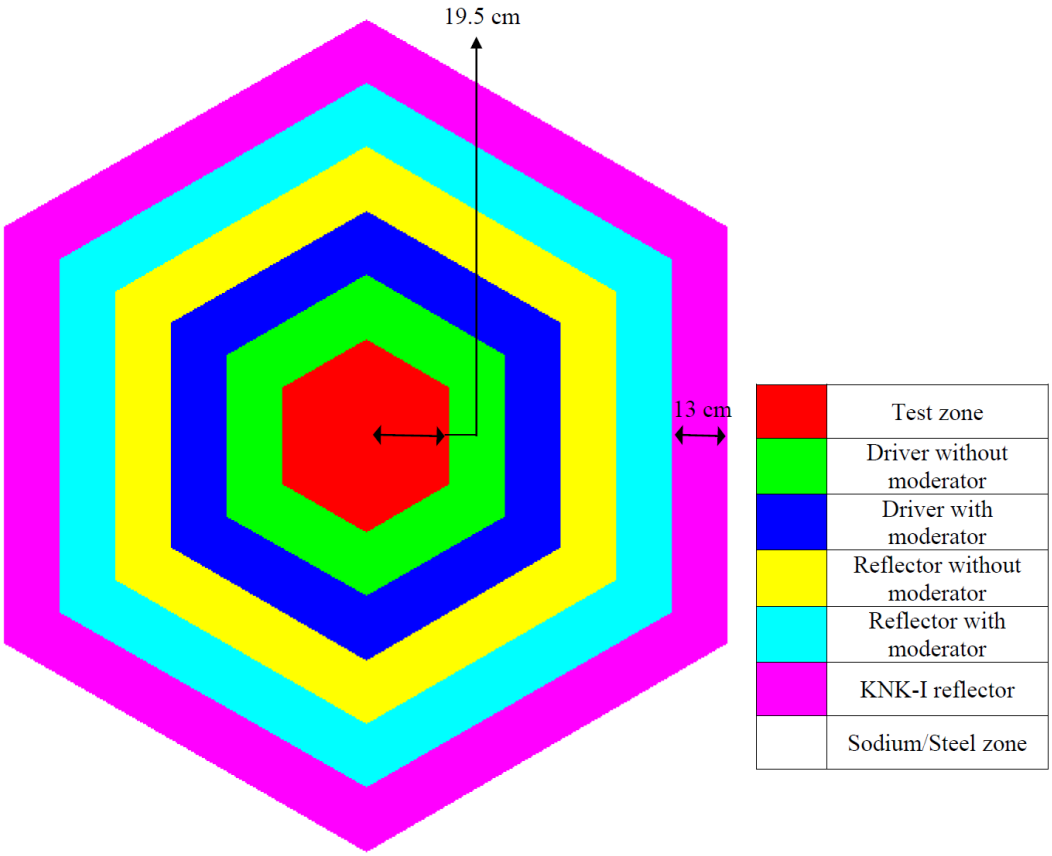

Fig. 2.20: KNK-II benchmark problem with a non-reentrant boundary. The dimensions are as shown in the figure. Representation of various materials with different colors is as indicated in the legend table.

Table- 2.15: Fission, absorption and total cross sections (cm^{-1}) of KNK-II benchmark problem with non-reentrant boundary
(χ: 0.908564, 0.087307, 0.004129, 0.0)

Material	g	Σ_a	$\nu\Sigma_f$	Σ_t
Test zone	1	7.14117E-03	1.79043E-02	1.24526E-01
	2	8.00576E-03	1.59961E-02	2.01025E-01
	3	1.45876E-02	2.40856E-02	2.86599E-01
	4	4.98120E-02	7.33104E-02	3.68772E-01
Driver without moderator	1	7.09892E-03	1.59878E-02	1.40226E-01
	2	9.02877E-03	1.64446E-02	2.28245E-01
	3	1.72478E-02	2.71451E-02	3.25806E-01
	4	5.74211E-02	8.45807E-02	4.18327E-01
Driver with moderator	1	4.67223E-03	1.01663E-02	1.41428E-01
	2	5.57965E-03	9.46359E-03	2.45394E-01
	3	1.32590E-02	1.87325E-02	3.98255E-01
	4	6.51184E-02	8.25335E-02	4.35990E-01
Reflector without moderator	1	4.64814E-04	0	1.59346E-01
	2	4.76496E-04	0	2.16355E-01
	3	1.23810E-03	0	3.48692E-01
	4	4.94323E-03	0	6.24243E-01
Reflector with moderator	1	3.97516E-04	0	1.39164E-01
	2	3.02674E-04	0	2.46993E-01
	3	1.22034E-03	0	4.52425E-01
	4	2.41527E-02	0	5.36256E-01
KNK-I reflec Tor	1	4.58692E-04	0	1.51644E-01
	2	4.59443E-04	0	1.42382E-01
	3	1.07883E-03	0	1.65132E-01
	4	5.91325E-03	0	8.04845E-01
Na /Steel zone	1	2.25039E-04	0	9.65097E-02
	2	2.33696E-04	0	9.87095E-02
	3	5.39303E-04	0	1.34200E-01
	4	3.03759E-03	0	4.12670E-01

Table- 2.16: Scattering cross sections (cm^{-1}) of KNK-II benchmark problem with non-reentrant boundary
(χ: 0.908564, 0.087307, 0.004129, 0.0)

Material	g	$\Sigma_{sg\to1}$	$\Sigma_{sg\to2}$	$\Sigma_{sg\to3}$	$\Sigma_{sg\to4}$
Test zone	1	1.05964E-01	1.12738E-02	1.46192E-04	9.62178E-07
	2	0	1.89370E-01	3.64847E-03	1.06888E-06
	3	0	0	2.70207E-01	1.80479E-03
	4	0	0	0	3.18960E-01
Driver without moderator	1	1.19887E-01	1.30790E-02	1.59938E-04	1.07166E-06
	2	0	2.15213E-01	4.00117E-03	1.82716E-06
	3	0	0	3.06885E-01	1.67341E-03
	4	0	0	0	3.60906E-01
Driver with moderator	1	1.14337E-01	2.09664E-02	1.39132E-03	6.10281E-05
	2	0	2.12006E-01	2.67269E-02	1.08186E-03
	3	0	0	3.52093E-01	3.29030E-02
	4	0	0	0	3.70872E-01
Reflector without moderator	1	1.47969E-01	1.06607E-02	2.49956E-04	1.82565E-06
	2	0	2.10410E-01	5.46711E-03	1.00157E-06
	3	0	0	3.42085E-01	5.36879E-03
	4	0	0	0	6.19306E-01
Reflector with moderator	1	1.05911E-01	2.96485E-02	3.06502E-03	1.41697E-04
	2	0	1.84820E-01	5.91780E-02	2.69229E-03
	3	0	0	3.73072E-01	7.81326E-02
	4	0	0	0	5.12103E-01
KNK-I reflec Tor	1	1.38427E-01	1.23901E-02	3.66930E-04	1.69036E-06
	2	0	1.37502E-01	4.41927E-03	1.63280E-06
	3	0	0	1.60722E-01	3.33075E-03
	4	0	0	0	7.98932E-01
Na /Steel zone	1	8.83550E-02	7.73409E-03	1.94719E-04	8.89615E-07
	2	0	9.52493E-02	3.22568E-03	7.98494E-07
	3	0	0	1.30756E-01	2.90481E-03
	4	0	0	0	4.09632E-01

CHAPTER 3

Solution of Neutron Transport Equation by Method of Characteristics

3.1 Introduction

Traditional reactor physics calculation follows two step process (lattice → core) for estimating the reactor core parameters. However, due to the phenomenal increase in computing power in recent time, there have been developments in computational methods for solving the transport equation directly in full reactor core. The Method of Characteristics or MOC seems to be promising in this respect due to its ability to (i) treat complex geometries commonly encountered in reactor cores, (ii) produce detailed flux and power distribution over the region of solution, (iii) handle anisotropic scattering. MOC is not only popular for lattice calculations but also for whole core calculations without homogenization.

In the present chapter, we describe the development of a computer code based on MOC to solve the neutron transport equation for mainly assembly level lattice calculation with reflective and periodic boundary conditions and to some extent core level calculation with vacuum boundary condition. The code is able to simulate square, circular, and hexagonal geometries and their combinations. Delaunay triangulation together with the Bower-Watson algorithm is used to divide the problem geometry into triangular meshes. Ray tracing technique is developed to draw characteristics lines along different directions over the geometry and the transport equation is solved over

these lines to obtain the neutron flux distribution and multiplication factor for the geometry. A number of benchmark problems available in literature are analyzed to demonstrate the capability and validity of the code.

3.2 Solution of Neutron Transport Equation by MOC

The neutron transport equation is an integro-differential equation which describes the distribution of neutron angular flux (Ψ) as a function of space (r), angle (Ω), energy (E) and time (t) in reactor core. Since our interest is in steady state (mostly "k" eigenvalue) problems, for the present purpose, the steady state neutron transport equation is written below.

$$\Omega.\vec{\nabla}\Psi_g\left(\vec{r},\Omega\right)+\Sigma_g^t\left(\vec{r}\right)\Psi_g\left(\vec{r},\Omega\right)=Q_g\left(\vec{r},\Omega\right) \tag{3.1}$$

g is energy group corresponding to energy E, Σ_g^t is g-th group total (absorption and scattering) macroscopic cross section and Q_g is g-th group total neutron source including fission source, scattering source and external source, if any. The source is related to the angular flux by the relations

$$Q_g\left(\vec{r},\Omega\right)=Q_g^{ext}\left(\vec{r},\Omega\right)+\sum_{g'=1}^{G}\int_{4\pi}\left(\chi_g\nu\Sigma_{fg'}\left(\vec{r}\right)+\Sigma_{sg'\to g}\left(\vec{r},\Omega'.\Omega\right)\right)\Psi_{g'}\left(\vec{r},\Omega'\right)d\Omega' \tag{3.2}$$

for the problem with an external source Q_g^{ext} and

$$Q_g\left(\vec{r},\Omega\right)=\sum_{g'=1}^{G}\int_{4\pi}\left(\frac{\chi_g\nu\Sigma_{fg'}\left(\vec{r}\right)}{k}+\Sigma_{sg'\to g}\left(\vec{r},\Omega'.\Omega\right)\right)\Psi_{g'}\left(\vec{r},\Omega'\right)d\Omega' \tag{3.3}$$

for the k eigenvalue problem. $\Sigma_{sg'\to g}$ is macroscopic scattering cross section from group g' to group g, χ_g is g-th group fission spectrum, ν is average number of neutrons released per fission and Σ_{fg} is g-th group macroscopic fission cross section. Applying the MOC, Eq.3.1 is converted into a linear ordinary differential equation [See section 2.3.3].

$$\frac{d}{ds}\Psi_g\left(\vec{r_0}+s\Omega,\Omega\right)+\Sigma_g^t\left(\vec{r_0}+s\Omega\right)\Psi_g\left(\vec{r_0}+s\Omega,\Omega\right)=Q_g\left(\vec{r_0}+s\Omega,\Omega\right) \qquad (3.4)$$

The characteristics in this case are straight lines - essentially the collision free flight paths of the neutrons - whose equations are given by $\vec{r}=\vec{r_0}+s\Omega$ where s is distance measured along Ω direction from $\vec{r_0}$ which is an arbitrary starting point on the characteristic line. By varying the coordinates of $\vec{r_0}$ a set of lines parallel to the direction vector Ω (in 3-dimensional space) is obtained. Changing Ω gives another set of parallel lines with a different orientation. Eq.3.4 can be solved along any of these lines, provided an initial value of the angular flux and the source distribution are known. The problem domain is divided into meshes having uniform material composition within each mesh. If we further assume that the flux variation within a mesh is small, we can take the source to be uniformly distributed (flat) within a mesh. It is then easy to write the following solution of Eq.3.4 for a mesh i and direction j

$$\Psi_{i,j,g}\left(s\right)=\Psi_{i,j,g}^{in}e^{-\Sigma_{i,g}^t s}+\frac{Q_{i,g}}{\Sigma_{i,g}^t}\left(1-e^{-\Sigma_{i,g}^t s}\right) \qquad (3.5)$$

where $Q_{i,g}$ is the flat source in mesh i and group g. This gives us the following equation for the outgoing angular flux i.e. the flux at the end of a segment intercepted by the mesh boundary.

$$\Psi_{i,j,g}^{out}=\Psi_{i,j,g}^{in}e^{-\Sigma_{i,g}^t \Delta t_{i,j}}+\frac{Q_{i,g}}{\Sigma_{i,g}^t}\left(1-e^{-\Sigma_{i,g}^t \Delta t_{i,j}}\right) \qquad (3.6)$$

$\Delta t_{i,j}$ is length of the segment along direction j, in mesh i as shown in Fig. 3.1. The average angular flux for the given characteristic in mesh i for the direction j is

$$\overline{\Psi}_{i,j,g} = \frac{Q_{i,g}}{\Sigma^t_{i,g}} + \frac{\Psi^{in}_{i,j,g} - \Psi^{out}_{i,j,g}}{\Sigma^t_{i,g}\Delta t_{i,j}} \tag{3.7}$$

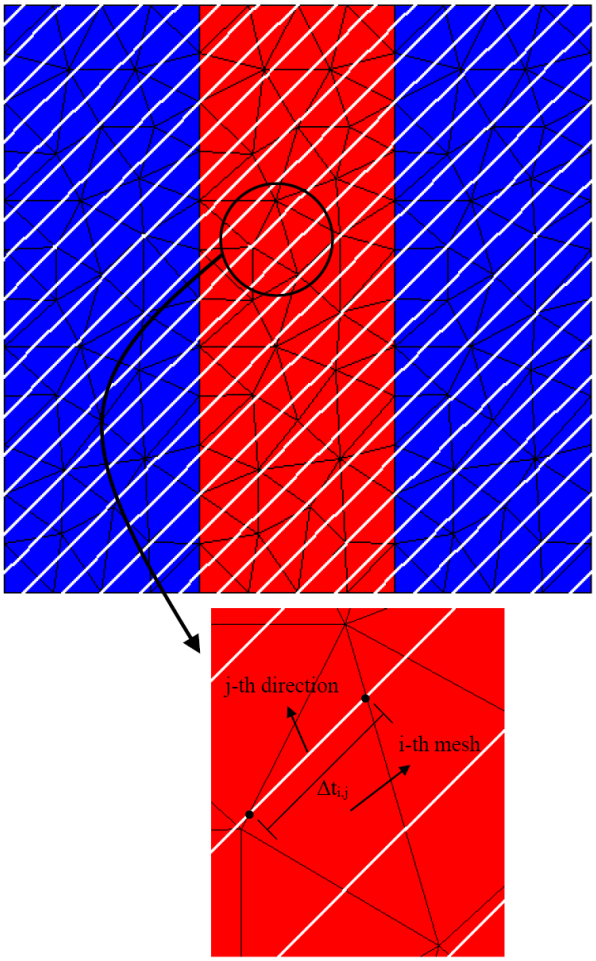

Fig. 3.1: A set of parallel lines are drawn on a triangulated rectangular slab. In zoomed view of a portion of the slab, it is seen that i th mesh is intersected by a straight line along j-th direction and $\Delta t_{i,j}$ is length of a segment of the line intercepted by the mesh boundary.

To calculate the source $Q_{i,g}$ as well as the reaction rates, the scalar flux and (in case of anisotropic scattering) other moments of the angular flux are required. This is obtained by first averaging the expression in Eq.3.7 over all characteristics parallel to the direction Ω and passing through the mesh i. This is the average angular flux for

direction j and mesh i. This must be then integrated over all directions to get the scalar flux.

In practice, we can obtain numerical values of the average angular flux only over a finite number of suitably chosen directions and over a finite number of characteristic lines in a given direction. The choice may be dictated by several considerations. For example, one may choose the directions from the well-known fully symmetric quadrature set used in the Sn method. Since our geometry is two dimensional, being uniform and infinite in the z direction, we choose a plane perpendicular to the z direction on which the lines are drawn. These lines are projections of the actual (3-D) characteristics. The orientation (θ_j, φ_j) of the characteristics is defined by a finite set of polar angles θ_j and azimuthal angles φ_j which is closed under reflection. Variation of the angular flux with the polar angle is smooth. So, θ_j-s' are chosen such that $\mu_j = \cos\theta_j$ are points of a Gaussian quadrature set. However, variation of angular flux with azimuthal angle is not as smooth as with polar angle [70]. Hence, φ_j-s' are distributed uniformly in the interval [0, 2π]. For a given φ_j, (all the projections corresponding to different θ_j are common) we use a set of equally spaced lines which form the projections of the characteristics. The intercepts are calculated for these projected lines on the plane. Then they are converted into actual intercepts of the characteristics by dividing by $\sin\theta_j$.

If n is index number of the parallel lines passing through i-th mesh along direction j, then an angular flux, averaged over all those lines with a weighting factor equals to the product of chord lengths ($\Delta t_{i,j}$) and separation between two consecutive parallel lines (Δw_j), can be defined as below.

$$\overline{\overline{\Psi}}_{i,j,g} = \frac{\sum_n \overline{\Psi}_{i,j,n,g} \Delta t_{i,j,n} \Delta w_{j,n}}{\sum_n \Delta t_{i,j,n} \Delta w_{j,n}} \tag{3.8}$$

The scalar flux is obtained by multiplying angular flux in Eq.3.8 by its angular weight for both θ_j and φ_j and then summing over all the directions.

$$\phi_{i,g} = \sum_j w_j \overline{\overline{\Psi}}_{i,j,g} \tag{3.9}$$

where w_j is the product of weights of the direction having polar angle θ_j ($w_{\theta j}$, obtained from Gauss quadrature set) and azimuthal angle φ_j ($w_{\varphi j}$). If there are total "m" number of φ_j directions considered, then $w_{\varphi j}=(2\pi/m)$. The average value of the scattering source and the fission source can be obtained from the average scalar flux obtained above, assuming the scattering to be isotropic in nature.

$$Q_{i,g}^{scattering} = \frac{1}{4\pi}\left(\sum_{g'=1}^{G} \Sigma_{sg'\to g} \phi_{i,g'}\right) \tag{3.10}$$

$$Q_{i,g}^{fission} = \frac{1}{4\pi}\left(\frac{\chi_g}{k}\sum_{g'=1}^{G} \nu\Sigma_{fg'} \phi_{i,g'}\right) \tag{3.11}$$

Thus the set of equations becomes closed.

From solution given in Eq.3.6 and average angular flux in Eq.3.7, multiplication factor (k) for a system and the flux distribution are calculated by carrying out a number of outer and inner iterations, which is described in brief in the following text. Before starting the iterations, some guess values are assumed for "k", incoming angular flux (Ψ_{in}) of all energy groups at boundary meshes and scalar flux (ϕ) of all energy groups in all meshes of the system.

The outer iteration starts by calculating fission sources, as defined in Eq.3.11, in all meshes and for all energy groups. It proceeds with the highest energy group for calculation of the flux in the group (by way of inner iterations) and moves down to the

next energy group and then the next till we reach the last group. Before the iteration moves to the next energy group we need the in-scatter source for the group. Using the sum of the fission and in-scatter sources, we then obtain the updated flux distribution for that group by the inner iterations. This is repeated for all the groups till the last group. Using the updated flux distribution in all the groups, the updated fission source is computed and thereafter a better estimate of k is obtained as follows

$$k^{new} = k^{old} \frac{\sum_{i,g} \nu\Sigma_{fi,g} \phi_{i,g}^{new} V_i}{\sum_{i,g} \nu\Sigma_{fi,g} \phi_{i,g}^{old} V_i} \tag{3.12}$$

V_i is volume of i-th mesh. This brings us to the end of the outer iteration. The outer iterations are halted when this k-value converges within a specified limit.

The inner iteration for a particular energy group g, involves calculating the angular flux in all meshes along different directions and scalar flux in all meshes. Inner iteration starts from a boundary mesh along a straight line n_j of direction j. Using Eq.3.6 with latest value of incoming angular flux at boundary mesh and scattering source, as defined in Eq.3.10 and added to fission source in Eq.3.11 to obtain total source for the mesh, outgoing angular flux of the mesh is calculated. The way angular flux is updated during inner iteration is similar to Gauss-Seidel method.

This outgoing angular flux is nothing but the incoming angular flux for next mesh along the same straight line. In a similar way, outgoing angular flux for the next mesh is also calculated and this calculation continues for all other meshes (i = 1 to $I(n_j)$) falling on the line until the line reaches boundary. Same recipe is followed for rest of the lines parallel to this line (n_j = 1 to N(j)) and subsequently for all other directions (j = 1 to J). Using Eq.3.7, average angular fluxes in all meshes are calculated for straight lines intersecting those meshes. Eq.3.8 is used to obtain the average angular flux in

mesh i along direction j. Finally, in mesh i, collecting such average values of angular fluxes along all directions and weighting them with proper weighting factor, the scalar flux (ϕ) is calculated using Eq.3.9. This way scalar fluxes in all other meshes are calculated. The inner iterations are terminated when the scalar flux converges within a specified error limit or the number of inner iterations per outer iteration reaches a specified upper limit (whichever is earlier). Such criterion is adopted for termination of inner iteration as there is no point to converge flux fully (i.e. up to the extent specified by flux convergence criterion) in initial rounds of iterations when the k value calculated is not accurate. Thus we set an upper limit on the number of inner iterations per outer iteration. Applying the limit, it is found that the flux converges partially for the initial outer iterations and as the "k" approaches towards its asymptotic value, the flux starts converging even before the number of inner iterations reach the specified upper limit. This strategy saves computation time which would be consumed otherwise to enforce full flux convergence in initial rounds of iteration. While it is known from experience that this strategy works well and speeds up the convergence, and is also supported by theoretical work valid for some specific situations [71], it might be difficult to provide a general proof.

During inner iteration, fission source, as calculated by Eq.3.11, is kept constant and is only updated once converged scalar flux value is obtained after the termination of inner iterations in all the groups, i.e. at the end of the outer iteration. A flowchart of solution technique of transport equation is given in Fig. 3.2.

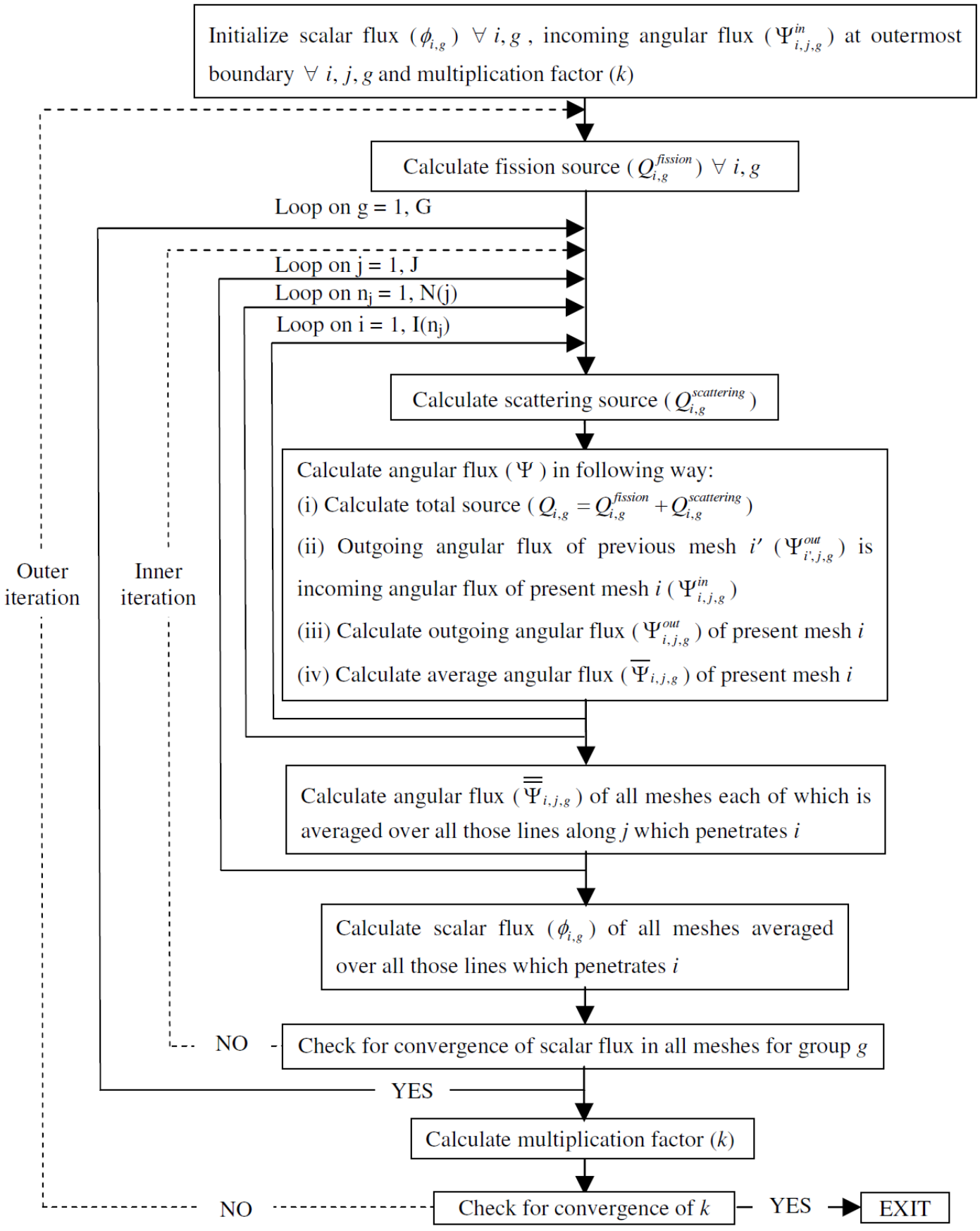

Fig. 3.2: Flowchart of MOC solution technique of neutron transport equation.

Reflective, periodic or vacuum boundary conditions may be used, depending upon the type of problem (Fig. 3.3). In principle, reflective boundary condition means outgoing angular flux [in a direction (θ, φ_i)] at a point r on the boundary being equal to incoming angular flux at the same point in a direction (θ, φ_r) where φ_r is the direction along which the incident line, after falling on the planar boundary, is reflected. Hence, the reflective boundary condition requires that (i) the set of directions chosen must be closed under reflection and (ii) the starting points of the characteristics in the incoming (reflected) direction should exactly coincide with the end points of the characteristics in the outgoing direction. While the former constraint can easily be satisfied by proper choice of angular directions, the latter is neither easily possible nor desirable for certain choices of directions which may be closed under reflection, however, may not cover the full space. Therefore, exact mirror like reflection of each characteristic line is not carried out to implement the reflective boundary condition. Rather the average value of outgoing angular flux for all those characteristics in the outgoing direction terminating at the side of a triangle at boundary is equated with the incoming angular flux for all the characteristics in the incoming direction starting from the same side of the triangle. For periodic boundary condition, the average value

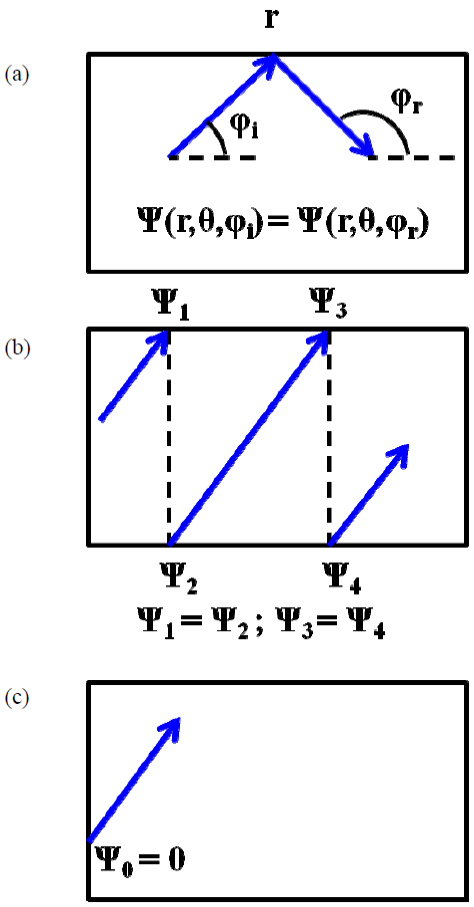

Fig. 3.3: Implementation of various boundary conditions in the MOC code: (a) Reflective, (b) Periodic and (c) Vacuum.

of outgoing angular flux for all those characteristics in the outgoing direction terminating at the side of a triangle at boundary is made equal to the average value of incoming angular flux for all those characteristics drawn in the same direction but starting from an equivalent side, as projected by a dotted line on the opposite side boundary in the figure. Vacuum boundary condition is treated by equating boundary values of all incoming angular fluxes to zero throughout the calculation.

3.3 Method of implementation of MOC in a computer code

The geometry of a reactor core or a fuel assembly is divided into a number of triangular meshes in such a manner that the material properties are uniform within each mesh. Further, each of the meshes must be small enough so that the assumption of a constant or flat source distribution made for solving the transport equation along a characteristic line over its intercept in the mesh, is valid. This is then followed by a selection of the characteristic lines along which the equation is to be solved and their construction by a ray tracing procedure. Finally, the transport equation is solved along the characteristics in an iterative fashion to obtain the flux distribution and the eigenvalue as described in the previous section. We describe the details of the procedures adopted by us for choosing the mesh and constructing the projections of the characteristics in this section.

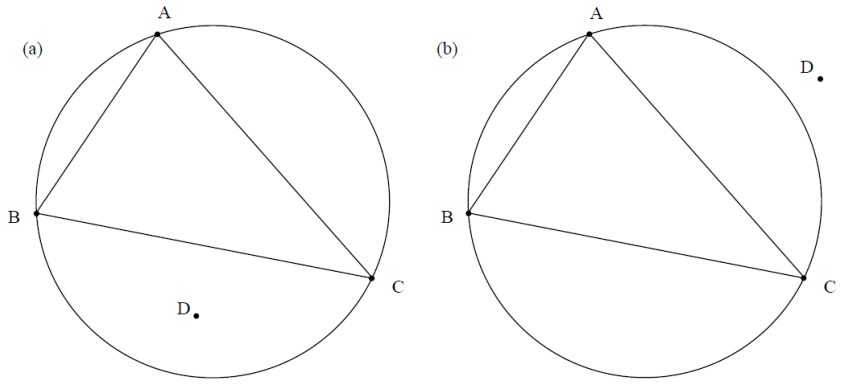

Fig. 3.4: (a) Delaunay triangulation criterion is not satisfied as point D resides within the circumcircle of triangle formed by three points A, B and C at its vertices. (b) Delaunay triangulation criterion is satisfied as point D is not within the circumcircle of triangle formed by three points A, B and C at its vertices.

3.3.1 Mesh generation

Since the geometry is uniform along the z direction, we consider only the two dimensional figure formed in a plane perpendicular to the z axis. In the MOC based computer code, mesh generation is done by a triangulation technique, introduced by S. Rebay [72]. This technique is generic in nature and applicable even for an irregular shaped geometry. It has the advantage that a fairly faithful representation of the geometry is possible without an excessively large number of meshes. Generally, the problem domain is divided into various homogeneous regions such as fuel, clad, moderator etc. which are triangulated one by one by this technique. As a first step of triangulation, a number of points are identified on the boundary of each of these regions. These points are then joined together to form triangles by Delaunay triangulation technique, as shown in Fig. 3.4, which guarantees that there exists no such triangle, formed by three points at its vertices, which contains another point within its circumcircle. After doing this initial triangulation step, further sub division into triangles is carried out as follows. Depending on the problem as well as the region to be

triangulated, different mesh sizes may be required. Here we will talk about mesh size in terms of the radius of the circumcircle of each triangular mesh. Let us define a function f(x, y) which gives the desired circumcircle radius at any point (x, y) in each region of the problem domain. In addition to that, a parameter is introduced which is defined at any point (x, y) as the ratio of circumcircle radius of a triangle, whose centroid is at (x, y), to the desired circumcircle radius at (x, y). Now, values of this parameter for triangles formed in initial triangulation are calculated and then the triangles are listed based on descending order of their parameter values. If the parameter of the triangle at top of the list is less than 1, then the triangulation will not be disturbed any more as desired mesh structure has already been reached. Otherwise, we continue further triangulation of the region by adding a new point at the circumcentre of the triangle at top of the list. The addition of this new point to the triangulated region violates the Delaunay triangulation criterion as the addition of a new point at the circumcentre of a triangle makes the point contained in circumcircle of the triangle and possibly in neighboring triangles also. Hence, this portion of the existing triangulation is reconstructed by Bowyer-Watson algorithm according to which, the triangles which fail to satisfy Delaunay triangulation criterion after insertion of the new point, are to be deleted and re-triangulation is to be done by connecting each vertex of convex hull, formed by this deletion, with the newly added point as shown in Fig. 3.5. After each re-triangulation, the list of triangles, already mentioned above, is updated since some triangles are deleted and some new triangles are formed by this time and again a point is added at the circumcentre of the triangle sitting at top of the updated list. This re-triangulation process is terminated when the parameter, as defined above, for all triangles comes within unity. Mesh size can be made variable by choosing suitable

functional form of f(x, y) across a region. The process is repeated for each of the regions. For a region lying on the opposite side of a boundary already triangulated, the same set of initial boundary points must be selected so that triangulation of one region is consistent with that of an adjoining region. The triangulation technique, discussed above, is unique in a sense that it is applicable for any geometry and not many researchers have reported such application in the field of neutron transport theory.

3.3.2 Ray tracing technique

In order to solve neutron transport equation by MOC, projections of the characteristic lines along different directions are to be constructed over the triangulated geometry. For a particular direction, there are number of parallel lines each having identical slope "m", but variable intercepts "c". "m" is different for different directions.

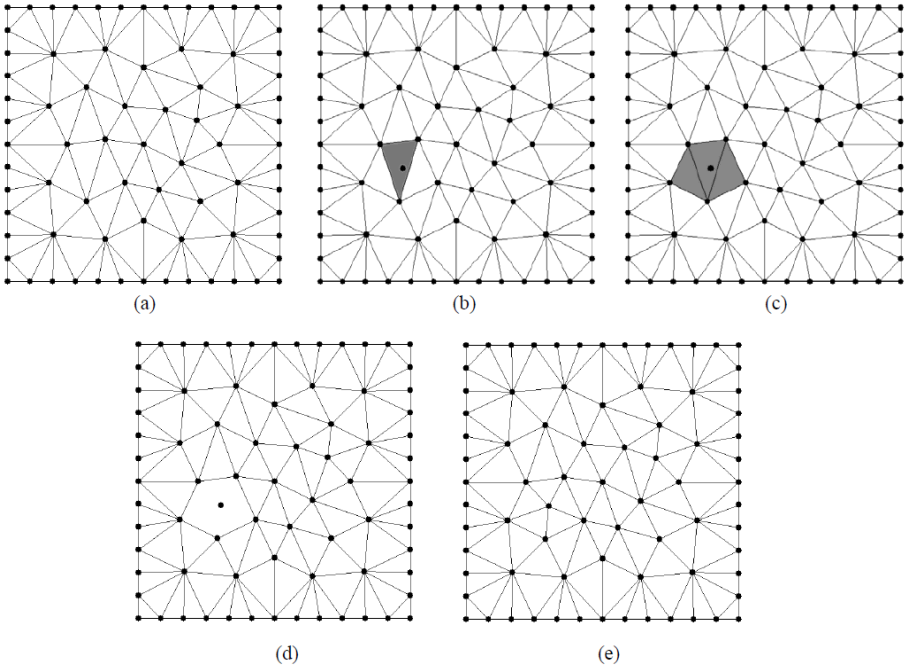

Fig. 3.5: Illustration of Bower-Watson algorithm: (a) Triangulation of a square. (b) Insertion of a new point at the circumcentre of a triangle (shaded). (c) Insertion of the point violates Delaunay criterion for three adjacent triangles (shaded). (d) Deletion of the three triangles forms a convex hull at their places. (e) Vertices of the convex hull are connected to the newly inserted point to form finer triangles. Applying the algorithm, the triangulation in (a) is converted to triangulation in (e).

The equation of each of these characteristic can be written as $y = mx + c$. Each triangular mesh in the geometry is characterized by coordinates of its three vertices, namely P (x_1, y_1), Q (x_2, y_2) and R (x_3, y_3). Using the coordinates of P and Q, equation of PQ side of the triangle can be constructed as $y = y_1 + \{(x-x_1)(y_2-y_1)\}/(x_2-x_1)$. Equations of other two sides, QR and PR, can be written in a similar way. The characteristics, while passing through the geometry, may or may not intersect the sides of a triangle depending on the equations of characteristics and the sides of the triangle. If both the characteristic and the side of a triangle have same slope, then they will not intersect each other while for different slope, there must be some point of intersection which can be found by solving the equations of both straight lines. However, a check is made whether the point of intersection lies on the segment forming the side of the triangle or not. Coordinates of all these points of intersection are arranged in increasing or decreasing order of their abscissa values according to the direction of characteristics. In addition to coordinates of points, triangle identification number, the length of the segment of the projection of the characteristic line intercepted by the triangle (Δs) and the material filled in the triangle are stored for subsequent flux calculations by the inner-outer iteration technique. The length of the intercept along the characteristic, used in Eq.3.6 or 3.7, is given by $\Delta t = \Delta s/\sin\theta$ where θ is the polar angle made by the characteristic line and positive z axis.

3.3.3 Description of the code

The MOC based computer code, written in FORTRAN90 language, is divided into six modules, namely GEOMCIRCLE, GEOMHEXAGON, GEOMSQUARE, INTERSECT, PLOT and TRANSPORT. In the first three modules, the mesh generation program is written for circular, hexagonal and square geometries respectively. INTERSECT module deals with ray tracing technique in the said geometries. DISLIN,

a high level plotting library [73], is used in the PLOT module to display the geometries using different colours for different materials. These five modules are called in module TRANSPORT to calculate k for a given system with the help of MOC. There are two input files for the program. “INPUT.txt” contains geometry specification, the θ and φ values of directions considered and their corresponding weights, number of parallel lines in a particular direction, number of energy groups, upper limit of number of inner iterations per outer iteration, boundary condition and convergence criterion for scalar flux and k value. Relevant cross sections of materials are supplied in another file called “CROSS_SECTION.txt”. The program output i.e. coordinates of vertices of all triangular meshes, material inside each of these meshes, chord lengths, sequence of triangles falling on each line, sides of triangles making the boundary, sides at starting and end point for each line are stored in “OUTPUT.txt” file. If repetitive program run is required (for carrying out burn up calculation at a later stage of the present work), then triangulation will be done only once and subsequently will be read from “OUTPUT.txt” file to reduce computation time. Gauss-Legendre quadrature set is used for θ values and their corresponding weights while φ values are chosen to comply with the imposed boundary condition and weights are chosen to be uniform for all φ values.

3.4 Verification and benchmarking of the Code

In order to verify the capability of the MOC based computer code to handle various geometries and to validate its k calculation, a number of benchmark problems in square as well as hexagonal geometries have been analyzed. Description and the input parameter required to analyze the benchmark problems are available in Chapter 2. Results obtained using the MOC code are given below.

Table- 3.1: Comparison between reference k_∞ values and the values calculated by the MOC based code for eighteen benchmark problems

No. of energy groups	Benchmark problem	Ref. k_∞ (k_1)	Calculated k_∞ (k_2)	Relative % difference $(\|k_1-k_2\|/ k_1)\times 100$
1	Pua-1-0-IN	2.612903	2.612903	0
	Pub-1-0-IN	2.290323	2.290322	4×10^{-5}
	Ua-1-0-IN	2.250000	2.250000	0
	Ub-1-0-IN	2.330917	2.330917	0
	Uc-1-0-IN	2.256083	2.256090	3×10^{-4}
	Ud-1-0-IN	2.232667	2.232665	9×10^{-5}
	UD2O-1-0-IN	1.133333	1.133333	0
	Ue-1-0-IN	2.1806667	2.1806665	9×10^{-6}
2	Pu-2-0-IN	2.683767	2.683767	0
	U-2-0-IN	2.216349	2.216349	0
	UAL-2-0-IN	2.662437	2.662436	4×10^{-5}
	URRa-2-0-IN	1.631452	1.631450	1×10^{-4}
	URRb-2-0-IN	1.365821	1.365818	2×10^{-4}
	URRc-2-0-IN	1.633380	1.633378	1×10^{-4}
	URRd-2-0-IN	1.034970	1.034972	2×10^{-4}
	UD2O-2-0-IN	1.000221	1.000220	1×10^{-4}
3	URR-3-0-IN	1.600000	1.600000	0
6	URR-6-0-IN	1.600000	1.600000	0

3.4.1 Analytical benchmark test set

Infinite multiplication factor (k_∞) is calculated for 18 benchmark problems given in section 2.5.1. In the first problem, the fissile material is Plutonium-a (Pua), the number of energy group is one (1), scattering is isotropic (0) and the problem is for an infinite medium (IN) (abreviated as Pua-1-0-IN). The meaning of abbreviations for other problems is in line with this. The cross sections of materials of all problems are given in Table- 2.3. In the MOC calculation, 6 point Gauss-Legendre quadrature set with weights 0.17, 0.36 and 0.47 corresponding to angles $\pm$ 21.18^0, $\pm$ 48.61^0 and $\pm$ 76.19^0 respectively is used for θ integration since this is adequate to approximate Bickley functions (resulting from θ integration in the collision probability method). 12 angles viz. 0^0, 30^0, 60^0, 90^0, 120^0, 150^0, 180^0, 210^0, 240^0, 270^0, 300^0 and 330^0, with

equal weights are used for φ integration. 200 parallel lines are considered for each (θ, φ) pair. Convergence criterion for neutron flux and k_∞-value are 10^{-5} and 10^{-7} respectively. Since the medium is infinite, any geometry of arbitrary dimension with reflective or periodic boundary condition can be used for the calculation. A square of 3 cm × 3 cm size with reflective boundary condition is used in the simulation. The k_∞ values calculated by MOC code are summarized in Table- 3.1 along with the reference values. The excellent agreement demonstrates that the overall conservation is satisfied.

3.4.2 One energy group eigen-value problem

We use this problem to demonstrate the effect of changing mesh size, number of characteristic lines per direction and number of azimuthal directions on the computation time and the accuracy of the computed k_{eff} value. In the present methodology, flat flux approximation is made which means we assume a constant flux within each of the triangles of the triangulated domain. For the approximation to be valid, size of the triangles should be sufficiently small. If we choose the triangle size bigger than the required value, then the assumption of constancy of flux will be violated since flux, in actual case, varies with distance.

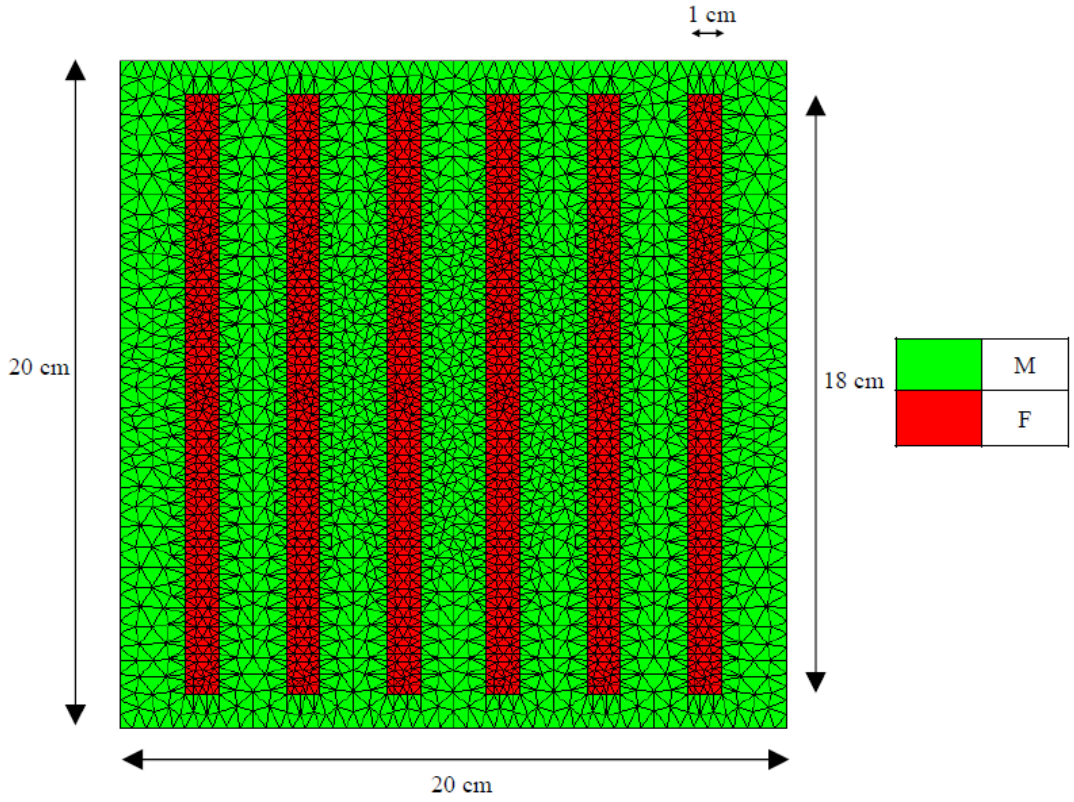

Fig. 3.6: Triangulated geometry of Mono energy group eigen-value problem, as generated by MOC code. Dimensions are shown in the figure. Representation of different materials with different colors is indicated in the legend table.

In this connection, Table- 3.2 demonstrates how the number of triangles, which is inversely proportional to the size of the triangles, affects the accuracy of k_{eff} value. As number of triangles increases, triangle size decreases and flat flux approximation becomes more and more appropriate which in turn improves the k_{eff} value. With number of triangles, though the computation time increases, number of outer iterations required for getting the converged k_{eff} value remains approximately same. Finally, we arrive at 7,060 number of triangles which is the required number for the present problem (Fig. 3.6). More number of triangles could have been used to improve the k_{eff} but at the cost of computation time (hence not shown in the table). In this calculation, other two parameters i.e. number of characteristic lines per direction and number of azimuthal

directions are kept 200 and 12, respectively. For generating the k_{eff} values of Table- 3.3, number of lines drawn in each direction is increased from 100 to 1000 taking 7,060 number of meshes and 12 azimuthal directions. While 100 lines per direction takes very less time to compute, the accuracy deteriorates and more number of iterations is required to get the converged estimate of k_{eff}. Beyond 200 lines per direction, there is not any appreciable change in both k_{eff} and number of iterations. However, the computation time increases four fold. Table- 3.4 shows the variation in the computed k_{eff} with the number of azimuthal angles chosen while the number of meshes and characteristic lines per direction are kept 7,060 and 200, respectively. As evident from the table, 12 azimuthal angles are sufficient for the present problem as there would not be much incentive in terms of the accuracy in k_{eff} and the number of iterations though the computation time increases with number of azimuthal directions. Therefore, it is concluded that the accuracy of results and computation time are more sensitive to the number of meshes, lines per direction and azimuthal directions than the number of iterations or the convergence of the iterative calculation. In this entire optimization study, 6 point Gauss-Legendre quadrature set is used for θ directions and their corresponding weights. Convergence criterion for neutron flux and k_{eff}-value are 10^{-5} and 10^{-7} respectively.

Table- 3.2: Effect of increasing the number of triangles for Mono energy group eigenvalue problem

No. of triangles	k_{eff}	Computation time (sec)	No. of outer iteration
882	0.7875718	11	52
1350	0.7911381	13	49
2948	0.7985724	20	50
3874	0.7988869	25	52
4348	0.8023327	32	51
5184	0.8019810	32	53
7060	0.8064641	39	56

Table- 3.3: Effect of increasing the number of lines per direction for Mono energy group eigenvalue problem

No. of lines/direction	k_{eff}	Computation time (sec)	No. of outer iteration
100	0.6382050	39	95
200	0.8064641	39	56
400	0.8025243	75	51
600	0.8039558	116	52
800	0.8024756	123	51
1000	0.8034197	162	52

Table- 3.4: Effect of increasing the number of φ values for Mono energy group eigenvalue problem

No. of φ values	k_{eff}	Computation time (sec)	No. of outer iteration
4	0.8594938	20	78
8	0.8118086	29	61
12	0.8064641	39	56
16	0.8042727	50	53
24	0.8027967	74	53

3.4.3 IAEA benchmark problem

The entire problem domain is divided into 23,478 triangular meshes (Fig. 3.7). 6 point Gauss-Legendre quadrature set is used for θ directions and their corresponding weights and 12 angles are used for φ directions with equal weighting factor. 500 parallel lines are considered for each direction. Convergence criterion for neutron flux and k_{eff}-value are 10^{-5} and 10^{-7} respectively.

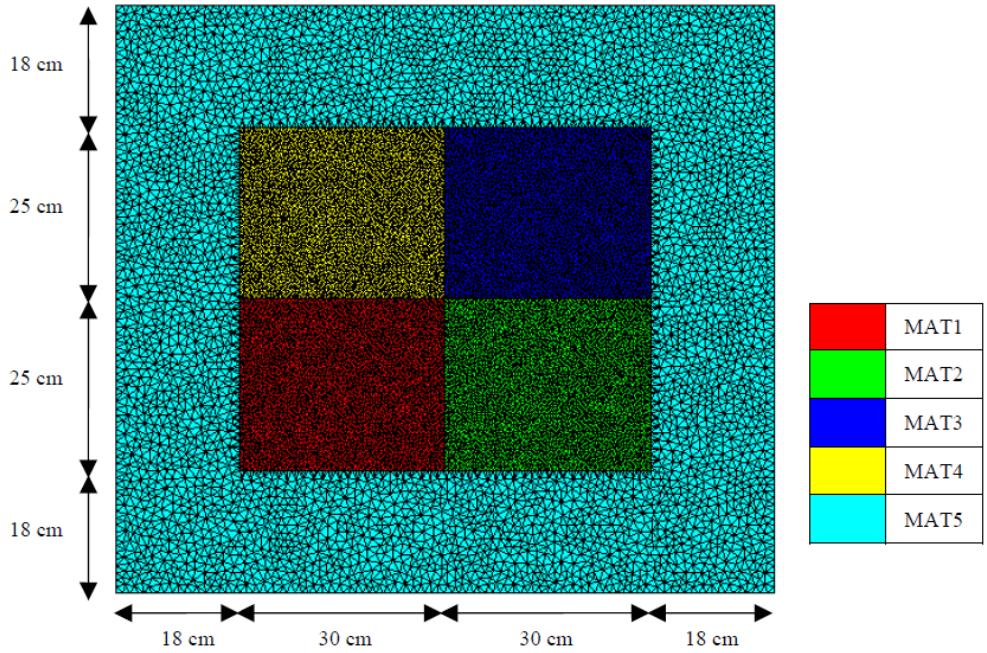

Fig. 3.7: Triangulated geometry of IAEA benchmark problem, as generated by MOC code. Dimensions are shown in the figure. Representation of different materials with different colors is indicated in the legend table.

3.4.4 BWR benchmark problem

The whole lattice is triangulated into 2,600 meshes (Fig. 3.8). 6 point Gauss-Legendre quadrature set for θ directions and their corresponding weights and 12 angles for φ directions with equal weighting factor are used for the calculation. Along each direction 200 parallel lines are drawn. Convergence criterion for neutron flux and k_∞-value are 10^{-5} and 10^{-7} respectively.

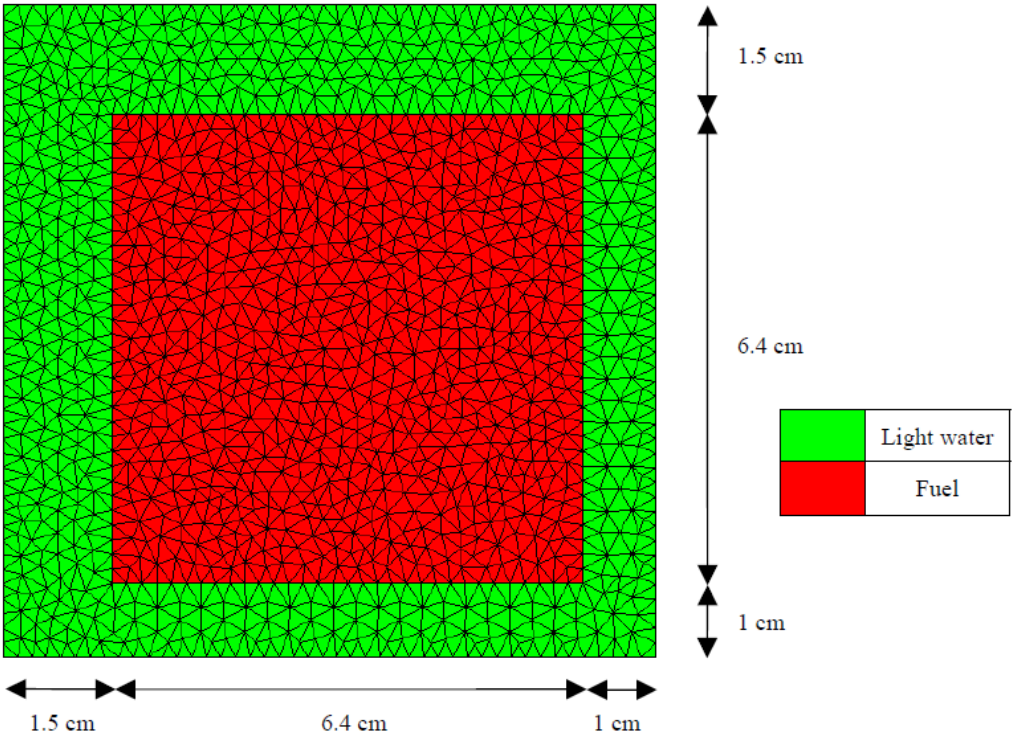

Fig. 3.8: Triangulated geometry of BWR benchmark problem, as generated by MOC code. Dimensions are shown in the figure. Representation of different materials with different colors is indicated in the legend table.

3.4.5 LWR benchmark problem with burnable poison

The entire lattice is triangulated into 8,160 meshes (Fig. 3.9). 6 point Gauss-Legendre quadrature set is used for θ directions and their corresponding weights while 12 angles are used for φ directions with equal weighting factor. 800 parallel lines are considered for each direction. Convergence criterion for neutron flux and k_∞-value are 10^{-5} and 10^{-7} respectively.

3.4.6 BWR benchmark problem with Gd pins

The entire lattice is divided into 7,260 triangles (Fig. 3.10). 6 point Gauss-Legendre quadrature set is used for θ directions and their corresponding weights, but

unlike previous cases 24 angles viz. 0^0, 15^0, 30^0, 45^0, 60^0, 75^0, 90^0, 105^0, 120^0, 135^0, 150^0, 165^0, 180^0, 195^0, 210^0, 225^0, 240^0, 255^0, 270^0, 285^0, 300^0, 315^0, 330^0 and 345^0 are used for φ directions with equal weighting factor as equal number of azimuthal angles are used for the solution obtained by DRAGON [21]. There are 500 parallel lines for each direction. Convergence criterion for neutron flux and k_∞-value are set at 10^{-5} and 10^{-7} respectively.

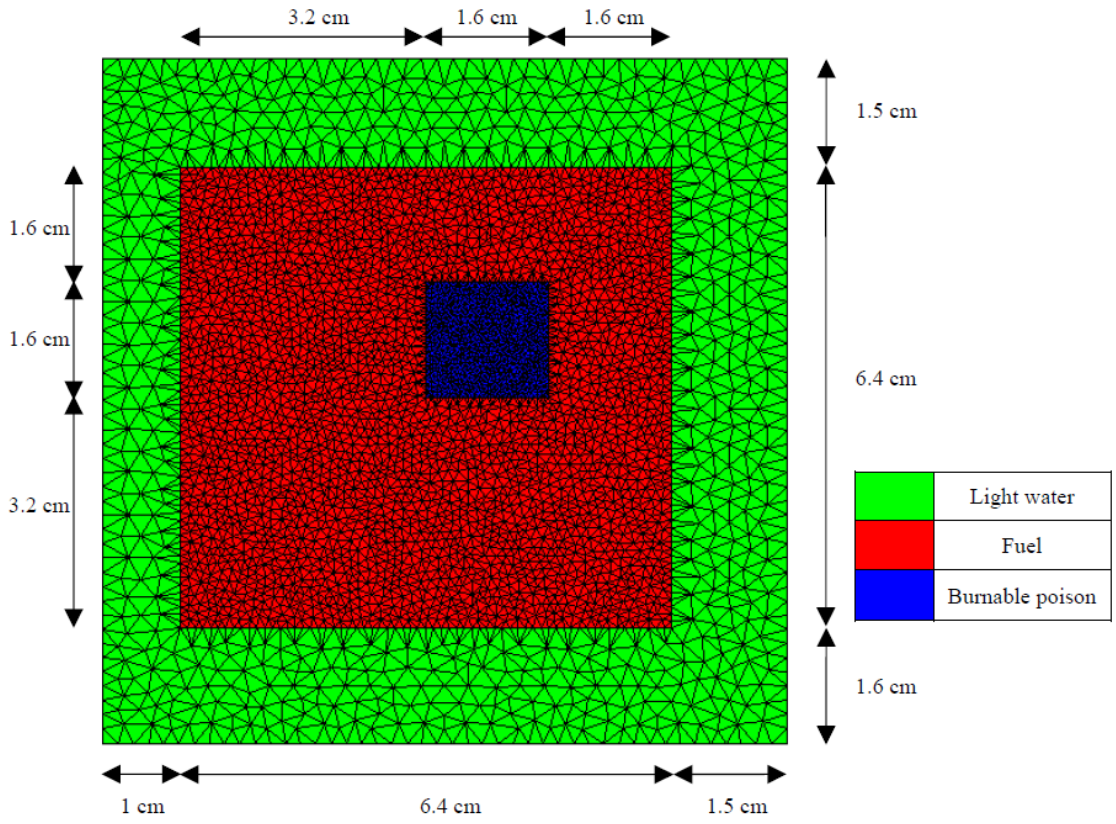

Fig. 3.9: Triangulated geometry of LWR benchmark problem with burnable poison, as generated by MOC code. Dimensions are shown in the figure. Representation of different materials with different colors is indicated in the legend table.

3.4.7 Hexagonal assembly problem

The hexagon is divided into 2,088 triangles (Fig. 3.11). 6 point Gauss-Legendre quadrature set is used for θ directions and their corresponding weights and 12 angles are used for φ directions with equal weighting factor. Transport equation is integrated

over 200 parallel lines in each direction. Convergence criterion for neutron flux and k_{eff}-value are set at 10^{-5} and 10^{-7} respectively.

3.4.8 Hexagonal cell problem with central breeding pin

The cell is divided into 4,148 triangles (Fig. 3.12). 6 point Gauss-Legendre quadrature set is used for θ directions and their corresponding weights and 12 angles are used for φ directions with equal weighting factor. A set of 200 parallel lines are considered for each direction. Convergence criterion for neutron flux and k_{∞}-value are 10^{-5} and 10^{-7} respectively.

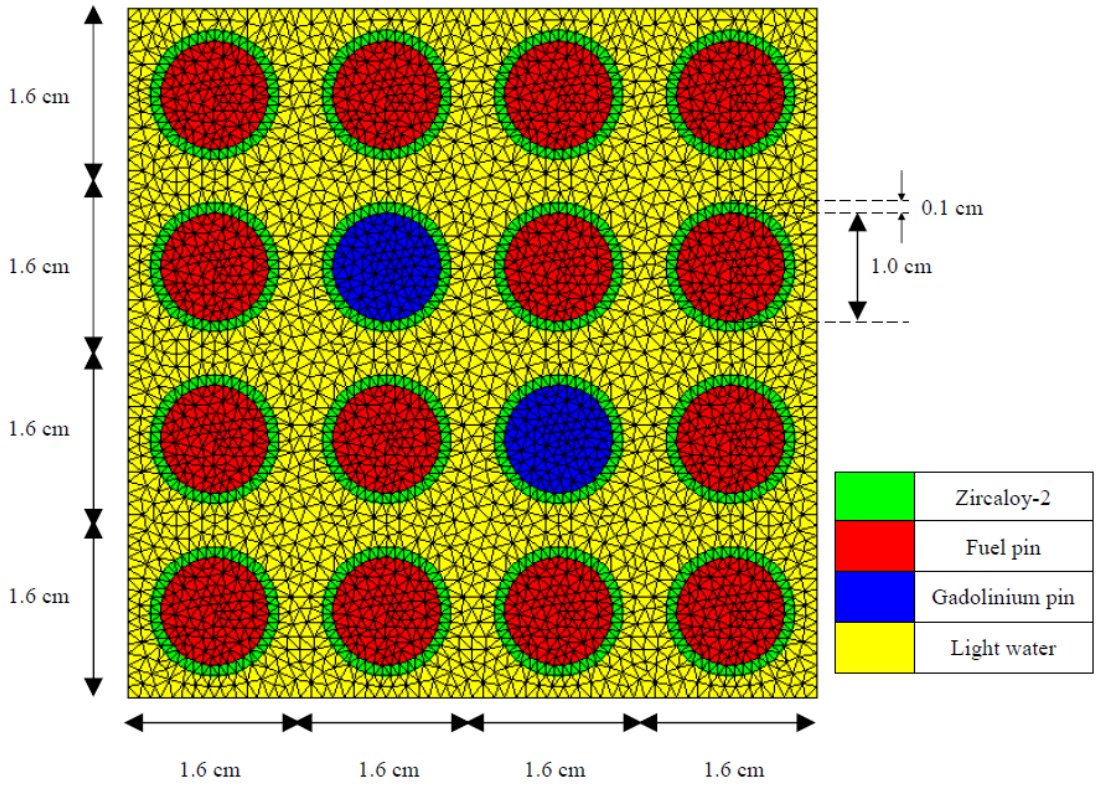

Fig. 3.10: Triangulated geometry of BWR benchmark problem with gadolinium pins, as generated by MOC code. Dimensions are shown in the figure. Representation of different materials with different colors is indicated in the legend table.

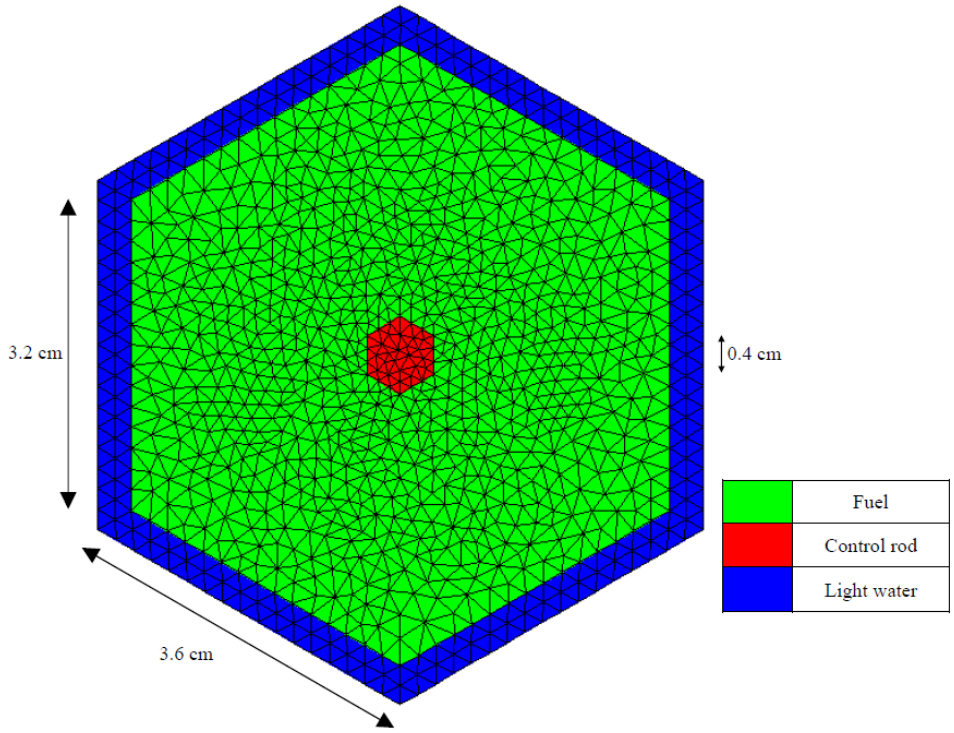

Fig. 3.11: Triangulated geometry of Hexagonal assembly problem, as generated by MOC code. Dimensions are shown in the figure. Representation of different materials with different colors is indicated in the legend table.

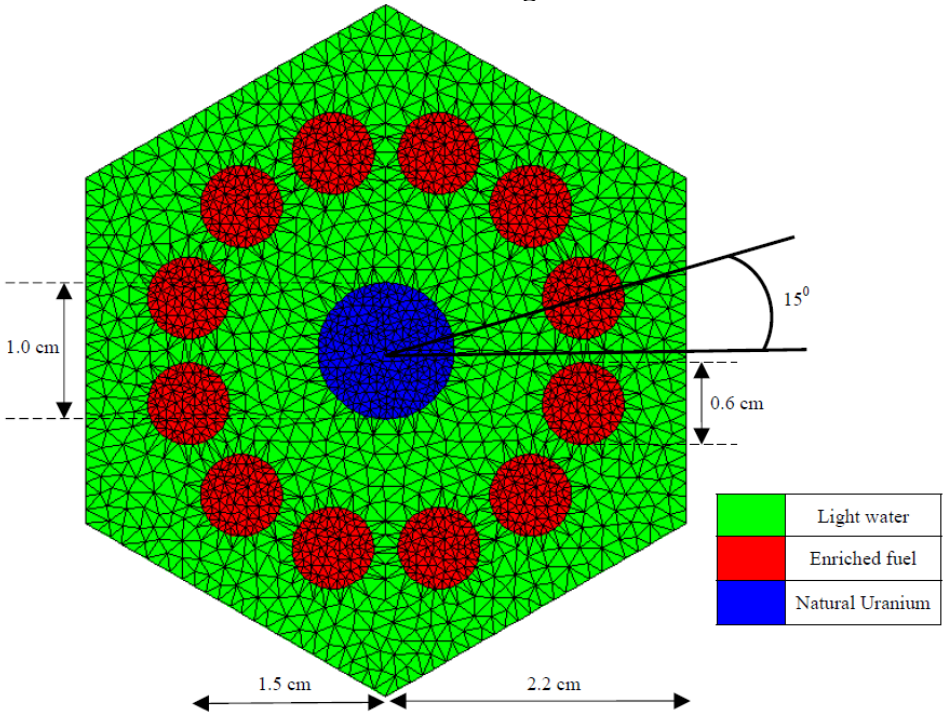

Fig. 3.12: Triangulated geometry of Hexagonal cell problem with central breeding pin, as generated by MOC code. Dimensions are shown in the figure. Representation of different materials with different colors is indicated in the legend table.

3.5 Results and discussion

The benchmark problems, discussed above, are solved using the MOC based code, described in section 3.3.3 and the results are compared in Table- 3.5 with the results of other codes available in literature. In one energy group eigen-value problem, k_{eff}-value calculated by MOC code agrees well with the simplified P_3 approximation by [63], a transmission probability method based code TPTRI [30] and a spherical harmonic method based code TEPFEM [74]. A good agreement is observed between k_{eff}-value of IAEA benchmark problem calculated by MOC code and other codes available in literature e.g. a neutron integral transport calculation code SURCU [25], a finite element code FELICIT [61], TEPFEM and TPTRI. The calculated k_∞ value of BWR benchmark problem is also in line with the results of SURCU, TEPFEM and TPTRI. In LWR benchmark problem with burnable poison, k_∞-value calculated by MOC code agrees with the results of TPTRI and SURCU. In another BWR benchmark problem (with Gd pins), result of MOC code is well comparable with the results of DRAGON and MOCUM [22]. For hexagonal assembly problem, k_{eff} value as obtained from MOC code exactly matches with the value obtained from CRX [18] code while it differs only by few mk from TWOHEX [68] code. In another hexagonal cell problem (with central breeding pin), k_∞ value calculated by MOC code is closer to Monte Carlo method based code MG-MCNP3B [75] in comparison with TPTRI and TEPFEM.

The table also shows the relative difference between the results obtained from the MOC code and one of the reference results for which the difference is maximum. It is seen that the MOC code agrees with the reference results generally within a fraction of a percent. However, the various reference results themselves have differences of about the same magnitude. Unlike the much simpler problems presented in Table- 3.1

where the errors were extremely low, the present problems involve complex geometry and hence the relatively larger differences. The much larger differences might also be due to the inadequacy of the spatial mesh or the number of characteristics used.

Table- 3.5: Comparison between k values calculated by other codes and the values calculated by the MOC based code for few benchmark problems

No. of energy groups	Benchmark problem	k value (k_1) calculated by other codes (Name of code)	k value (k_2) calculated by MOC code	Relative % difference ($\|k_1-k_2\|/k_1)\times100$*
1	Mono group eigen-value problem	0.798617 (SP_3) 0.803068 (TEPFEM) 0.806123 (TPTRI)	0.806464	1.0
1	IAEA	1.0069 (SURCU) 1.0069 (FELICIT) 1.0079 (TEPFEM) 1.0070 (TPTRI)	1.0044	0.3
2	BWR	1.2127 (SURCU) 1.2136 (TEPFEM) 1.2128 (TPTRI)	1.2119	0.1
2	LWR problem with burnable poison	0.8805 (SURCU) 0.8828 (TPTRI)	0.8790	0.4
2	BWR problem with gadolinium pins	0.986561 (DRAGON) 0.987785 (MOCUM)	0.9896832	0.3
1	Hexagonal assembly problem	0.7124 (TWOHEX) 0.7100 (CRX)	0.7100	0.3
2	Hexagonal cell problem with central breeding pin	1.090803 (MG-MCNP3B) 1.086598 (TEPFEM) 1.085775 (TPTRI)	1.088701	0.3

* Since, there are more than one k_1 value available for each problem, so maximum relative % difference is only quoted.

CHAPTER 4

Linear Representation of Source in Method of Characteristics

4.1 Introduction

A common assumption in the solution of the neutron transport equation by the Method of Characteristics or MOC is that the source (or flux) is constant within a mesh. This assumption is adequate provided the meshes are small enough so that the spatial variation of flux within a mesh may be ignored. Whether a mesh is small enough or not depends upon the flux gradient across a mesh, which in turn depends on factors like the presence of strong absorbers, localized sources or vacuum boundaries. The flat flux assumption often requires a very large number of meshes for solving the neutron transport equation with acceptable accuracy as was observed in earlier chapter. A significant reduction in the required number meshes is possible by using a higher order representation of the flux within a mesh.

In this chapter, we describe our work in which the source within a mesh is expanded up to first order (i.e. linear) terms. Thus, we permit the larger sized (and therefore fewer) meshes, which reduces the computation time without compromising the accuracy of calculation. Since the division of the geometry into meshes is through an automatic triangulation procedure using the Bowyer-Watson algorithm, representation of circular objects (e.g. cylindrical fuel rods) with coarse meshes is poorer and causes geometry related errors. A numerical recipe is presented to make a

correction to the automatic triangulation process and thereby eliminate this source of error. A number of benchmark problems are analyzed to emphasize the advantage of the source expansion method and the need to correct the triangular representation of the geometry.

4.2 Representation of source within coarse mesh in MOC

In the previous chapter, we have shown how the steady state neutron transport equation, which describes the distribution of neutron angular flux (Ψ) as a function of space (r), direction (Ω) and energy group (g) in reactor core, is converted into a linear ordinary differential equation (ODE), applying the MOC (Eq.3.4). This ODE governs the variation of the angular flux along a straight line characteristic, $\vec{r} = \vec{r_0} + s\Omega$, where s is the distance to the point ($\vec{r}$) measured from an arbitrary starting point ($\vec{r_0}$) on the characteristic line. For obtaining the numerical solution of the equation, we divide the problem domain into a number of meshes such that each mesh has a uniform material composition. If we make the additional assumption of a flat source within a mesh, then it is straightforward to write down the following solution of the ODE for a mesh i and direction j

$$\Psi_{i,j,g}^{out} = \Psi_{i,j,g}^{in} e^{-\Sigma_{i,g}^{t}\Delta t_{i,j}} + \frac{Q_{i,g}}{\Sigma_{i,g}^{t}}\left(1 - e^{-\Sigma_{i,g}^{t}\Delta t_{i,j}}\right) \tag{4.1}$$

where $Q_{i,g}$ is the flat source in mesh i and group g and $\Delta t_{i,j}$ is the "chord length" in mesh i, i.e. the length of the segment along direction j, in mesh i as shown in Fig. 3.1. The above equation gives us the outgoing angular flux i.e. the flux at the end of a segment intercepted by the mesh boundary.

However, if we divide the domain into coarser (large sized) meshes in order to reduce the computation time, then the flat source assumption will no more be valid as the variation of source within a mesh will be too large. Hence the solution cannot be given by Eq.4.1. For better representation of the spatial variation of the source within a mesh, we expand it in terms of some suitable basis functions. The polynomial functions are the simplest ones to choose in this regard. As a minimal improvement over the flat source assumption, we carry out an expansion up to the linear terms and write

$$Q_{i,j,g}(x,y) = a_0 + a_1 x + a_2 y \tag{4.2}$$

where a_0, a_1 and a_2 are the expansion coefficients. In mesh i, a local coordinate system, whose origin is set at the centroid (X_c, Y_c) of the mesh, is considered and x, y are the coordinates measured with respect to this local coordinate system. So, $x = X - X_c$ and $y = Y - Y_c$ where (X, Y) and (X_c, Y_c) are the coordinates measured with respect to a global coordinate system. In this chapter, lower case letters will be used for local coordinates and upper case letters for global coordinates.

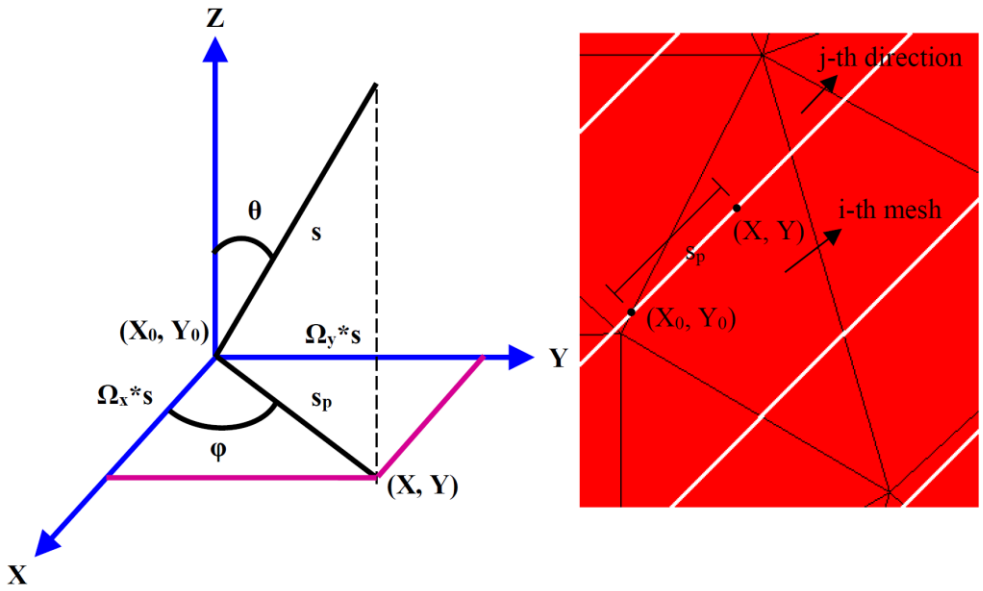

Fig. 4.1: Illustration of the polar angle θ and azimuthal angle φ made by the actual 3D characteristic

As the solution of the transport equation, given in Eq.4.1, is valid along a straight line characteristic in the direction j, we can replace X and Y with ($X_0+\Omega_x s$) and ($Y_0+\Omega_y s$) respectively in this equation. Here (X_0, Y_0) is the point at which the straight line enters the mesh, s is the chord length measured along the line between the points (X_0, Y_0) and (X, Y) and Ω_x, Ω_y are direction cosines mathematically expressed as $\Omega_x = \sin\theta\cos\varphi$ and $\Omega_y = \sin\theta\sin\varphi$ where θ and φ are the spherical polar angles in the X, Y, Z coordinate system (Fig. 4.1). After making these substitutions, the source can be written as follows

$$Q_{i,j,g}(s) = b_0 + b_1 s \tag{4.3}$$

where $b_0 = a_0+a_1(X_0-X_c)+a_2(Y_0-Y_c)$ and $b_1 = a_1\Omega_x+a_2\Omega_y$. b_0 and b_1 are the new expansion coefficients for the above expression. Integrating Eq.3.4, with the source in the form given in Eq.4.3, we obtain the following solution.

$$\Psi_{i,j,g}^{out} = \Psi_{i,j,g}^{in} e^{-\Sigma_{i,g}^{t}\Delta t_{i,j}} + \left[b_0 \frac{1-e^{-\Sigma_{i,g}^{t}\Delta t_{i,j}}}{\Sigma_{i,g}^{t}} + b_1 \left(\frac{\Delta t_{i,j}}{\Sigma_{i,g}^{t}} - \frac{1-e^{-\Sigma_{i,g}^{t}\Delta t_{i,j}}}{\Sigma_{i,g}^{t\ 2}} \right) \right] \tag{4.4}$$

This solution is consistent with the one obtained under the flat source assumption as Eq.4.4 reduces to Eq.3.6 on setting $b_0 = Q_{i,g}$ and $b_1 = 0$. The coefficients b_0 and b_1 have already been expressed in terms of a_0, a_1 and a_2 that will be obtained from the average source and its moments.

Integrating Eq.4.3 over "s" from 0 to $\Delta t_{i,j}$ and then dividing by $\Delta t_{i,j}$, we get the average source over one chord in direction j and passing through mesh i

$$\begin{aligned} \overline{Q}_{i,j,g} &= \frac{1}{\Delta t_{i,j}} \int_0^{\Delta t_{i,j}} Q_{i,j,g}(x,y)\, ds \\ &= a_0 + a_1 \left(X_0 - X_c + \Omega_x \frac{\Delta t_{i,j}}{2} \right) + a_2 \left(Y_0 - Y_c + \Omega_y \frac{\Delta t_{i,j}}{2} \right) \end{aligned} \tag{4.5}$$

There could be many such chords parallel to one another and passing through the mesh in the direction j. Averaging further over all those chords in direction j that pass through the mesh i, gives

$$\overline{\overline{Q}}_{i,j,g} = \frac{\sum_n \overline{Q}_{i,j,n,g} \Delta t_{i,j,n} \Delta w_{j,n}}{\sum_n \Delta t_{i,j,n} \Delta w_{j,n}} \tag{4.6}$$

A new index n has been introduced to indicate the parallel chords that pass through the mesh i. To carry out the averaging of the source over all chords passing through mesh i in direction j [in Eq.4.6], the contribution from each chord [as given in Eq.4.5], is weighted by the product of chord lengths ($\Delta t_{i,j,n}$) and the separation between two consecutive parallel chords ($\Delta w_{j,n}$) (i.e. the volume associated with each chord in mesh i). Substituting the expression given by the RHS of Eq.4.5 in Eq.4.6, we obtain

$$q_{i,g} = \overline{\overline{Q}}_{i,j,g} = a_0 + a_1 q_x + a_2 q_y \tag{4.7}$$

where q_x and q_y are given by

$$q_x = \frac{\sum_n \left(X_0 - X_c + \Omega_x \frac{\Delta t_{i,j,n}}{2} \right) \Delta t_{i,j,n} \Delta w_{j,n}}{\sum_n \Delta t_{i,j,n} \Delta w_{j,n}} \tag{4.8}$$

$$\text{and } q_y = \frac{\sum_n \left(Y_0 - Y_c + \Omega_y \frac{\Delta t_{i,j,n}}{2} \right) \Delta t_{i,j,n} \Delta w_{j,n}}{\sum_n \Delta t_{i,j,n} \Delta w_{j,n}} \tag{4.9}$$

In a similar fashion, the 1st moment of the source (with respect to x) over one chord in the direction j and passing through the mesh i is obtained by multiplying Eq.4.3 with "x", integrating over "s" from 0 to $\Delta t_{i,j}$, and dividing by $\Delta t_{i,j}$.

$$\begin{aligned} \overline{Q}^{x1}_{i,j,g} &= \frac{1}{\Delta t_{i,j}} \int_0^{\Delta t_{i,j}} x Q_{i,j,g}(x,y)\, ds \\ &= a_0 \left(X_0 - X_c + \Omega_x \frac{\Delta t_{i,j}}{2} \right) + a_1 \left[\left(X_0 - X_c \right)^2 + \left(X_0 - X_c \right) \Omega_x \Delta t_{i,j} + {\Omega_x}^2 \frac{{\Delta t_{i,j}}^2}{3} \right] \\ &+ a_2 \left[\left(X_0 - X_c \right)\left(Y_0 - Y_c \right) + \left\{ \left(Y_0 - Y_c \right) \Omega_x + \left(X_0 - X_c \right) \Omega_y \right\} \frac{\Delta t_{i,j}}{2} + \Omega_x \Omega_y \frac{{\Delta t_{i,j}}^2}{3} \right] \end{aligned} \tag{4.10}$$

As in Eq.4.6, we carry out averaging over various chords passing through the mesh i in direction j to obtain the average moment of the source (with respect to x) for the mesh i in direction j.

$$\overline{\overline{Q}}^{x1}_{i,j,g} = \frac{\sum_n \overline{Q}^{x1}_{i,j,n,g} \Delta t_{i,j,n} \Delta w_{j,n}}{\sum_n \Delta t_{i,j,n} \Delta w_{j,n}} \tag{4.11}$$

Substituting the expression given by the RHS of Eq.4.10 in Eq.4.11, we obtain

$$q_{i,g}^{x1} = \overline{\overline{Q}}^{x1}_{i,j,g} = a_0 q_x + a_1 q_{xx} + a_2 q_{xy} \tag{4.12}$$

where q_{xx} and q_{xy} are given by

$$q_{xx} = \frac{\sum_n \left(\left(X_0 - X_c\right)^2 + \left(X_0 - X_c\right)\Omega_x \Delta t_{i,j,n} + {\Omega_x}^2 \frac{{\Delta t_{i,j,n}}^2}{3} \right) \Delta t_{i,j,n} \Delta w_{j,n}}{\sum_n \Delta t_{i,j,n} \Delta w_{j,n}} \tag{4.13}$$

$$q_{xy} = \frac{\sum_n \left(\begin{array}{l} \left(X_0 - X_c\right)\left(Y_0 - Y_c\right) \\ + \left\{ \left(Y_0 - Y_c\right)\Omega_x + \left(X_0 - X_c\right)\Omega_y \right\} \frac{\Delta t_{i,j,n}}{2} + \Omega_x \Omega_y \frac{{\Delta t_{i,j,n}}^2}{3} \end{array} \right) \Delta t_{i,j,n} \Delta w_{j,n}}{\sum_n \Delta t_{i,j,n} \Delta w_{j,n}} \tag{4.14}$$

and q_x is already defined in Eq.4.8.

Another source moment (over one chord in direction j and passing through the mesh i) can be obtained by multiplying Eq.4.3 with "y" followed by integration over "s" from 0 to $\Delta t_{i,j}$ and dividing by $\Delta t_{i,j}$.

$$\begin{aligned} \overline{Q}^{y1}_{i,j,g} &= \frac{1}{\Delta t_{i,j}} \int_0^{\Delta t_{i,j}} y Q_{i,j,g}\left(x,y\right) ds \\ &= a_0 \left(Y_0 - Y_c + \Omega_y \frac{\Delta t_{i,j}}{2} \right) + a_1 \left[\begin{array}{l} \left(X_0 - X_c\right)\left(Y_0 - Y_c\right) \\ + \left\{ \left(Y_0 - Y_c\right)\Omega_x + \left(X_0 - X_c\right)\Omega_y \right\} \frac{\Delta t_{i,j}}{2} \\ + \Omega_x \Omega_y \frac{{\Delta t_{i,j}}^2}{3} \end{array} \right] \\ &+ a_2 \left[\left(Y_0 - Y_c\right)^2 + \left(Y_0 - Y_c\right)\Omega_y \Delta t_{i,j} + {\Omega_y}^2 \frac{{\Delta t_{i,j}}^2}{3} \right] \end{aligned} \tag{4.15}$$

Averaging over all the chords passing through the mesh i in direction j, we obtain

$$\overline{\overline{Q}}^{y1}_{i,j,g} = \frac{\sum_n \overline{Q}^{y1}_{i,j,n,g} \Delta t_{i,j,n} \Delta w_{j,n}}{\sum_n \Delta t_{i,j,n} \Delta w_{j,n}} \tag{4.16}$$

Substituting the expression given by the RHS of Eq.4.15 in Eq.4.16, we obtain

$$q_{i,g}^{y1} = \overline{\overline{Q}}_{i,j,g}^{=y1} = a_0 q_y + a_1 q_{xy} + a_2 q_{yy} \tag{4.17}$$

where q_y, q_{xy} are defined in Eq.4.9 and Eq.4.14 respectively and

$$q_{yy} = \frac{\sum_n \left(\left(Y_0 - Y_c\right)^2 + \left(Y_0 - Y_c\right)\Omega_y \Delta t_{i,j,n} + \Omega_y{}^2 \frac{\Delta t_{i,j,n}{}^2}{3} \right) \Delta t_{i,j,n} \Delta w_{j,n}}{\sum_n \Delta t_{i,j,n} \Delta w_{j,n}} \tag{4.18}$$

If we define (X_c, Y_c), the centroid of the mesh, in the following way

$$X_c = \frac{\sum_n \left(X_0 + \Omega_x \frac{\Delta t_{i,j,n}}{2} \right) \Delta t_{i,j,n} \Delta w_{j,n}}{\sum_n \Delta t_{i,j,n} \Delta w_{j,n}} \tag{4.19}$$

$$Y_c = \frac{\sum_n \left(Y_0 + \Omega_y \frac{\Delta t_{i,j,n}}{2} \right) \Delta t_{i,j,n} \Delta w_{j,n}}{\sum_n \Delta t_{i,j,n} \Delta w_{j,n}} \tag{4.20}$$

then both q_x and q_y will vanish. Solving Eq.4.7, Eq.4.12 and Eq.4.17, we get the following expressions for a_0, a_1 and a_2.

$$\left.\begin{aligned} a_0 &= q_{i,g} \\ a_1 &= \frac{q_{i,g}^{x1} q_{yy} - q_{i,g}^{y1} q_{xy}}{q_{xx} q_{yy} - q_{xy}{}^2} \\ a_2 &= \frac{q_{i,g}^{y1} q_{xx} - q_{i,g}^{x1} q_{xy}}{q_{xx} q_{yy} - q_{xy}{}^2} \end{aligned}\right\} \tag{4.21}$$

In order to calculate the average source $q_{i,g}$ and its moments $q_{i,g}{}^{x1}$ and $q_{i,g}{}^{y1}$ in the above equations, we need the scalar flux and its moments. First, we determine the average flux and its moments along one chord in direction j and passing through mesh i. On integrating Eq.3.4 over "s" from 0 to $\Delta t_{i,j}$ and dividing by $\Delta t_{i,j}$, we obtain the average angular flux over one chord in direction j and passing through the mesh i.

$$\overline{\Psi}_{i,j,g} = \frac{1}{\Delta t_{i,j}} \int_0^{\Delta t_{i,j}} \Psi_{i,j,g}(s)\,ds = \frac{\overline{Q}_{i,j,g}}{\Sigma^t_{i,g}} + \frac{\Psi^{in}_{i,j,g} - \Psi^{out}_{i,j,g}}{\Delta t_{i,j}\Sigma^t_{i,g}} \tag{4.22}$$

Similarly, on multiplying Eq.3.4 by "x" [or "y"], integrating over "s" from 0 to $\Delta t_{i,j}$, and dividing by $\Delta t_{i,j}$, we obtain the first moment of flux (with respect to x [or y]) over one chord in direction j and passing through the mesh i.

$$\overline{\Psi}^{x1}_{i,j,g} = \frac{1}{\Delta t_{i,j}} \int_0^{\Delta t_{i,j}} x\Psi_{i,j,g}(s)\,ds = (X_0 - X_c)\overline{\Psi}_{i,j,g} + \Omega_x \overline{\Psi}^1_{i,j,g} \tag{4.23}$$

$$\overline{\Psi}^{y1}_{i,j,g} = \frac{1}{\Delta t_{i,j}} \int_0^{\Delta t_{i,j}} y\Psi_{i,j,g}(s)\,ds = (Y_0 - Y_c)\overline{\Psi}_{i,j,g} + \Omega_y \overline{\Psi}^1_{i,j,g} \tag{4.24}$$

Eq.4.23 and Eq.4.24 contain the first moment of angular flux with respect to s, $\overline{\Psi}^1_{i,j,g}$, which needs to be calculated. After multiplying Eq.3.4 with "s" and then integrating the equation over "s" from 0 to $\Delta t_{i,j}$ and finally dividing by $\Delta t_{i,j}$, we get

$$\overline{\Psi}^1_{i,j,g} = \frac{1}{\Delta t_{i,j}} \int_0^{\Delta t_{i,j}} s\Psi_{i,j,g}(s)\,ds = \frac{1}{\Sigma^t_{i,g}}\left[\overline{Q}^1_{i,j,g} - \Psi^{out}_{i,j,g} + \overline{\Psi}_{i,j,g}\right] \tag{4.25}$$

Putting Eq.4.25 in Eq.4.23 and Eq.4.24, we get

$$\overline{\Psi}^{x1}_{i,j,g} = (X_0 - X_c)\overline{\Psi}_{i,j,g} + \frac{\Omega_x}{\Sigma^t_{i,g}}\left[\overline{Q}^1_{i,j,g} - \Psi^{out}_{i,j,g} + \overline{\Psi}_{i,j,g}\right] \tag{4.26}$$

$$\overline{\Psi}^{y1}_{i,j,g} = (Y_0 - Y_c)\overline{\Psi}_{i,j,g} + \frac{\Omega_y}{\Sigma^t_{i,g}}\left[\overline{Q}^1_{i,j,g} - \Psi^{out}_{i,j,g} + \overline{\Psi}_{i,j,g}\right] \tag{4.27}$$

Further averaging Eq.4.22, Eq.4.26 and Eq.4.27 over all the chords in direction j and passing through the mesh i yields

$$\overline{\overline{\Psi}}_{i,j,g} = \frac{\sum_n \overline{\Psi}_{i,j,n,g} \Delta t_{i,j,n} \Delta w_{j,n}}{\sum_n \Delta t_{i,j,n} \Delta w_{j,n}} \tag{4.28}$$

$$\overline{\overline{\Psi}}^{x1}_{i,j,g} = \frac{\sum_n \overline{\Psi}^{x1}_{i,j,n,g} \Delta t_{i,j,n} \Delta w_{j,n}}{\sum_n \Delta t_{i,j,n} \Delta w_{j,n}} \tag{4.29}$$

$$\overline{\overline{\Psi}}^{y1}_{i,j,g} = \frac{\sum_n \overline{\Psi}^{y1}_{i,j,n,g} \Delta t_{i,j,n} \Delta w_{j,n}}{\sum_n \Delta t_{i,j,n} \Delta w_{j,n}} \tag{4.30}$$

Finally, integrating Eqs.4.28-4.30 over all the directions, we get the scalar flux and its moments which are used to calculate the average source and its moments in Eq.4.21.

$$\phi_{i,g} = \sum_j w_{\varphi_j} w_{\theta_j} \overline{\overline{\Psi}}_{i,j,g} \tag{4.31}$$

$$\phi^{x1}_{i,g} = \sum_j w_{\varphi_j} w_{\theta_j} \overline{\overline{\Psi}}^{x1}_{i,j,g} \tag{4.32}$$

$$\phi^{y1}_{i,g} = \sum_j w_{\varphi_j} w_{\theta_j} \overline{\overline{\Psi}}^{y1}_{i,j,g} \tag{4.33}$$

where $w_{\theta j}$ and $w_{\varphi j}$ are the weights corresponding to the cosine of the polar angle θ_j (obtained from a Gauss quadrature set) and the azimuthal angle φ_j respectively, associated with the direction j. The weights $w_{\theta j}$ s' and $w_{\varphi j}$ s' are normalized to 2 and 2 π respectively.

The average source and its moments can now be related to the scalar flux and its moments using the definition of the source in the presence of isotropic scattering.

$$q_{i,g} = \frac{1}{4\pi} \left(\frac{\chi_g}{\lambda} \sum_{g'=1}^{G} \nu \Sigma^i_{fg'} + \sum_{g'=1}^{G} \Sigma^i_{s,g' \to g} \right) \phi_{i,g'} \tag{4.34}$$

$$q^{x1}_{i,g} = \frac{1}{4\pi} \left(\frac{\chi_g}{\lambda} \sum_{g'=1}^{G} \nu \Sigma^i_{fg'} + \sum_{g'=1}^{G} \Sigma^i_{s,g' \to g} \right) \phi^{x1}_{i,g'} \tag{4.35}$$

$$q_{i,g}^{y1} = \frac{1}{4\pi}\left(\frac{\chi_g}{\lambda} \sum_{g'=1}^{G} \nu\Sigma^{i}_{fg'} + \sum_{g'=1}^{G} \Sigma^{i}_{s,g'\to g} \right) \phi_{i,g'}^{y1} \tag{4.36}$$

$\overline{Q}_{i,j,g}$ and $\overline{Q}^{1}_{i,j,g}$ in Eq.4.22, Eq.4.26 and Eq.4.27 are determined using Eq.4.3.

$$\overline{Q}_{i,j,g} = \frac{1}{\Delta t_{i,j}} \int_{0}^{\Delta t_{i,j}} Q_{i,j,g}(s)\, ds = b_0 + b_1 \frac{\Delta t_{i,j}}{2} \tag{4.37}$$

$$\overline{Q}^{1}_{i,j,g} = \frac{1}{\Delta t_{i,j}} \int_{0}^{\Delta t_{i,j}} sQ_{i,j,g}(s)\, ds = \Delta t_{i,j}\left(\frac{b_0}{2} + b_1 \frac{\Delta t_{i,j}}{3} \right) \tag{4.38}$$

Thus the set of equations [Eq.4.4, 4.13, 4.14, 4.18, 4.21, 4.22, 4.26, 4.27, 4.28-4.36, 4.37, 4.38] becomes closed.

The solution of the transport equation for obtaining the multiplication factor (k) and the flux distribution for a system (lattice or full core) is obtained using the usual inner-outer iteration method, as described in previous chapter. A flowchart of the solution method is given in Fig. 4.2.

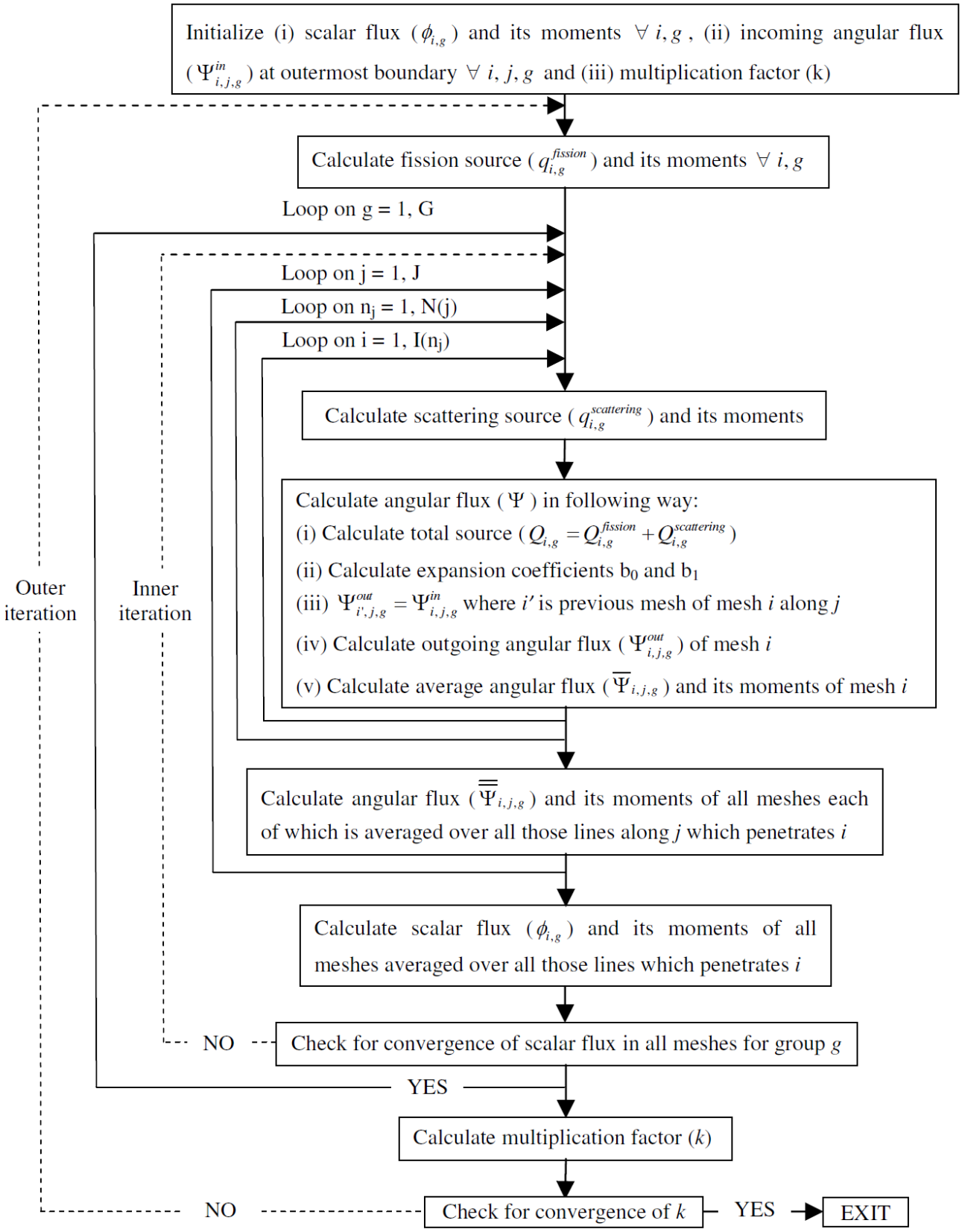

Fig. 4.2: Flowchart of the inner outer iteration technique for solution of the neutron transport equation by the MOC of with a linear representation of the source.

4.3 Correction in the representation of curved surfaces with coarse triangular meshes

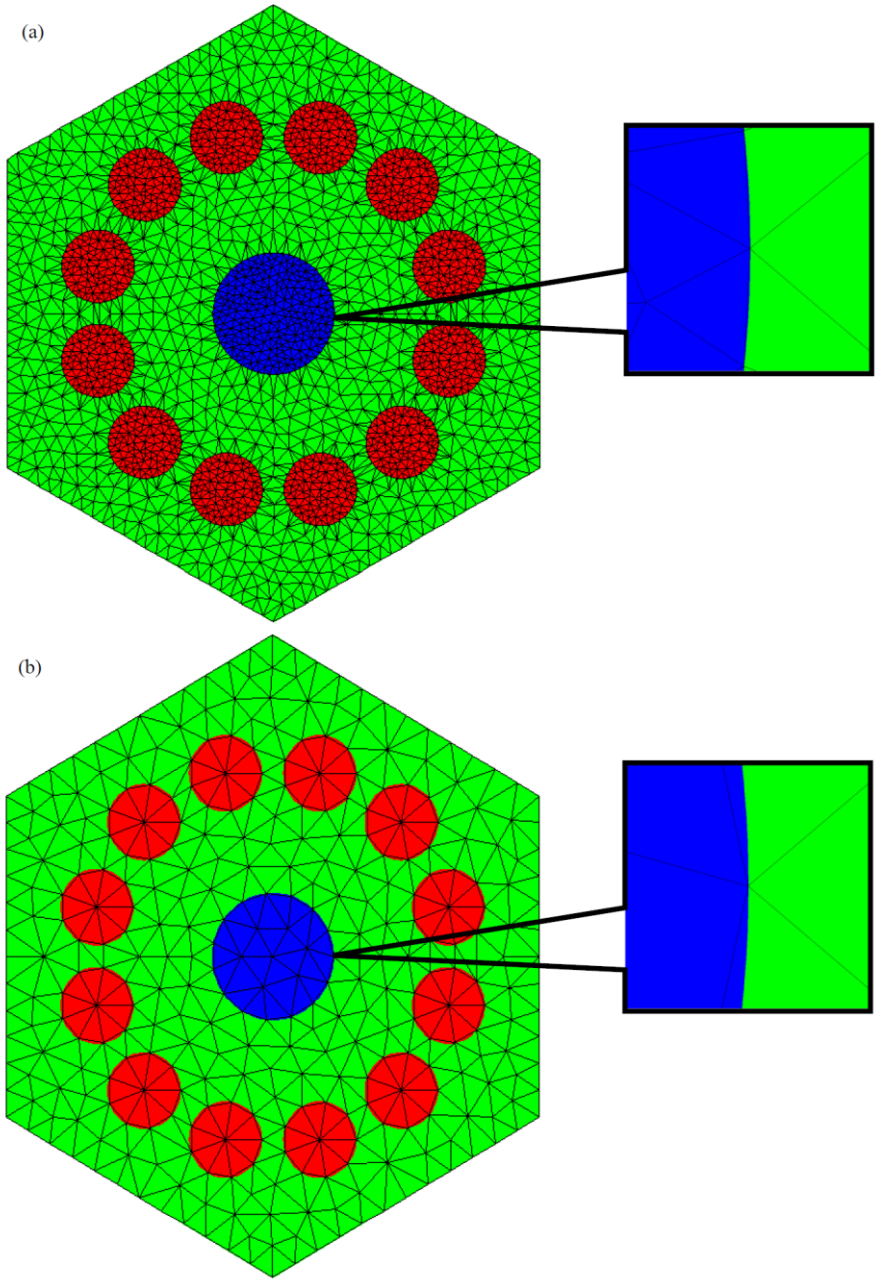

Fig. 4.3: (a) Triangulation of a geometry with fine meshes; In the zoomed view, the circular boundary of the central pin coincides with the straight edges of the fine triangles. (b) Triangulation of a geometry with coarse meshes; In the zoomed view, the circular boundary of the central pin does not coincide with the straight edges of the coarse triangles.

In the earlier version of MOC code, discussed in the previous chapter, the geometry of the problem (lattice or full core) was divided into a number of triangular meshes by Delaunay triangulation technique along with Bowyer-Watson algorithm in such a manner that the material properties are uniform within each mesh. Care was taken in choosing the mesh size small enough so that the assumption of a constant or flat source distribution made for solving the transport equation along a characteristic line over its intercept in the mesh remains valid. In the present chapter, we have already worked out the modification in the solution of transport equation by MOC if we choose to consider coarse mesh in order to reduce the computation time. The main difference between the earlier and the present formulations is the representation of the source within a mesh. In the earlier case, it was a constant while in present case, the source is expanded up to linear terms. In case of problems with planar material boundaries, the triangular mesh boundaries coincide exactly with the material boundaries. In such situations the geometrical representation using triangulations is exact whether the mesh is coarse or fine and results for such a geometry divided into a few coarse meshes are found to be in good agreement with that for the same geometry divided into large number of fine meshes. However, this is not true for problems involving cylindrical bodies (e.g. boundaries of fuel pin, clad, coolant channel etc.) whose circular projections cannot be represented exactly with the straight edges of the triangular meshes. With this approximate representation, shown in Fig. 4.3, it was possible to get accurate results since the error introduced due to the geometrical representation was negligible for a fine mesh. However, with a coarse mesh, the geometrical representation error becomes large and the results deviate appreciably from the fine mesh values as will be clear from the

results shown in the next section. Therefore, use of coarse triangular mesh in such geometries is inadequate even with the polynomial expansion. To get good results we need to change the shape of those meshes that fall on the circular boundary. One of the straight edges of triangular meshes on either side of a circular boundary needs to be corrected to an arc so that the meshes can exactly represent a circle. This is exactly what we have implemented in MOC code (Fig. 4.4). After triangulating a problem domain

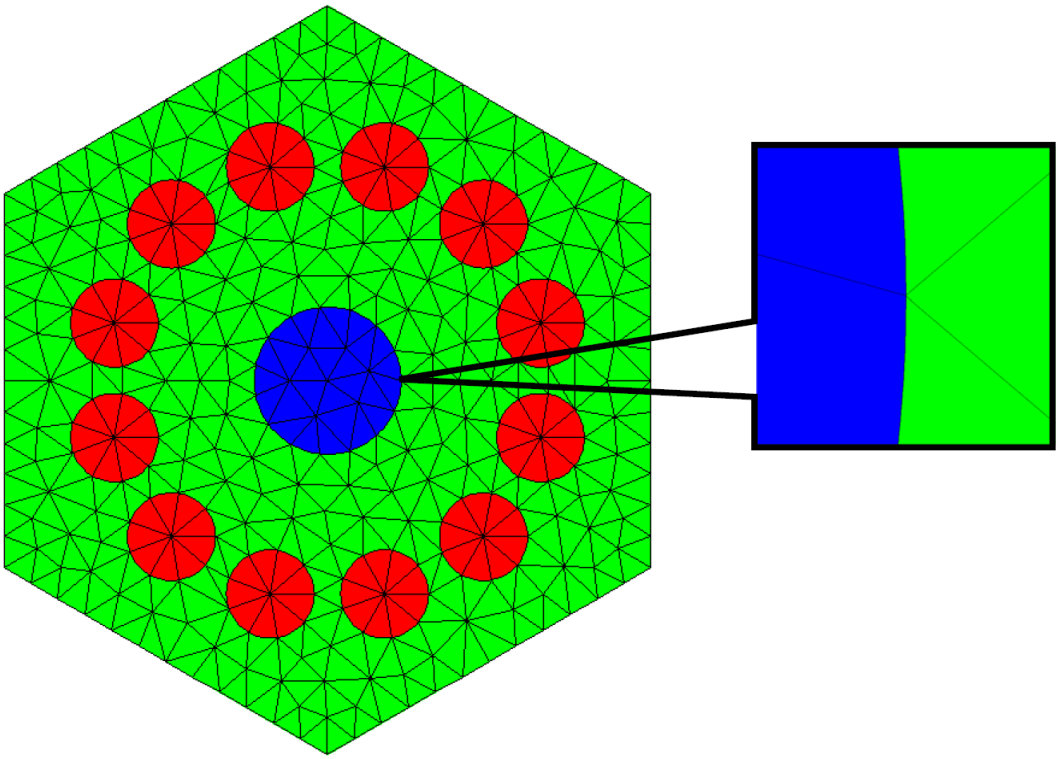

Fig. 4.4: Triangulation of a geometry with coarse meshes; One of the straight edges of a coarse triangle, which falls on the circular boundary, is modified as an arc of the circle. Hence, as seen in the zoomed view, the circular boundary of the central pin coincides with the arcs of the coarse triangle.

by Delaunay triangulation in conjunction with the Bowyer-Watson algorithm, the edges of the triangular meshes, which approximate the circular boundary of the domain, are identified and then replaced with arcs. DISLIN is used in MOC code to plot rectangular, circular and hexagonal geometries with different colours for different materials. For

plotting purposes, a side (say AB) of a triangular mesh (ΔABC) is defined only by the coordinates of two end points of the side [A=(x_1, y_1) and B=(x_2, y_2)]. On the other hand, an arc AB is defined by five parameters, namely centre of curvature (x_0, y_0), radius of curvature (r_0) and angles θ_i and θ_f made by the radius vector r_0 [connecting (x_0, y_0) and (x_1, y_1) for θ_i and connecting (x_0, y_0) and (x_2, y_2) for θ_f] with a reference line as shown in Fig. 4.5. This is followed by ray tracing in which intersection between triangular

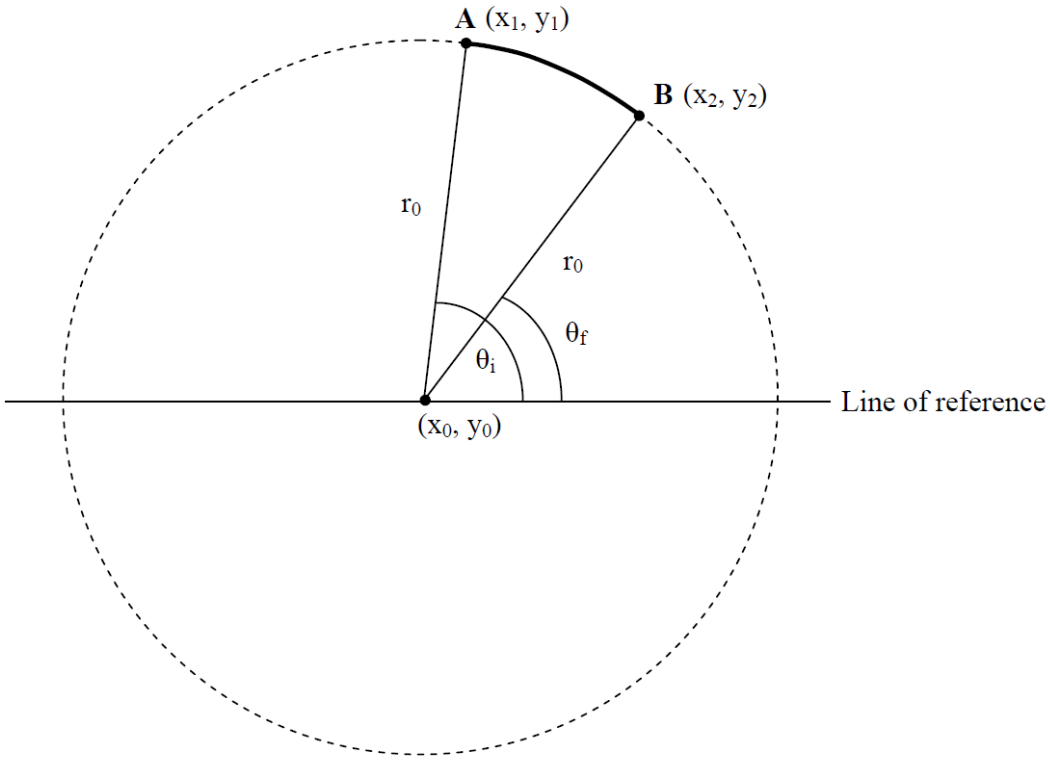

Fig. 4.5: Illustration of the parameters required to draw an arc AB.

meshes and characteristic lines (called "rays") drawn in different directions (as a set of parallel lines in a particular direction) are determined and the information about the tracks of the rays passing through the problem domain are stored for the next level of the calculation. The equation of each of these rays can be written as y = mx + c. For a particular direction, there are a number of parallel lines each having identical slope "m", but variable intercepts "c". "m" is different for different directions. Each triangular

mesh in the geometry is characterized by coordinates of its three vertices, namely A (x_1, y_1), B (x_2, y_2) and C (x_3, y_3). In our previous work, it has been shown how to determine the point of intersection between side AB and the characteristic line. In circular geometry, if vertices A and B fall on the circumference of a circle having its centre at (x_0, y_0) and radius r_0, then the equation of the arc AB of the triangle can be constructed using (x_0, y_0) as the centre of curvature and r_0 as the radius of curvature of the arc. The equation of the arc can be written in the form $y = y_0 \pm \sqrt{r_0^2 - (x - x_0)^2}$. It may be noted that, at the most, one side of a triangle can be replaced by an arc. Hence, the mesh division is carried out in such a manner that only one of the sides of the triangle needs to be replaced by an arc. The characteristic line, while passing through the geometry, may or may not intersect the arc. If the perpendicular distance d to the characteristic line from (x_0, y_0) is greater than r_0, then they will not intersect each other. But, if the distance is less than r_0, then, generally speaking, there will be two points of intersection D (x*, y*) which can be found by solving the equations of the arc and the characteristic line. A check is always made whether D lies within the arc or not using the following prescription. If d_1, d_2 and d_3 are distances between D and A, D and B, A and B respectively, then $\max(d_1, d_2) < d_3$ guarantees that D lies on the arc AB (Fig. 4.6). Coordinates of all the points of intersection are arranged in increasing or decreasing order of their abscissa values according to the direction of characteristics. In addition to coordinates of points, mesh identification number (i.e. some sort of index number which is given to each triangle during mesh generation so that the triangles can be identified later) of each triangle intercepted by the ray, length of the segment of the projection of

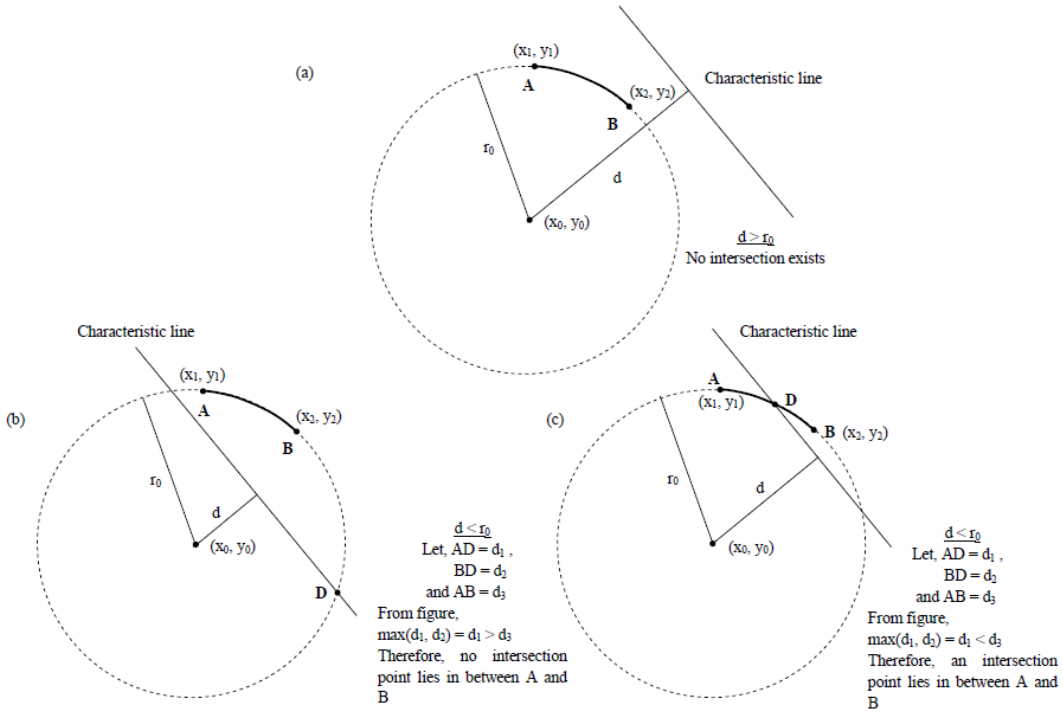

Fig. 4.6: (a) A characteristic line will not intersect the arc AB if $d > r_0$; (b) Even if $d<r_0$, the characteristic line may not intersect arc AB if the point of intersection D is lying outside of AB. (c) $d<r_0$ and the characteristic line intersects arc AB as the point of intersection D is lying within AB.

the characteristic line in direction j and intercepted by the triangle i ($s_{i,j}$) and the material filled in the triangle are stored for subsequent flux calculations by the inner-outer iteration technique. Volumes of all the triangular meshes are required for multiplication factor calculation (Eq.3.12). If the mesh is a perfect triangle with its three vertices at (x_1, y_1), (x_2, y_2) and (x_3, y_3), then its volume per unit length along z axis will be

$$V = \frac{1}{2}\left|x_1\left(y_2 - y_3\right) + x_2\left(y_3 - y_1\right) + x_3\left(y_1 - y_2\right)\right| \tag{4.39}$$

If one of the three edges of a triangular mesh is an arc, as shown by AB in Fig. 4.5, then its volume per unit length along z axis will be

$$V = V_{\Delta} \pm V_{cs} = V_{\Delta} \pm \frac{R^2}{2}\left(\frac{\pi\theta}{180} - \sin\theta\right) \tag{4.40}$$

Here the triangular mesh is divided into two segments – one which forms a perfect triangle (joining points A and B, shown in Fig. 4.5, by a straight line) and another is a circular segment formed by enclosing the region between the straight line AB and the arc. V_{Δ} is the volume of the perfect triangle, as defined in Eq.4.39, and V_{cs} is the volume of the circular segment. In order to obtain the exact volume of the whole mesh, V_{cs} is either added with or subtracted from V_{Δ} depending on the curvature of arc (convex or concave). R is the radius of curvature of the arc and θ is the central angle defined as (θ_i – θ_f) [θ_i and θ_f are shown in Fig. 4.5] and measured in degree. The length of the intercept along the characteristic, used in the calculation, is calculated by dividing the length of the projected segment $s_{i,j}$ with $\sin\theta_j$ where θ_j is the polar angle made by the characteristic line and the z axis (Fig. 4.1).

4.4 Verification and benchmarking of the Code

In order to validate the calculation of multiplication factor (k) with the improved source expansion scheme within a mesh, a number of benchmark problems in square as well as hexagonal geometries have been analyzed. Description and the input parameter required to analyze the benchmark problems are available in Chapter 2. Results obtained using the MOC code are given below. Apart from verification of the code, an attempt has been made to bring out the advantage of the source expansion within a coarse mesh and the necessity of introducing the "geometry correction" whenever the problem includes cylindrical bodies (circular regions in the projection) and the domain is divided into coarse triangular meshes.

4.4.1 IAEA benchmark problem

Integration over the angular variable θ is carried out using the 6 point Gauss-Legendre quadrature set whereas 12 uniformly spaced angles with equal weights are used for integration over φ. 500 parallel lines are considered per direction. Convergence criteria for the neutron flux and the k_{eff} are chosen to be 10^{-5} and 10^{-7} respectively. With the flat and linear source assumption, k_{eff} is calculated using 1,850, 11,238 and 23,478 triangular meshes. The results are displayed in Table- 4.1. The mesh size decreases with increasing number of meshes and the flat source assumption becomes progressively better. Therefore, the k_{eff} calculated by MOC code for 23,478 meshes, is quite close (relative percentage error ~ 0.2% - 0.3%) to the values calculated by SURCU [25], TEPFEM [74] and TPTRI [30]. With a reduction in number of meshes, the results deteriorate (relative percentage error increases from 0.3% to 3.7%). However, such a discrepancy does not appear when a linear source representation is used. The result does not vary much with the number of meshes. It is clear from the table that the k_{eff}, in the linear source case, changes only at the fourth decimal places (relative percentage error ~ 0.1 - 0.2%) if we change the number of meshes and results of all three mesh sizes agree well with the reference values obtained by the codes mentioned above. It is interesting to see the effect of mesh size variation on the neutron scalar flux in both flat and linear source assumption. Accordingly, region averaged scalar flux is calculated for all five regions of the problem. The one group fluxes are scaled to satisfy

$$\sum_{i=1}^{N} \nu\Sigma_f^i \phi^i = 1.0 \tag{4.41}$$

where N is total number of meshes in fuel region.

Table- 4.1: k_{eff} values obtained by MOC code using various number of meshes with flat and linear source assumption for the IAEA benchmark problem and their comparison with reference results

Type of assumption	No. of meshes	k_{eff} calculated by MOC code	Relative % error with respect to reference codes		
			SURCU (k_{eff} =1.0069)	TEPFEM (k_{eff} =1.0079)	TPTRI (k_{eff} =1.0070)
Flat	23,478	1.0044123	0.247	0.346	0.257
	11,238	1.0005144	0.634	0.733	0.644
	1,850	0.9701908	3.646	3.741	3.655
Linear	23,478	1.0087264	0.181	0.082	0.171
	11,238	1.0087122	0.180	0.081	0.170
	1,850	1.0084268	0.152	0.052	0.142

Table- 4.2: Region averaged neutron flux values obtained by MOC code using various number of meshes with flat source assumption for the IAEA benchmark problem and its comparison with reference results

Region	Avg. flux calculated by		No of meshes	Avg. flux	Relative % error with respect to reference codes	
	TEPFEM	TPTRI			TEPFEM	TPTRI
Zone 1	0.016860	0.016850	1,850	0.016847	0.08	0.02
			11,238	0.016855	0.03	0.03
			23,478	0.016856	0.02	0.04
Zone 2	0.000125	0.000130	1,850	0.000170	36.00	30.77
			11,238	0.000134	7.20	3.08
			23,478	0.000129	3.20	0.77
Zone 3	0.000035	0.000026	1,850	0.000058	65.71	123.08
			11,238	0.000042	20.00	61.54
			23,478	0.000040	14.29	53.85
Zone 4	0.000297	0.000282	1,850	0.000406	36.57	43.83
			11,238	0.000317	6.73	12.41
			23,478	0.000304	2.36	7.80
Zone 5	0.000786	0.000766	1,850	0.000963	22.52	25.72
			11,238	0.000831	5.73	8.49
			23,478	0.000810	3.05	5.74

Table- 4.3: Region averaged neutron flux values obtained by MOC code using various number of meshes with linear source assumption for the IAEA benchmark problem and its comparison with reference results

Region	Avg. flux calculated by		No of meshes	Avg. flux	Relative % error with respect to reference codes	
	TEPFEM	TPTRI			TEPFEM	TPTRI
Zone 1	0.016860	0.016850	1,850	0.016857	0.02	0.04
			11,238	0.016857	0.02	0.04
			23,478	0.016857	0.02	0.04
Zone 2	0.000125	0.000130	1,850	0.000125	0.00	3.85
			11,238	0.000124	0.80	4.62
			23,478	0.000124	0.80	4.62
Zone 3	0.000035	0.000026	1,850	0.000037	5.71	42.31
			11,238	0.000038	8.57	46.15
			23,478	0.000038	8.57	46.15
Zone 4	0.000297	0.000282	1,850	0.000295	0.67	4.61
			11,238	0.000294	1.01	4.26
			23,478	0.000294	1.01	4.26
Zone 5	0.000786	0.000766	1,850	0.000791	0.64	3.26
			11,238	0.000789	0.38	3.00
			23,478	0.000789	0.38	3.00

In Table- 4.2 and Table- 4.3, the scaled flux values are compared with the values calculated by TEPFEM and TPTRI. It is clear from the table that the error in scalar flux for flat source assumption is, in general, higher than the error for linear source assumption. Reduction in mesh size improves the accuracy of flat source assumption. But, even then, the accuracy of the flat source assumption is found to be lesser than the accuracy of the linear source assumption.

4.4.2 MZA fast reactor benchmark problem

Integration over the angular variable θ is carried out using the 6 point Gauss-Legendre quadrature set whereas 12 uniformly spaced angles with equal weights are used for integration over φ. 600 parallel lines are considered per direction. Convergence

criteria for the neutron flux and the k_{eff} are chosen to be 10^{-5} and 10^{-7} respectively. Assuming flat as well as linear source, k_{eff} of the reference core is calculated for 224, 754 and 3,952 triangular meshes. The results are displayed in Table- 4.4. The linear source assumption predicts k_{eff} very close (relative percentage error < 0.1%) to the values calculated by GMVP [76] and MOCUM [22] and is seen to be insensitive to number of meshes (the k_{eff} changes only by 20-30 pcm and relative percentage error remains below 0.1% when mesh number is reduced by a factor of 18). In case of the flat source representation, the k_{eff} is 800 pcm (relative percentage error ~ 0.9%) less than the reference values calculated by the codes mentioned above, even with 3,952 meshes. The same trend is seen in the other three cases as is clear from Table- 4.4. Instead of full core, simulation of 1/4th core with reflective boundary condition at the inner surfaces and vacuum boundary condition at the outer surfaces (Fig. 4.7) helps in reducing the computation time by a factor of 2. In order to make the reflective boundary condition compatible with the linear source assumption, a linear interpolation between the outgoing angular flux values at the end points of the adjacent parallel characteristics in the outgoing direction is used to calculate the incoming angular flux for the nearby characteristic in the incoming direction unlike the case of flat source assumption, as discussed in section 3.2.

4.4.3 KNK-II benchmark problem with non-reentrant boundary

Integration over the angular variable θ is carried out using the 6 point Gauss-Legendre quadrature set whereas 12 uniformly spaced angles with equal weights are used for integration over φ. 1000 parallel lines are considered for each direction. The convergence criteria for the neutron flux and the k_{eff} are chosen to be 10^{-5} and 10^{-7} respectively. With flat and linear source representations, the k_{eff} is calculated using 468,

1,882 and 11,628 triangular meshes (Fig. 4.8). The results are displayed in Table- 4.5. As the number of meshes increases, the k_{eff} calculated by MOC code with flat source assumption improves and comes closer to the values calculated by TWOHEX [68], SPANDOM-TA [67] and DIAMANT2 [69].

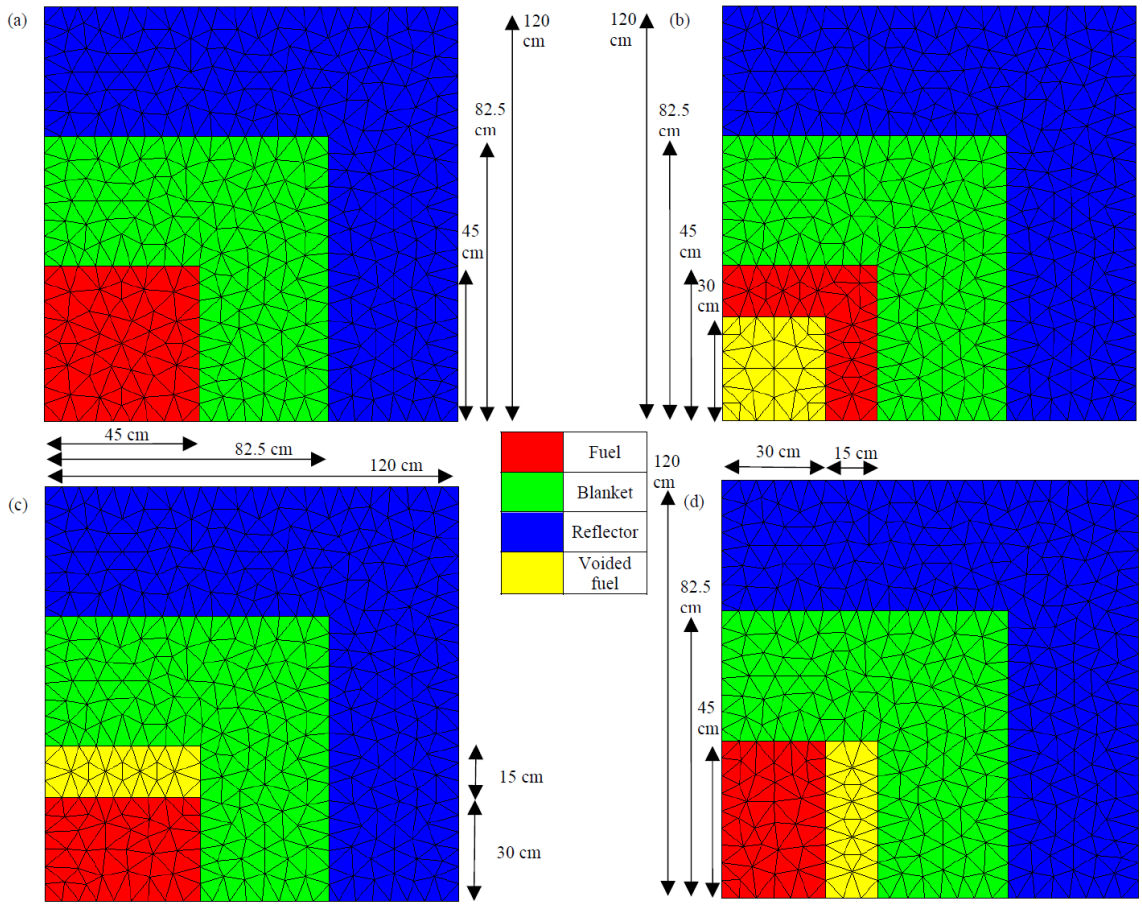

Fig. 4.7: Triangulated geometry of the MZA fast reactor benchmark problem, as generated by MOC code, in four cases: (a) Reference case; (b) void is formed at the centre of fuel region; (c) void is formed at the horizontal boundaries between the fuel and blanket regions, (d) void is formed at the vertical boundaries between the fuel and blanket regions. The dimensions are as shown in the figure. Representation of various materials with different colors is as indicated in the legend table.

For 468 meshes, there is a large difference of about 6800 pcm (relative percentage error ~ 9.2%) between results of MOC code and the codes mentioned above whereas for 11,628 meshes, this difference comes down to 400 pcm (relative percentage error ~ 0.5%). However, with a linear representation of the source, the results are not at

all sensitive to the number of meshes. It is seen from the table, that the k_{eff}, in this case, changes only at fourth or fifth decimal place if we change the number of meshes and the result also agrees well (~ 20-40 pcm difference in reactivity and relative percentage error < 0.1%) with the reference values obtained by the codes mentioned above.

Table- 4.4: k_{eff} values obtained by MOC code using various number of meshes with flat and linear source assumption for the MZA fast reactor benchmark problem and their comparison with reference results

Core configuration	Type of assumption	No. of meshes	k_{eff} calculated by MOC code	Relative % error with respect to reference codes	
				GMVP (k_{eff} =1.18649)	MOCUM (k_{eff} =1.18654)
Case-I (Reference core)	Flat	3,952	1.1755393	0.923	0.927
		754	1.1404651	3.879	3.883
		224	1.0782063	9.126	9.130
	Linear	3,952	1.1860379	0.038	0.042
		754	1.1861337	0.030	0.034
		224	1.1857052	0.066	0.070
				GMVP (k_{eff} =1.19489)	MOCUM (k_{eff} =1.19502)
Case-II (Void at centre)	Flat	3,986	1.1844367	0.875	0.886
		772	1.1521010	3.581	3.591
		218	1.0678615	10.631	10.641
	Linear	3,986	1.1943943	0.041	0.052
		772	1.1944741	0.035	0.046
		218	1.1938117	0.090	0.101
				GMVP (k_{eff} =1.17990)	MOCUM (k_{eff} =1.17995)
Case-III (Void at horizontal boundaries)	Flat	3,972	1.1697185	0.863	0.867
		770	1.1380998	3.543	3.547
		220	1.0662592	9.631	9.635
	Linear	3,972	1.1794480	0.038	0.043
		770	1.1795412	0.030	0.035
		220	1.1788164	0.092	0.096
				GMVP (k_{eff} =1.17988)	MOCUM (k_{eff} =1.17995)
Case-IV (Void at vertical boundaries)	Flat	3,960	1.1696637	0.866	0.872
		770	1.1375036	3.592	3.597
		222	1.0741929	8.957	8.963
	Linear	3,960	1.1794401	0.037	0.043
		770	1.1795299	0.030	0.036
		222	1.1787885	0.093	0.098

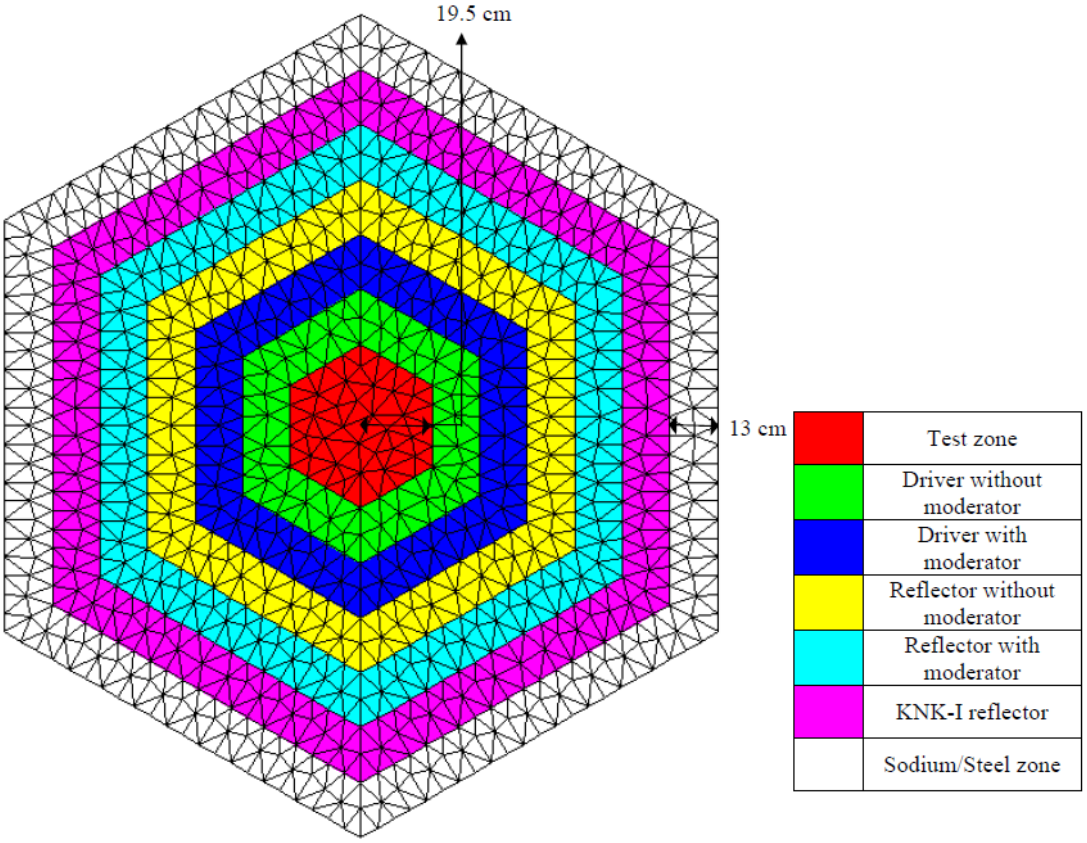

Fig. 4.8: Triangulated geometry of the KNK-II benchmark problem with a non-reentrant boundary, as generated by MOC code. The dimensions are as shown in the figure. Representation of various materials with different colors is as indicated in the legend table.

Table- 4.5: k_{eff} values obtained by MOC code using various number of meshes with flat and linear source assumption for the KNK-II benchmark problem (with non-reentrant boundary) and their comparison with reference results

Type of assumption	No. of meshes	k_{eff} calculated by MOC code	Relative % error with respect to reference codes		
			TWOHEX (k_{eff} =1.49208)	SPANDOM-TA (k_{eff} =1.49132)	DIAMANT2 (k_{eff} =1.49205)
Flat	11628	1.4836363	0.566	0.515	0.564
	1882	1.4461575	3.078	3.028	3.076
	468	1.3539351	9.259	9.212	9.257
Linear	11628	1.4916295	0.030	0.021	0.028
	1882	1.4916604	0.028	0.023	0.026
	468	1.4911083	0.065	0.014	0.063

4.4.4 Hexagonal cell problem with central breeding pin

Integration over the angular variable θ is carried out using the 6 point Gauss-Legendre quadrature set whereas 12 uniformly spaced angles with equal weights are used for integration over φ. 200 parallel lines are considered for each direction. Convergence criteria for the neutron flux and the k_∞ are chosen to be 10^{-5} and 10^{-7} respectively. With flat as well as linear source assumptions, the k_∞ is calculated for 606, 2,298 and 4,266 triangular meshes and the results are displayed in Table- 4.6. Unlike previous benchmark problems, the linear source expansion method does not provide any improvement over the flat source method. As the number of meshes increases, the k_∞ calculated by MOC code using both methods improves and comes closer to the values calculated by MG-MCNP3B [75], TEPFEM and TPTRI. The necessity of circular geometry correction, especially when coarse meshes are considered, was explained in section 4.3. The effect of the correction on the k_∞ calculation can be gauged from Table- 4.6. Before correcting the geometry, the difference between the results of MOC code [using linear source representation and a small number (606) of meshes] and the results of the reference codes mentioned above was about 1500-2000 pcm (relative percentage error ~ 2.0%). After introducing the correction, the k_∞ calculated by MOC code agrees well with the values obtained using the reference codes. It may be noted that our results are closer to the ones by the Monte Carlo method based code MG-MCNP3B.

Table- 4.6: k_∞ values obtained by MOC code using various number of meshes with flat and linear source assumption for the Hexagonal cell problem with central breeding pin and their comparison with reference results

Case	Type of assump -tion	No. of meshes	k_∞ calculated by MOC code	Relative % error with respect to reference codes		
				MCNP3B (k_∞ =1.090803)	TEPFEM (k_∞ =1.086598)	TRTRI (k_∞ =1.085775)
Before geometry correction	Flat	4266	1.0887730	0.186	0.200	0.276
		2298	1.0846146	0.567	0.183	0.107
		606	1.0669462	2.187	1.809	1.734
	Linear	4266	1.0887318	0.190	0.196	0.272
		2298	1.0845718	0.571	0.186	0.111
		606	1.0668671	2.194	1.816	1.741
After geometry correction	Flat	4266	1.0901448	0.060	0.326	0.402
		2298	1.0901898	0.056	0.331	0.407
		606	1.0902084	0.055	0.332	0.408
	Linear	4266	1.0901043	0.064	0.323	0.399
		2298	1.0901443	0.060	0.326	0.402
		606	1.0901212	0.063	0.324	0.400

4.4.5 CANDU-6 annular cell benchmark problem

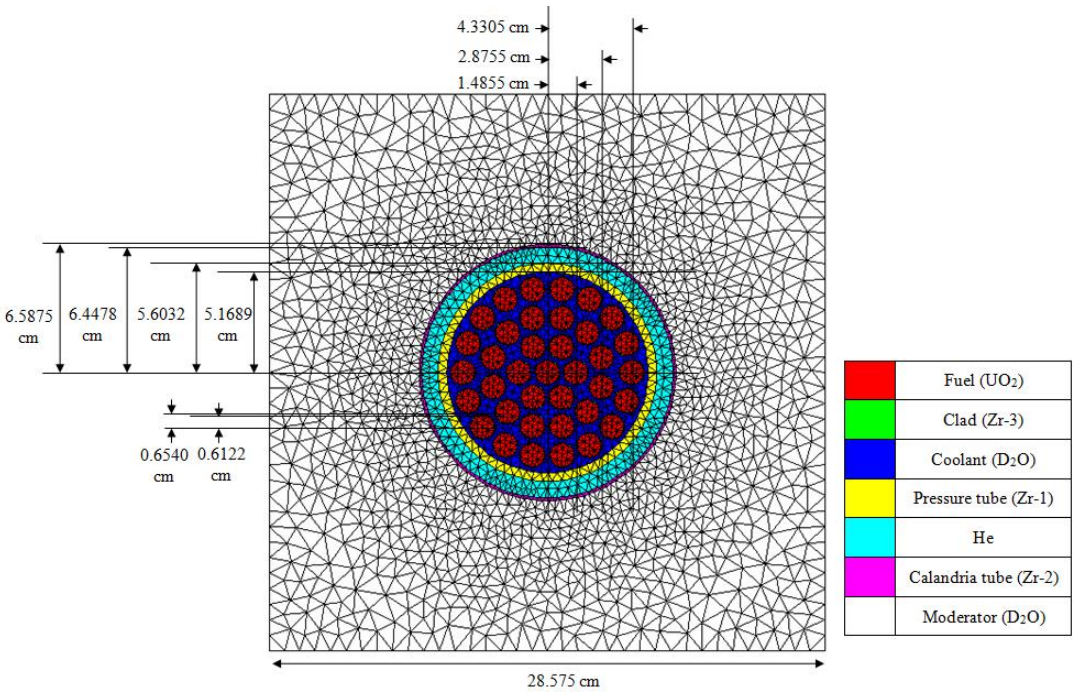

Fig. 4.9: Triangulated geometry of CANDU-6 annular cell benchmark problem, as generated by MOC code. The dimensions are as shown in the figure. Representation of various materials with different colors is as indicated in the legend table.

Table- 4.7: k_∞ values obtained by MOC code using various number of meshes with flat and linear source assumption for the CANDU-6 annular cell benchmark problem and their comparison with reference results

Case	Type of assumption	No. of meshes	k_∞ calculated by MOC code	Relative % error with respect to reference codes	
				DRAGON (k_∞ =1.112837)	MOCUM (k_∞ =1.111358)
Before geometry correction	Flat	7,270	1.1124649	0.033	0.100
		3,004	1.1213775	0.767	0.902
		2,144	1.1227134	0.887	1.022
	Linear	7,270	1.1122050	0.057	0.076
		3,004	1.1206843	0.705	0.839
		2,144	1.1213791	0.768	0.902
After geometry correction	Flat	7,270	1.1113616	0.133	3.2×10^{-4}
		3,004	1.1153494	0.226	0.359
		2,144	1.1159864	0.283	0.416
	Linear	7,270	1.1110956	0.156	0.024
		3,004	1.1146314	0.161	0.295
		2,144	1.1146761	0.165	0.299

Integration over the angular variable θ is carried out using the 6 point Gauss-Legendre quadrature set whereas 12 uniformly spaced angles with equal weights are used for integration over φ. 1500 parallel lines are considered for each direction. The convergence criteria for the neutron flux and the k_∞ are chosen as 10^{-5} and 10^{-7} respectively. With flat as well as linear source assumptions, the k_∞ is calculated using 2,144, 3,004 and 7,270 triangular meshes (Fig. 4.9). Within the same square cell, the rather thin clad (~0.4 mm) region is present along with a large D_2O moderator region which is 7.7 cm thick. Hence, it is very difficult to triangulate the domain with very few meshes unless we choose to permit some ill-shaped triangles which may introduce error in the calculation. The results are displayed in Table- 4.7. It appears that the linear source expansion method does not provide any improvement in the k_∞ over the flat source method. As the number of meshes increases, the k_∞, calculated by MOC code using both methods, improves and tends to the values calculated by DRAGON and MOCUM. However, applying the correction described in section 4.3 on account of the

circular geometry has a significant impact on the results. Before correcting for the geometry, there was a difference of about 600-900 pcm between the results of MOC code for 2,144 meshes and the codes mentioned above whereas after introducing the correction, the k_∞ value agrees pretty well with the results of the reference codes, mentioned above, irrespective of the number of meshes.

Table- 4.8: k_∞ values obtained by MOC code using various number of meshes with flat and linear source assumption for the LWR benchmark problem with burnable poison and their comparison with reference results

Type of assumption	No. of meshes	k_∞ calculated by MOC code	Relative % error with respect to reference codes	
			SURCU (k_∞ = 0.8805)	TPTRI (k_∞ = 0.8828)
Flat	8,160	0.8789742	0.173	0.433
	2,360	0.8786269	0.213	0.473
	978	0.8781078	0.272	0.532
	304	0.8751662	0.606	0.865
Linear	8,160	0.8790279	0.167	0.427
	2,360	0.8789652	0.174	0.434
	978	0.8789674	0.174	0.434
	304	0.8786380	0.211	0.471

4.4.6 LWR benchmark problem with burnable poison

Integration over the angular variable θ is carried out using the 6 point Gauss-Legendre quadrature set whereas 12 uniformly spaced angles with equal weights are used for integration over φ. 800 parallel lines are considered per direction. Convergence criteria for the neutron flux and the k_∞ are chosen to be 10^{-5} and 10^{-7} respectively. For both flat and linear source representations, the k_∞ is calculated for 8,160, 2,360, 978 and 304 meshes. The results are displayed in Table- 4.8. With a large number of meshes, the k_∞ values for both assumptions are close to the values calculated by SURCU and TPTRI. However, the results for the flat source assumption deviate significantly with a reduction in the number of meshes whereas for the linear source, the deviation is

minimal. It is encouraging to observe that the k_∞ obtained using 2,360 meshes with the flat source assumption is equal to the k_∞ value obtained with only 304 meshes with the linear source assumption. This proves the fact that for coarse meshes the linear source assumption is much better than the flat source assumption.

4.4.7 HTTR benchmark problem

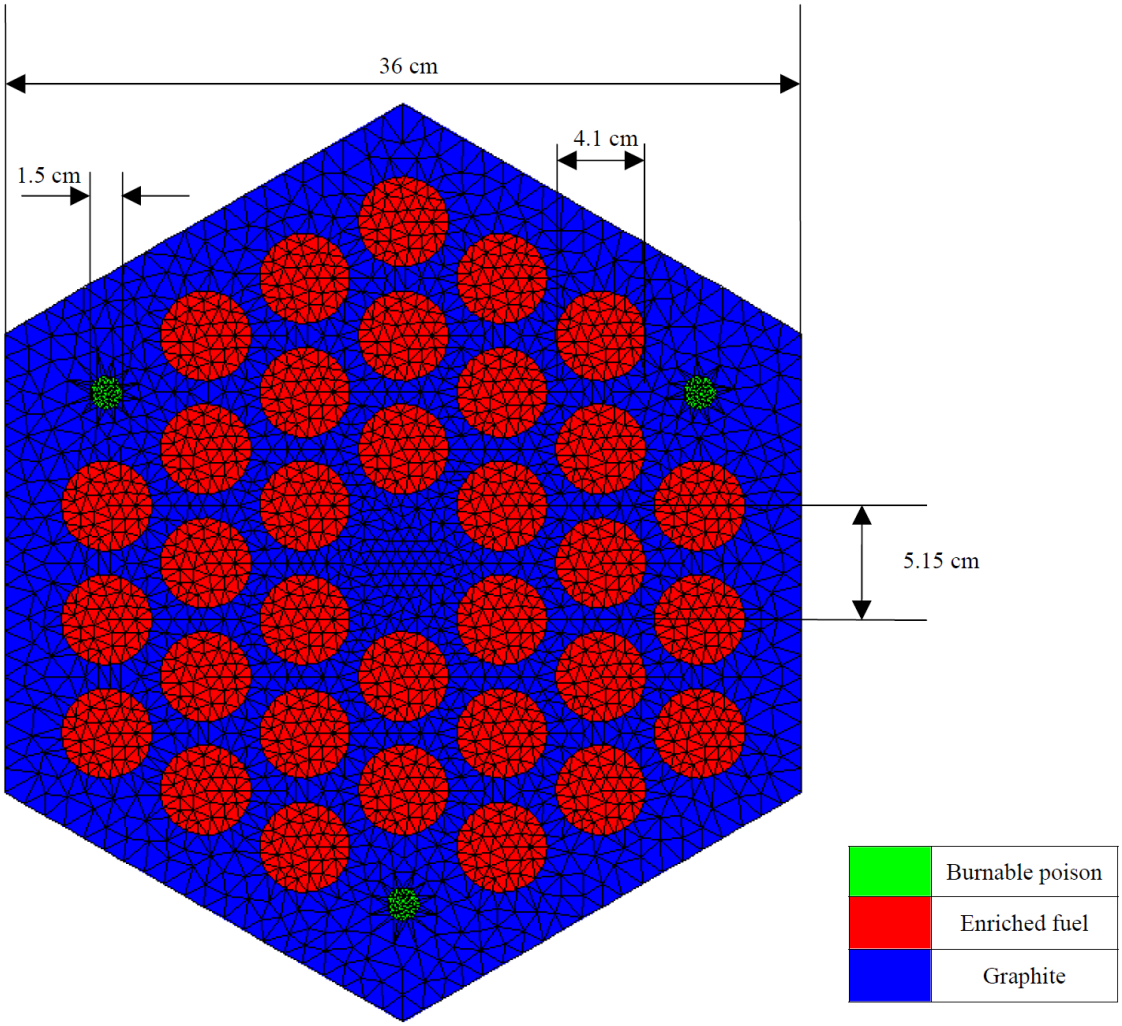

Fig. 4.10: Triangulated geometry of HTTR benchmark problem, as generated by MOC code. The dimensions are as shown in the figure. Representation of various materials with different colors is as indicated in the legend table.

Table- 4.9: k_∞ values obtained by MOC code using various number of meshes with flat and linear source assumption for the HTTR benchmark problem and their comparison with reference results

Case	Type of assumption	No. of meshes	k_∞ calculated by our code	Relative percentage error with respect to reference codes	
				MCNP5 (k_∞ = 1.04119)	MOCUM (k_∞ = 1.04084)
Before geometry correction	Flat	5027	1.0435405	0.226	0.259
		4010	1.0532733	1.161	1.195
		1764	1.0535779	1.190	1.224
	Linear	5027	1.0443165	0.300	0.334
		4010	1.0548177	1.309	1.343
		1764	1.0556274	1.387	1.421
After geometry correction	Flat	5027	1.0402014	0.095	0.061
		4010	1.0389201	0.218	0.184
		1764	1.0381983	0.287	0.254
	Linear	5027	1.0409793	0.020	0.013
		4010	1.0405503	0.061	0.028
		1764	1.0403652	0.079	0.046

Integration over the angular variable θ is carried out using the 6 point Gauss-Legendre quadrature set whereas 12 uniformly spaced angles with equal weights are used for integration over φ. 500 parallel lines are considered per direction. Convergence criteria for the neutron flux and the k_∞ are chosen to be 10^{-5} and 10^{-7} respectively. With the flat and linear source assumption, k_∞ is calculated using 1,764, 4,010 and 5,027 triangular meshes (Fig. 4.10). The results are shown in Table- 4.9. From the first part of the table (Case under the heading 'Before geometry correction'), it appears that the linear source assumption does not provide any improvement in the k_∞ over the flat source assumption. As the number of meshes increases, the k_∞, calculated by MOC code with flat as well as linear source assumptions, improves and tends to the values calculated by MCNP5 [62] and MOCUM. However, applying the correction described in section 4.3 on account of the circular geometry has a significant impact on the results. Before correcting for the geometry, there was a relative percentage error of about 1.2-

1.4% between the results of MOC code for 1,764 meshes and the codes mentioned above whereas after introducing the correction, the k_∞ value agrees pretty well (relative percentage error < 0.3%) with the results of the reference codes, mentioned above, irrespective of the number of meshes. It is to be noted that the error in k_∞ for flat source assumption, even after the geometry correction, increases from 0.1% to 0.3% as the meshes become coarser (i.e. number of meshes decreases) while the error, though increases slightly with the reduction in number of meshes, remains well below 0.1% for linear source assumption. Another quantity, called fission density, is calculated for all the fuel pins in the fuel block. Fission density f_k of k-th fuel pin is defined as

$$f_k = 33\frac{\sum_{g=1}^{G}\sum_{i=1}^{I_k}\Sigma_{fg}\phi_g^i V_i}{\sum_{g=1}^{G}\sum_{i=1}^{I}\Sigma_{fg}\phi_g^i V_i} \qquad (k = 1, 2, ..., 33) \qquad (4.42)$$

where I_k and I are the total number of meshes in k-th fuel pin all 33 fuel pins, respectively. Numerator, in the above definition, represents total fission reaction rate in a single fuel pin and denominator represents total fission reaction rate in all 33 fuel pins. This fractional quantity is then normalized to 33 i.e. the total number of fuel pins. Trend of the improvement of fission density distribution, similar to k_∞, can be found from the values shown in Table- 4.9, Table- 4.10, Table- 4.11 and Table- 4.12. Fission density can be calculated accurately if we incorporate the correction in the simulation of circular geometry. After making the geometry correction, linear source assumption becomes more promising as compared to flat source assumption since error in the fission density, calculated by linear assumption, is relatively less sensitive to the size (or number) of the meshes as compared to flat assumption.

Table- 4.10: Pin wise fission density distribution obtained by MOC code using various number of meshes with flat source assumption for the HTTR benchmark problem and its comparison with reference results (Without geometry correction)

		Flat source assumption			
		1,764 mesh		5,027 mesh	
Pin	MCNP5	Fission density	Relative % error	Fission density	Relative % error
1	0.99862	0.99938	0.08	0.99865	0.00
2	1.01562	1.01451	0.11	1.01511	0.05
3	1.03231	1.02977	0.25	1.03143	0.09
4	0.99872	0.99968	0.10	0.99878	0.01
5	0.98256	0.98439	0.19	0.98310	0.05
6	0.99015	0.99055	0.04	0.99024	0.01
7	0.99967	0.99939	0.03	0.99976	0.01
8	1.01565	1.01444	0.12	1.01509	0.06
9	1.01566	1.01461	0.10	1.01522	0.04
10	0.99020	0.99058	0.04	0.99031	0.01
11	0.98733	0.98773	0.04	0.98748	0.02
12	0.98944	0.98966	0.02	0.98964	0.02
13	0.99013	0.99054	0.04	0.99024	0.01
14	0.99860	0.99946	0.09	0.99861	0.00
15	1.03227	1.02987	0.23	1.03151	0.07
16	0.99969	0.99941	0.03	0.99986	0.02
17	0.98936	0.98970	0.03	0.98970	0.03
18	0.98725	0.98773	0.05	0.98750	0.03
19	0.98252	0.98436	0.19	0.98310	0.06
20	1.01562	1.01454	0.11	1.01522	0.04
21	0.99016	0.99055	0.04	0.99034	0.02
22	0.98723	0.98771	0.05	0.98754	0.03
23	0.98937	0.98963	0.03	0.98971	0.03
24	0.99008	0.99052	0.04	0.99032	0.02
25	0.99856	0.99954	0.10	0.99869	0.01
26	0.99859	0.99961	0.10	0.99873	0.01
27	0.98252	0.98433	0.18	0.98318	0.07
28	0.99021	0.99049	0.03	0.99035	0.01
29	0.99971	0.99935	0.04	0.99988	0.02
30	1.01561	1.01446	0.11	1.01522	0.04
31	0.99862	0.99929	0.07	0.99874	0.01
32	1.01564	1.01445	0.12	1.01524	0.04
33	1.03234	1.02976	0.25	1.03154	0.08

Table- 4.11: Pin wise fission density distribution obtained by MOC code using various number of meshes with linear source assumption for the HTTR benchmark problem and their comparison with reference results (Without geometry correction)

		Linear source assumption			
		1,764 mesh		5,027 mesh	
Pin	MCNP5	Fission density	Relative % error	Fission density	Relative % error
1	0.99862	0.99978	0.12	0.99868	0.01
2	1.01562	1.01482	0.08	1.01523	0.04
3	1.03231	1.03053	0.17	1.03163	0.07
4	0.99872	0.99978	0.11	0.99873	0.00
5	0.98256	0.98393	0.14	0.98296	0.04
6	0.99015	0.99027	0.01	0.99014	0.00
7	0.99967	0.99935	0.03	0.99974	0.01
8	1.01565	1.01480	0.08	1.01523	0.04
9	1.01566	1.01485	0.08	1.01530	0.04
10	0.99020	0.99029	0.01	0.99018	0.00
11	0.98733	0.98731	0.00	0.98734	0.00
12	0.98944	0.98933	0.01	0.98953	0.01
13	0.99013	0.99025	0.01	0.99016	0.00
14	0.99860	0.99972	0.11	0.99869	0.01
15	1.03227	1.03056	0.17	1.03173	0.05
16	0.99969	0.99938	0.03	0.99982	0.01
17	0.98936	0.98934	0.00	0.98957	0.02
18	0.98725	0.98729	0.00	0.98736	0.01
19	0.98252	0.98390	0.14	0.98301	0.05
20	1.01562	1.01483	0.08	1.01535	0.03
21	0.99016	0.99027	0.01	0.99025	0.01
22	0.98723	0.98730	0.01	0.98741	0.02
23	0.98937	0.98931	0.01	0.98960	0.02
24	0.99008	0.99024	0.02	0.99023	0.02
25	0.99856	0.99973	0.12	0.99875	0.02
26	0.99859	0.99973	0.11	0.99882	0.02
27	0.98252	0.98390	0.14	0.98309	0.06
28	0.99021	0.99024	0.00	0.99027	0.01
29	0.99971	0.99930	0.04	0.99986	0.02
30	1.01561	1.01475	0.08	1.01535	0.03
31	0.99862	0.99970	0.11	0.99883	0.02
32	1.01564	1.01476	0.09	1.01538	0.03
33	1.03234	1.03046	0.18	1.03178	0.05

Table- 4.12: Pin wise fission density distribution obtained by MOC code using various number of meshes with flat source assumption for the HTTR benchmark problem and their comparison with reference results (With geometry correction)

		Flat source assumption			
		1,764 mesh		5,027 mesh	
Pin	MCNP5	Fission density	Relative % error	Fission density	Relative % error
1	0.99862	0.99843	0.02	0.99867	0.01
2	1.01562	1.01511	0.05	1.01534	0.03
3	1.03231	1.03113	0.11	1.03186	0.04
4	0.99872	0.99879	0.01	0.99874	0.00
5	0.98256	0.98331	0.08	0.98292	0.04
6	0.99015	0.99040	0.03	0.99017	0.00
7	0.99967	0.99977	0.01	0.99981	0.01
8	1.01565	1.01504	0.06	1.01533	0.03
9	1.01566	1.01524	0.04	1.01537	0.03
10	0.99020	0.99043	0.02	0.99019	0.00
11	0.98733	0.98769	0.04	0.98735	0.00
12	0.98944	0.98981	0.04	0.98956	0.01
13	0.99013	0.99038	0.03	0.99017	0.00
14	0.99860	0.99854	0.01	0.99864	0.00
15	1.03227	1.03126	0.10	1.03185	0.04
16	0.99969	0.99981	0.01	0.99982	0.01
17	0.98936	0.98985	0.05	0.98956	0.02
18	0.98725	0.98769	0.04	0.98737	0.01
19	0.98252	0.98327	0.08	0.98294	0.04
20	1.01562	1.01518	0.04	1.01534	0.03
21	0.99016	0.99042	0.03	0.99018	0.00
22	0.98723	0.98769	0.05	0.98738	0.02
23	0.98937	0.98979	0.04	0.98959	0.02
24	0.99008	0.99038	0.03	0.99022	0.01
25	0.99856	0.99862	0.01	0.99870	0.01
26	0.99859	0.99873	0.01	0.99862	0.00
27	0.98252	0.98328	0.08	0.98293	0.04
28	0.99021	0.99037	0.02	0.99021	0.00
29	0.99971	0.99977	0.01	0.99986	0.02
30	1.01561	1.01510	0.05	1.01540	0.02
31	0.99862	0.99841	0.02	0.99867	0.01
32	1.01564	1.01511	0.05	1.01538	0.03
33	1.03234	1.03118	0.11	1.03188	0.04

Table- 4.13: Pin wise fission density distribution obtained by MOC code using various number of meshes with flat and linear source assumption for the HTTR benchmark problem and their comparison with reference results (With geometry correction)

		Linear source assumption			
		1,764 mesh		5,027 mesh	
Pin	MCNP5	Fission density	Relative % error	Fission density	Relative % error
1	0.99862	0.99885	0.02	0.99869	0.01
2	1.01562	1.01543	0.02	1.01545	0.02
3	1.03231	1.03192	0.04	1.03206	0.02
4	0.99872	0.99889	0.02	0.99870	0.00
5	0.98256	0.98282	0.03	0.98278	0.02
6	0.99015	0.99011	0.00	0.99007	0.01
7	0.99967	0.99973	0.01	0.99979	0.01
8	1.01565	1.01541	0.02	1.01547	0.02
9	1.01566	1.01548	0.02	1.01546	0.02
10	0.99020	0.99013	0.01	0.99007	0.01
11	0.98733	0.98727	0.01	0.98721	0.01
12	0.98944	0.98947	0.00	0.98944	0.00
13	0.99013	0.99009	0.00	0.99009	0.00
14	0.99860	0.99880	0.02	0.99873	0.01
15	1.03227	1.03196	0.03	1.03207	0.02
16	0.99969	0.99977	0.01	0.99979	0.01
17	0.98936	0.98948	0.01	0.98944	0.01
18	0.98725	0.98725	0.00	0.98724	0.00
19	0.98252	0.98278	0.03	0.98284	0.03
20	1.01562	1.01547	0.01	1.01547	0.01
21	0.99016	0.99012	0.00	0.99009	0.01
22	0.98723	0.98726	0.00	0.98724	0.00
23	0.98937	0.98946	0.01	0.98948	0.01
24	0.99008	0.99009	0.00	0.99013	0.01
25	0.99856	0.99883	0.03	0.99876	0.02
26	0.99859	0.99886	0.03	0.99871	0.01
27	0.98252	0.98281	0.03	0.98284	0.03
28	0.99021	0.99011	0.01	0.99013	0.01
29	0.99971	0.99973	0.00	0.99984	0.01
30	1.01561	1.01540	0.02	1.01553	0.01
31	0.99862	0.99884	0.02	0.99876	0.01
32	1.01564	1.01544	0.02	1.01552	0.01
33	1.03234	1.03191	0.04	1.03213	0.02

4.5 Results and discussion

The results presented above show that, in general, a considerably smaller number of meshes is required with the linear source assumption to match the accuracy achieved with the flat source assumption. In order to compare the computation time [recorded for only the inner-outer iteration on a 64 bit machine with intel i5 processor (clock speed 2.5 GHz and memory (RAM) 12 GB)] required for both the assumptions to achieve the same level of accuracy, the KNK-II benchmark problem is taken up as a test case. As evident from Table- 4.14, for a fixed number of meshes in the benchmark problem, linear source assumption takes about 2-3 times of the computation time taken by the flat source assumption due to additional calculation of higher order source moments and expansion coefficients. However, the computation time required for the flat source assumption (with 11,628 meshes) is increased by a factor of 7 to reach an accuracy level close to that achieved by the linear source assumption with only 468 meshes. From this comparison, it can be concluded that the linear source representation reduces the computation time significantly without sacrificing the accuracy.

Table- 4.14: Comparison of computation time with flat and linear source assumption for the KNK-II benchmark problem (with non-reentrant boundary)

No. of meshes	k_{eff} calculated by MOC code with		Computation time (sec) for	
	Flat source assumption	Linear source assumption	Flat source assumption	Linear source assumption
11628	1.4836363	1.4916295	1456	3691
1882	1.4461575	1.4916604	510	1459
468	1.3539351	1.4911083	209	586

The gradient of the flux across the problem domain is less in problems with reflective boundary conditions (in section 4.4.4, 4.4.5, 4.4.6 and 4.4.7). Therefore, for such problems, there is not much difference in the results obtained using flat and linear representations of the source. Problems with vacuum boundary conditions (in section

4.4.1, 4.4.2 and 4.4.3) have much larger flux gradient across the geometry. Hence, the advantage of using the linear source representation is more manifest in these cases. Apart from the boundary condition, the presence of high neutron absorbing material (like B, Gd, Cd etc.) in the geometry influences the neutron flux a lot. This is evident in the HTTR benchmark problem. In this problem, in spite of the reflective boundary condition we find improvement in the result, using the linear source assumption because the burnable poison introduces a relatively larger flux gradient. Result of LWR benchmark problem with burnable poison also shows little improvement due to the use of the linear source assumption.

Finally, it is seen that in problems involving cylindrical elements (having circular projections as in cases of Hexagonal cell problem with central breeding pin, CANDU-6 annular cell problem and HTTR benchmark problem), the use of coarse meshes with linear representation of the source must also be accompanied by a modification of the triangulation process. This modified procedure brings the meshes in conformity with the circular elements of the geometry. As far as computation time is concerned, the modification needs to be done once for all and that too before starting the iterative calculation. Hence, the little increase in computation would not affect the advantage of the linear source representation significantly.

CHAPTER 5

Krylov Acceleration Technique in Method of Characteristics

5.1 Introduction

In previous chapters, we have discussed about a computer code based on Method of Characteristics (MOC) for solving the neutron transport equation for mainly assembly level lattice calculation with reflective and periodic boundary conditions and to some extent core level calculation with vacuum boundary condition. Performance of the MOC code has also been demonstrated with flat and linear source approximations. Since, neutron transport calculation involves extensive computation, an attempt is made to develop an efficient numerical recipe, which will expedite the solution of transport equation and reduce the computation time. At first, conventional MOC solution of neutron transport equation is transformed into a matrix equation to apply Krylov subspace iteration method for accelerating the solution. It is found that forming the matrix equation explicitly by storing its non-zero elements, even in the most sophisticated and compact formats, requires extremely large computer memory. Hence, an alternate way to apply the Krylov iteration is demonstrated by incorporating the effect of the matrix-based approach into the solution without storing the matrix elements. This computationally viable and novel acceleration technique is used in combination with the existing formalism of flat as well as linear source approximation to solve a number of benchmark problems. The results obtained show significant

improvement in terms of faster convergence of solution over the conventional iteration (Gauss-Seidel like iteration, as discussed in section 3.2) without compromising the accuracy.

The present chapter is organized in the following way. At first, the linear algebra based formalism of MOC, which makes use of the Krylov subspace iteration method, is discussed for various boundary conditions. Thereafter, limitation of the formalism and its solution are described in brief. In order to demonstrate the advantage of Krylov iteration over the inner-outer one, few benchmark problems are solved and performances of these two iterative methods are compared while the accuracy of results are confirmed comparing with that available in literature. Scope of coupling between the Krylov iteration and the linear source approximation, though limited, is also explored in the results section.

5.2 Mathematical formulation of Krylov Subspace Iteration in MOC

In linear algebra, Krylov subspace of order m is the linear subspace spanned by the repeated application of a square matrix A of dimension (n × n) on a vector r_0 of dimension n. Mathematically, this subspace is represented by $\kappa_m(A, r_0) = \text{Span}\{r_0, Ar_0, A^2r_0, \ldots\ldots, A^{m-1}r_0\}$, where m is the order of the subspace. In order to solve a matrix equation A.x=b in an efficient manner, we form a Krylov subspace in which solution x_0 of the matrix equation lies. This subspace is constructed by m number of mutually conjugate vectors which are searched in an iterative manner. Starting from an initial guess of the solution, it is required to calculate some metric in each iteration step to know about the proximity of the solution. In present case, the metric is $r_k = (b - A.x_k)$, which can be looked into as the residue at k-th iteration step. The conjugate vectors are obtained by repeated application of A on this metric r_k. Finally, the solution x_0 of the

matrix equation, being one of the members of the subspace, is expressed as a linear combination of the conjugate vectors. Iterative methods, which work in this way, are called Krylov subspace iteration methods. These methods are mainly used to solve large system of linear equations for which direct methods are found to be computationally expensive. Before applying the Krylov subspace iteration to the MOC, it is necessary to form a matrix equation similar to A.x=b out of the equations (Eq.3.6-3.11) mentioned in Chapter 3.

In Eq.3.7, $\Psi^{out}_{i,j,g}$, i.e. outgoing angular flux of mesh i along a characteristic in direction j, is replaced with the expression of Eq.3.6.

$$\overline{\Psi}_{i,j,g} = \Psi^{in}_{i,j,g}\left(\frac{1-e^{-\Sigma^t_{i,g}\Delta t_{i,j}}}{\Sigma^t_{i,g}\Delta t_{i,j}}\right)+\frac{Q_{i,g}}{\Sigma^t_{i,g}}\left(1-\frac{1-e^{-\Sigma^t_{i,g}\Delta t_{i,j}}}{\Sigma^t_{i,g}\Delta t_{i,j}}\right) \tag{5.1}$$

Further, in Eq.3.8, $\overline{\Psi}_{i,j,n,g}$, i.e. average angular flux for n-th characteristic along direction j in mesh i, is replaced with the above expression.

$$\overline{\overline{\Psi}}_{i,j,g} = \frac{\sum_n\left[\Psi^{in}_{i,j,n,g}\left(\frac{1-e^{-\Sigma^t_{i,g}\Delta t_{i,j,n}}}{\Sigma^t_{i,g}\Delta t_{i,j,n}}\right)+\frac{Q_{i,g}}{\Sigma^t_{i,g}}\left(1-\frac{1-e^{-\Sigma^t_{i,g}\Delta t_{i,j,n}}}{\Sigma^t_{i,g}\Delta t_{i,j,n}}\right)\right]\Delta t_{i,j,n}\Delta w_{j,n}}{\sum_n \Delta t_{i,j,n}\Delta w_{j,n}} \tag{5.2}$$

Putting Eq.5.2 in Eq.3.9, we get the expression of scalar flux.

$$\phi_{i,g} = \sum_j\left(\frac{w_j}{\sum_n \Delta t_{i,j,n}\Delta w_{j,n}}\right)\left[\sum_n\left[\begin{array}{l}\Psi^{in}_{i,j,n,g}\left(\frac{1-e^{-\Sigma^t_{i,g}\Delta t_{i,j,n}}}{\Sigma^t_{i,g}\Delta t_{i,j,n}}\right)\\ +\frac{Q_{i,g}}{\Sigma^t_{i,g}}\left(1-\frac{1-e^{-\Sigma^t_{i,g}\Delta t_{i,j,n}}}{\Sigma^t_{i,g}\Delta t_{i,j,n}}\right)\end{array}\right]\Delta t_{i,j,n}\Delta w_{j,n}\right] \tag{5.3}$$

In the above equation, $\Psi^{in}_{i,j,n,g}$ is the incoming angular flux of mesh i along n-th characteristic in direction j. From angular flux continuity condition, it can be equated with the outgoing angular flux of an adjacent mesh, which is succeeded by mesh i, if

we march along the direction of characteristic. As shown in Eq.3.6, outgoing angular flux of a mesh depends on the incoming angular flux and the source in the mesh. Thus, it is always possible to relate the incoming angular flux of a mesh with that of a mesh on the same characteristic line at the external boundary. Let us consider a mesh i lying on the n-th characteristic in the direction j. If we march along the direction of the characteristic from the external boundary, where the characteristic starts, to the mesh i, then we will encounter a number of meshes which can be tagged as $bs_{j,n}$ (or simply bs, which indicates the mesh at the boundary), …… i_{3-}, i_{2-} and i_{1-} (where the subscript "1-" indicates that the mesh follows i-th mesh on the same characteristic, as shown in Fig. 5.1). From Eq.3.6, we can write following equation for i_{1-} mesh and relate its outgoing angular flux with the incoming angular flux of mesh i.

$$\Psi^{in}_{i,j,n,g} = \Psi^{out}_{i_{1-},j,n,g} = \Psi^{in}_{i_{1-},j,n,g} e^{-\Sigma^{t}_{i_{1-},g}\Delta t_{i_{1-},j,n}} + \frac{Q_{i_{1-},g}}{\Sigma^{t}_{i_{1-},g}}\left(1-e^{-\Sigma^{t}_{i_{1-},g}\Delta t_{i_{1-},j,n}}\right) \tag{5.4}$$

Similarly, for i_{2-} mesh,

$$\Psi^{in}_{i_{1-},j,n,g} = \Psi^{out}_{i_{2-},j,n,g} = \Psi^{in}_{i_{2-},j,n,g} e^{-\Sigma^{t}_{i_{2-},g}\Delta t_{i_{2-},j,n}} + \frac{Q_{i_{2-},g}}{\Sigma^{t}_{i_{2-},g}}\left(1-e^{-\Sigma^{t}_{i_{2-},g}\Delta t_{i_{2-},j,n}}\right) \tag{5.5}$$

Replacing $\Psi^{in}_{i_{1-},j,n,g}$ in Eq.5.5 by Eq.5.4, we get

$$\Psi^{in}_{i,j,n,g} = \Psi^{in}_{i_{2-},j,n,g} e^{-\left(\begin{matrix}\Sigma^{t}_{i_{1-},g}\Delta t_{i_{1-},j,n} \\ +\Sigma^{t}_{i_{2-},g}\Delta t_{i_{2-},j,n}\end{matrix}\right)} + \left[\begin{matrix}\frac{Q_{i_{1-},g}}{\Sigma^{t}_{i_{1-},g}}\left(1-e^{-\Sigma^{t}_{i_{1-},g}\Delta t_{i_{1-},j,n}}\right) \\ +\frac{Q_{i_{2-},g}}{\Sigma^{t}_{i_{2-},g}}\left(1-e^{-\Sigma^{t}_{i_{2-},g}\Delta t_{i_{2-},j,n}}\right)e^{-\Sigma^{t}_{i_{1-},g}\Delta t_{i_{1-},j,n}}\end{matrix}\right] \tag{5.6}$$

If we keep on replacing the incoming angular fluxes of meshes i_{2-}, i_{3-},……. and bs, then we will get the following equation, written in a compact form.

$$\Psi^{in}_{i,j,n,g} = \Psi^{in}_{bs,j,n,g} E_{j,n,g}\left(bs, i_{1-}\right) + \sum_{l=bs}^{i_{1-}} \frac{Q_{l,g}}{\Sigma^{t}_{l,g}} \left(1 - e^{-\Sigma^{t}_{l,g}\Delta t_{l,j,n}}\right) E_{j,n,g}\left(l_{1+}, i_{1-}\right) \qquad (5.7)$$

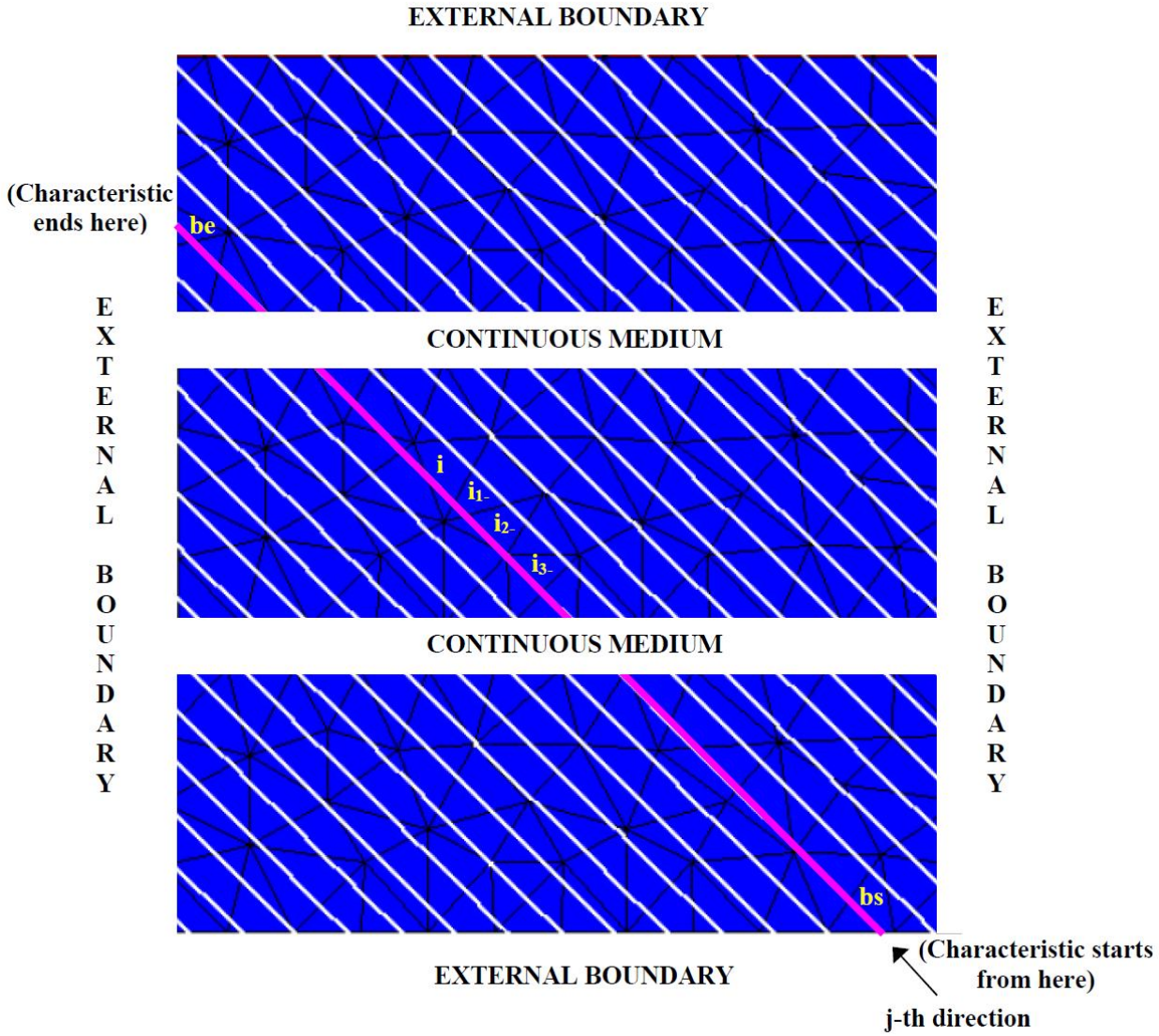

Fig. 5.1: Illustration of the mesh indices used in the derivation of Eq.5.7.

where the function $E_{j,n,g}$ is defined as $E_{j,n,g}\left(m_1, m_2\right) = e^{-\sum_{m=m_1}^{m_2} \Sigma^{t}_{m,g}\Delta t_{m,j,n}}$ if m_1 lies behind m_2 or $m_1 = m_2$ and = 1 otherwise. It is to be noted that i, in the above expression, could be any mesh lying on the n-th characteristic in the direction j (excluding bs). Replacing $\Psi^{in}_{i,j,n,g}$ in Eq.5.3 by Eq.5.7 and after doing some mathematical rearrangement, we get

$$\begin{aligned}&\Sigma_{i,g}^{t}\phi_{i,g}-\left[\sum_{j}C_{j}\sum_{n}R_{i,j,n,g}E_{j,n,g}\left(bs,i_{1-}\right)\right]\Psi_{bs,j,n,g}^{in}\\&=\left[\frac{1}{\Sigma_{i,g}^{t}}\sum_{j}C_{j}\sum_{n}P_{i,j,n,g}\right]Q_{i,g}+\left[\sum_{j}C_{j}\sum_{n}R_{i,j,n,g}\sum_{l=bs}^{i_{1-}}\frac{R_{l,j,n,g}}{\Sigma_{l,g}^{t}}E_{j,n,g}\left(l_{1+},i_{1-}\right)\right]Q_{l,g}\end{aligned}\tag{5.8}$$

where $C_j=\dfrac{w_j}{\sum_n \Delta t_{i,j,n}}$, $P_{i,j,n,g}=\Sigma_{i,g}^{t}\Delta t_{i,j,n}-1+e^{-\Sigma_{i,g}^{t}\Delta t_{i,j,n}}$, $R_{i,j,n,g}=1-e^{-\Sigma_{i,g}^{t}\Delta t_{i,j,n}}$.

This is the general form of the scalar flux equation used for Krylov iteration. If there are total M number of meshes and G number of energy groups, then there will be total M×G (=N_1, say) number of such equations. This set of equations is modified depending on the boundary condition applied at the external boundary of the problem domain.

5.2.1 Vacuum boundary condition

In case of vacuum boundary condition, all the incoming angular fluxes at the boundary ($\Psi_{bs,j,n,g}^{in}$) are zero and Eq.5.8 takes following form.

$$\begin{aligned}&\Sigma_{i,g}^{t}\phi_{i,g}\\&=\left[\frac{1}{\Sigma_{i,g}^{t}}\sum_{j}C_{j}\sum_{n}P_{i,j,n,g}\right]Q_{i,g}+\left[\sum_{j}C_{j}\sum_{n}R_{i,j,n,g}\sum_{l=bs}^{i_{1-}}\frac{R_{l,j,n,g}}{\Sigma_{l,g}^{t}}E_{j,n,g}\left(l_{1+},i_{1-}\right)\right]Q_{l,g}\end{aligned}\tag{5.9}$$

This is the only equation required to calculate scalar flux and multiplication factor for vacuum boundary condition.

5.2.2 Reflective boundary condition

Unlike vacuum boundary condition, $\Psi_{bs,j,n,g}^{in}$ is non zero, which motivates us to construct additional equations to close the set of equations (Eq.5.8). For doing so, $\Psi_{bs,j,n,g}^{in}$ is related with the outgoing angular flux, which is incident on the same

boundary, but from a different direction and reflected in the direction of incoming angular flux.

Starting with Eq.5.7, we can write outgoing angular flux of a mesh "be" which is at the boundary where the n-th characteristic in direction j ends (Fig. 5.1).

$$\Psi^{out}_{be,j,n,g} = \Psi^{in}_{bs,j,n,g} E_{j,n,g}\left(bs,be\right) + \sum_{l=bs}^{be} \frac{Q_{l,g}}{\Sigma^{t}_{l,g}} \left(1 - e^{-\Sigma^{t}_{l,g}\Delta t_{l,j,n}}\right) E_{j,n,g}\left(l_{1+},be\right) \qquad (5.10)$$

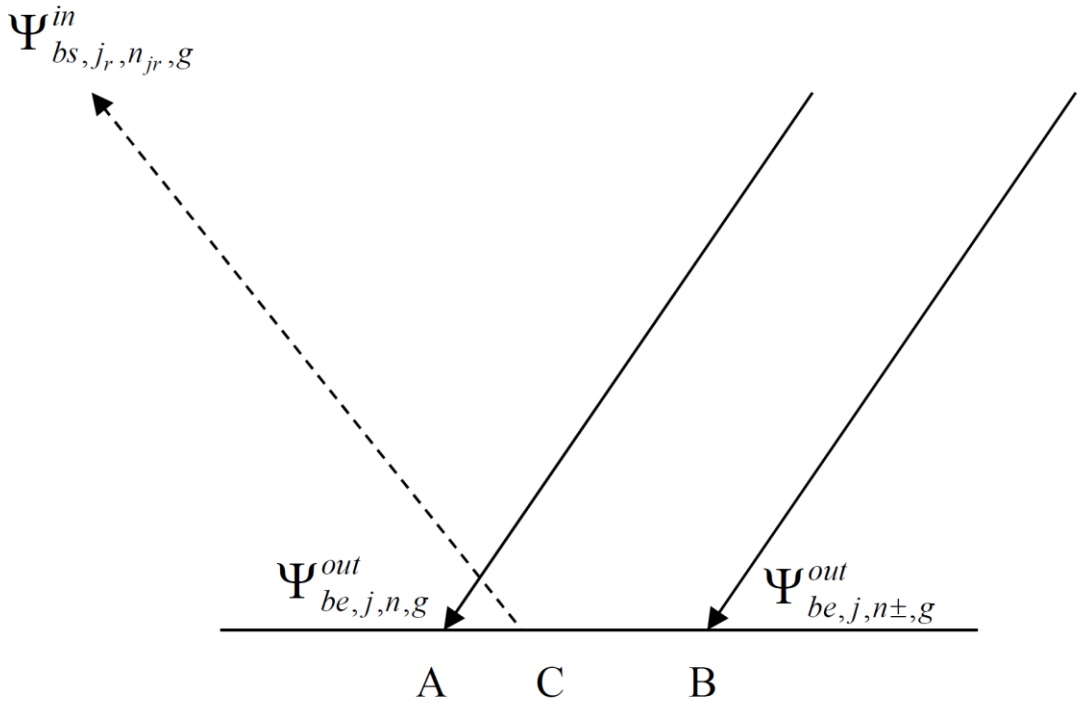

Fig. 5.2: Illustration of the use of reflective boundary condition in Method of Characteristic (MOC) based Neutron Transport Code.

Simulation of reflective boundary condition is illustrated in Fig. 5.2, which shows two consecutive characteristics in direction j, namely n and n± (±indicates that it is next or previous to n-th characteristic) are incident at A and B points and n_{jr}-th characteristic (since the index "n" is not necessarily same for the incident and reflected characteristic in the respective directions) in the reflected direction j_r starts from C point. If AC = x

and AB = d, then we can write the following expression of the outgoing angular flux at point C using linear interpolation scheme between A and B points.

$$\Psi^{out}_{C,j,g} = (1-\beta)\Psi^{out}_{be,j,n,g} + \beta\Psi^{out}_{be,j,n\pm,g} \tag{5.11}$$

where $\beta = x/d$. $\Psi^{out}_{C,j,g}$ is related with the incoming angular flux along the n_{jr}-th characteristic in the direction j_r through following boundary condition.

$$\Psi^{in}_{bs,j_r,n_{jr},g} = \alpha_b \Psi^{out}_{C,j,g} \tag{5.12}$$

where α_b is the fraction of outgoing angular flux reflected in the direction of incoming angular flux. This is 1.0 for reflective boundary condition and <1.0 for albedo boundary condition. Using Eq.5.10 and 5.11, Eq.5.12 can be rewritten as

$$\begin{aligned}&\Psi^{in}_{bs,j_r,n_{jr},g} - \left[\alpha_b(1-\beta)E_{j,n,g}(bs,be)\right]\Psi^{in}_{bs,j,n,g} - \left[\alpha_b\beta E_{j,n\pm,g}(bs,be)\right]\Psi^{in}_{bs,j,n\pm,g} \\ &= \alpha_b(1-\beta)\sum_{l=bs}^{be}\frac{R_{l,j,n,g}}{\Sigma^t_{l,g}}E_{j,n,g}(l_{1+},be)Q_{l,g} + \alpha_b\beta\sum_{l=bs}^{be}\frac{R_{l,j,n\pm,g}}{\Sigma^t_{l,g}}E_{j,n\pm,g}(l_{1+},be)Q_{l,g}\end{aligned} \tag{5.13}$$

If there are N_θ number of θ directions, N_φ number of φ directions and N_l number of parallel characteristics in each (θ, φ) direction, then there will be total $N_\theta \times N_\varphi \times N_l \times G$ (=N_2, say) number of such equations. Eq.5.8 and 5.13 are combined together to form the closed set of equations for reflective boundary condition.

5.3 Numerical recipe for Krylov subspace iteration

It is now possible to club the set of equations, i.e. Eq.5.9 for vacuum boundary; Eq.5.8 and 5.13 for reflective boundary, to form following matrix equation.

$$[A]_{N_3\times N_3}[\psi]_{N_3\times 1} = [q]_{N_3\times 1} \tag{5.14}$$

where $N_3=N_1+N_2$. For reflective boundary condition, $\psi = \begin{pmatrix} \left[\phi_{i,g}\right]_{N_1\times 1} \\ \left[\Psi^{in}_{bs,j,n_j,g}\right]_{N_2\times 1} \end{pmatrix}$, a column matrix containing scalar fluxes of all energy groups in all meshes and angular fluxes of all energy groups along all characteristics incoming to all boundary meshes. For vacuum boundary condition, incoming angular fluxes are zero. So, $\psi = \left[\phi_{i,g}\right]_{N_1\times 1}$. Before forming the above matrix equation, scattering sources on the RHS of respective equations are transferred to LHS. Hence, the column matrix "q" contains only the fission sources. "A" is a sparse, non-symmetric matrix constituted by the coefficients of $\phi_{i,g}$ and $\Psi^{in}_{bs,j,n_j,g}$. Its sparsity mainly depends on the interconnection between the meshes through the common characteristics they are sharing.

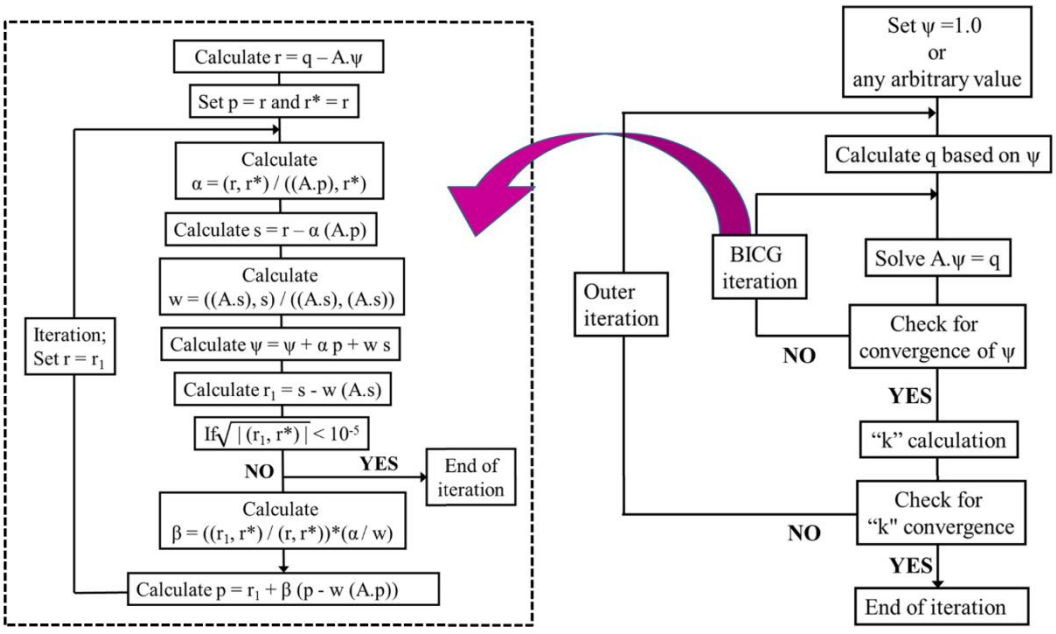

Fig. 5.3: Flowchart of MOC solution technique of neutron transport equation with Krylov iteration.

Biconjugate gradient or BICG is one kind of Krylov subspace iteration method, which is used for the numerical solution of non-symmetric linear system like Eq.5.14.

However, it suffers from the stability of convergence issue. Hence, Biconjugate Gradient stabilized method (BiCGSTAB), a variant of BICG, is selected to solve the Eq.5.14 [77]. As shown in Fig. 5.3, there will be nested loops of outer and BICG iterations. Before starting the iterations, ψ is initialized to 1.0 or any arbitrary value. The outer iteration starts by calculating q based on ψ. Once q is fixed, Eq.5.14 is solved by BiCGSTAB in an iterative manner. Each of these iterations is termed here as BICG iteration (details are shown in Fig. 5.3), which is terminated when the solution converges within a specified error limit or the number of BICG iterations per outer iteration reaches a specified upper limit, whichever is earlier. Reason behind the choice of such criterion has already been explained in Chapter 3. With the converged flux distribution obtained as the solution, multiplication factor or k is estimated. If this k value converges within a specified limit, then the outer iteration will be terminated. Otherwise, a new outer iteration will be started with a better estimate of q calculated based on latest ψ. This iteration technique is computationally faster than the conventional iteration. However, explicit formation of the matrix equation (Eq.5.14) may not be needed since this requires a lot of memory for a realistic problem even if we store the matrix elements in Compressed Sparse Row (CSR) format [53]. This motivates us to work on a recipe which will take the effect of matrix A on any vector (like ψ here) by employing the mesh-angle sweep, instead of forming the matrix explicitly [52][78]. This recipe, to the best of our knowledge, is used for the first time in Krylov accelerated MOC.

In Eq.5.14, N_1 number of scalar flux equations and N_2 number of angular flux equations are combined together. The scalar flux equation i.e. Eq.5.8 is derived from the definition given in Eq.3.9. After transferring the scattering sources from RHS to

LHS of Eq.5.8, the final matrix equation Aψ=q is constructed. The way scalar flux equation is converted into a matrix form suggests that the multiplication of the matrix A with the vector ψ is identical to the calculation of following quantity for all the meshes and energy groups.

$$\left[A\psi\right]_{m<N_1} = \left[\sum_{n=1}^{N_3} A_{mn}\psi_n\right]_{m<N_1} = \left[\psi_m\right]_{m<N_1} - \sum_j w_j \overline{\overline{\Psi}}_{i,j,g} \tag{5.15}$$

where "m" value depends on the storage location of $\phi_{i,g}$ in ψ. $\overline{\overline{\Psi}}_{i,j,g}$ is the same quantity as defined in Eq.3.8 except the fact that the total source $Q_{i,g}$, while calculating $\overline{\Psi}_{i,j,n,g}$ and $\Psi^{out}_{i,j,n,g}$, does not include the fission source. Hence,

$$\overline{\Psi}_{i,j,g} = \frac{q^s_{i,g}}{\Sigma^t_{i,g}} + \frac{\Psi^{in}_{i,j,n,g} - \Psi^{out}_{i,j,n,g}}{\Sigma^t_{i,g}\Delta t_{i,j,n}} \tag{5.16}$$

$$\Psi^{out}_{i,j,n,g} = \Psi^{in}_{i,j,n,g} e^{-\Sigma^t_{i,g}\Delta t_{i,j,n}} + \frac{q^s_{i,g}}{\Sigma^t_{i,g}}\left(1 - e^{-\Sigma^t_{i,g}\Delta t_{i,j,n}}\right) \tag{5.17}$$

Using Eq.5.17, $\Psi^{out}_{i,j,n,g}$ is calculated for all the meshes sitting on characteristic. This calculation starts from the boundary mesh "bs" (Fig. 5.1), where value of $\Psi^{in}_{i,j,n,g}$ depends on the boundary condition. For vacuum boundary, it is 0 while for reflective boundary, it is ≠ 0. The non-zero value corresponds to $\Psi^{in}_{bs,j,n,g}$ stored at a location "m′" in ψ.

$$\Psi^{out}_{bs,j,n,g} = \left[\psi_{m'}\right]_{m'>N_1} e^{-\Sigma^t_{bs,g}\Delta t_{bs,j,n}} + \frac{q^s_{bs,g}}{\Sigma^t_{bs,g}}\left(1-e^{-\Sigma^t_{bs,g}\Delta t_{bs,j,n}}\right) \quad \left[\text{Reflective}\right] \qquad (5.18)$$

$$\Psi^{out}_{bs,j,n,g} = \frac{q^s_{bs,g}}{\Sigma^t_{bs,g}}\left(1-e^{-\Sigma^t_{bs,g}\Delta t_{bs,j,n}}\right) \quad \left[\text{Vacuum}\right] \qquad (5.19)$$

The calculation of $\Psi^{out}_{i,j,n,g}$, mentioned above, ends at another boundary mesh "be" (Fig. 5.1). For reflective boundary condition, this outgoing angular flux can be related to incoming angular flux in some other direction (Eq.5.11 and 5.12). The difference between $\Psi^{in}_{bs,j,n,g}$ stored in ψ and the calculated one gives the multiplication of the matrix A with the vector ψ for $m>N_1$.

$$\left[A\psi\right]_{m>N_1} = \left[\sum_{n=1}^{N_3} A_{mn}\psi_n\right]_{m>N_1} = \left[\psi_m\right]_{m>N_1} - \Psi^{in}_{bs,j,n,g} \qquad (5.20)$$

where "m" stands for the storage location of $\Psi^{in}_{bs,j,n,g}$. Other than ψ, the matrix A is multiplied with vectors p and s, as shown in Fig. 5.3. In those cases, the recipe of matrix multiplication will remain same as above (Eq.5.15-5.20). In stabilized BICG iteration, residue "r" is calculated as the difference between q and Aψ. Since, calculating q again and again is time consuming, it is better to calculate the residue directly by modifying the above recipe. Previously, fission sources were not accounted for the calculation of $\overline{\Psi}_{i,j,n,g}$ and $\Psi^{out}_{i,j,n,g}$. In the residue calculation, Aψ is transferred from LHS to RHS. Therefore, the total source $Q_{i,g}$, in present case, includes both fission and scattering sources and Eq.5.15 and 5.20 are modified as

$$\left[A\psi\right]_{m<N_1} = \left[\sum_{n=1}^{N_3} A_{mn}\psi_n\right]_{m<N_1} = -\left[\psi_m\right]_{m<N_1} + \sum_j w_j \overline{\overline{\Psi}}_{i,j,g} \tag{5.21}$$

$$\left[A\psi\right]_{m>N_1} = \left[\sum_{n=1}^{N_3} A_{mn}\psi_n\right]_{m>N_1} = -\left[\psi_m\right]_{m>N_1} + \Psi^{in}_{bs,j,n,g} \tag{5.22}$$

respectively.

5.4 Verification and benchmarking of the code

In order to validate the MOC based neutron transport code accelerated by Krylov iteration technique for calculating multiplication factor (k) and neutron flux (or power), a number of benchmark problems in square as well as hexagonal geometries are analyzed. In this section, an attempt is made to establish the efficiency of Krylov iteration technique over the conventional iteration technique. We have already introduced the concept of linear source, which expands the neutron source within a mesh up to linear term. This expansion helps to divide the problem domain into coarse meshes and thereby reduces the computation time without compromising the accuracy. However, advantage of linear source is more manifest in case of large flux gradient observed in problems with vacuum boundary. Hence, it is tried to combine Krylov iteration with linear source assumption and apply the same to reduce the computation time to great extent at least for problems with vacuum boundary condition. The computation time, reported below, is recorded only for the iterative solution on a 64 bit machine with intel i5 processor (clock speed 2.5 GHz and memory (RAM) 12 GB). Convergence criteria for the neutron flux and the multiplication factor are chosen to be 10^{-5} and 10^{-7} respectively.

5.4.1 MZA fast reactor benchmark problem

Assuming flat source within a mesh, the reference core is divided into 19,782 meshes and k_{eff} is calculated using both conventional and Krylov based inner-outer iteration. As explained in Chapter 3, number of inner iterations per outer iteration influences computation time, which is studied for this problem. Computation time is nothing but the total number of outer iterations, multiplied by the average time taken by single outer iteration. If we consider very few number of inner per outer iteration, then the time taken by an outer iteration will be less, but the total number of outer iterations to be required to reach the convergence criterion will be more. Again, more number of inner per outer iteration makes total number of outer iterations less, but at the expense of more computation time for single outer iteration. Hence, in order to minimize the total computation time, the number of inner per outer iteration is to be optimized, as demonstrated in Table- 5.1. The optimization study shows that the computation time decreases with the increase in number of inner per outer iteration from 1 to 20 after which the time increases. In the study, similar variation of computation time with number of BICG iteration per outer iteration is also found. In case of conventional iteration, minimum computation time is obtained for 20 inner per outer iteration and the same, in case of Krylov iteration, is obtained for 2 BICG per outer iteration. As far as the computation time is concerned, Krylov iteration takes only 370 sec while conventional iteration takes 1243 sec to produce almost identical k_{eff}. Fig. 5.4 shows the convergence of flux (ε_ϕ[2]) and multiplication factor (ε_k[3]) with iteration step as well as computation time in both the iteration techniques. This computation time is reduced

[2] $\varepsilon_\phi = \left|\phi_{n+1} - \phi_n\right|$, n is index for outer iteration

[3] $\varepsilon_k = \left|k_{n+1} - k_n\right|$, n is index for outer iteration

further by a factor of 2 (conventional)/3 (Krylov) by assuming the source in a mesh to be linear and thereby reducing the number of meshes from 19,782 to 800. In Table- 5.2, k_{eff} values calculated by MOC code are compared with the values calculated by GMVP [76] and MOCUM [22]. The comparison shows that the values calculated for linear source assumption are closer (Flat source assumption: $\Delta\rho_{max}$[4] ~ 6.5 mk, Linear source assumption: $\Delta\rho_{max}$ ~ 0.9 mk) to the values calculated by GMVP and MOCUM. $\Delta\rho_{max}$ is further reduced to 0.3 mk by increasing the number of meshes from 800 to 3,100 with linear source assumption. However, such improvement in accuracy comes at the cost of increased computation time. Considering the fact that the same accuracy in flat source assumption will be possible if we take number of meshes more than what we have considered and therefore, the computation time will be much more, it can be said that the performance of linear assumption is encouraging. Similar trend has been found in rest three core configurations and all the results are given in Table- 5.2.

[4] $\Delta\rho_{max} = \left[\left(\frac{1}{k_{eff}} \right)_{\substack{\text{our} \\ \text{code}}} - \left(\frac{1}{k_{eff}} \right)_{\substack{\text{other} \\ \text{code}}} \right]_{max}$

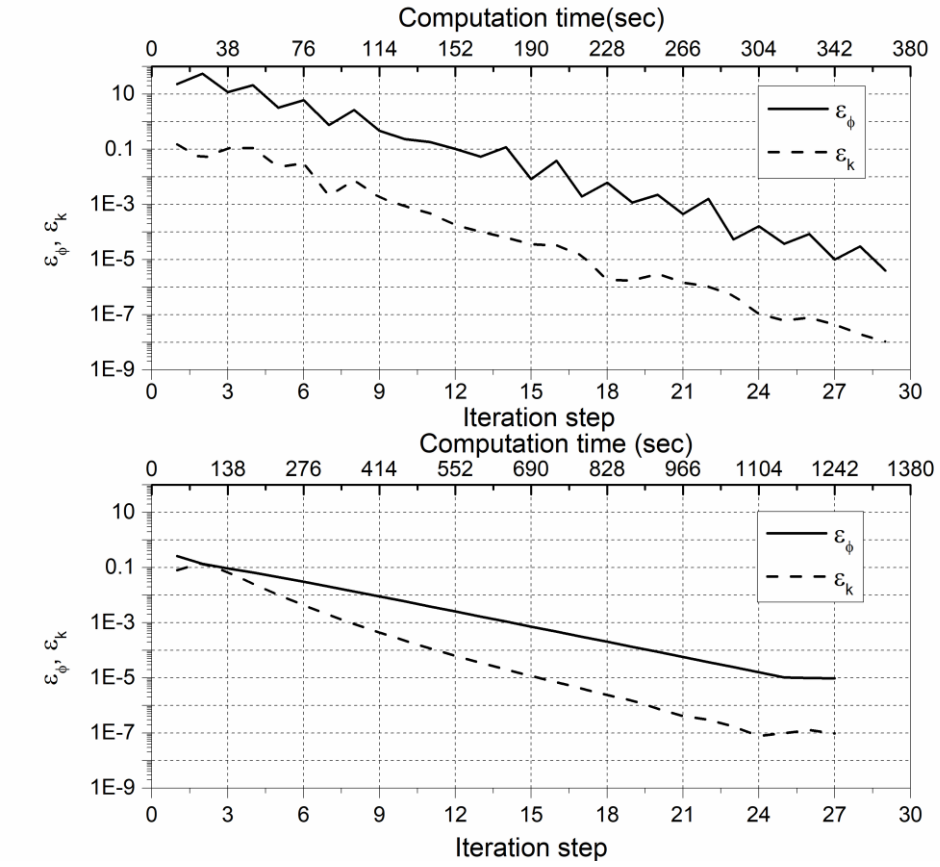

Fig. 5.4: Convergence of neutron flux (ε_ϕ) and multiplication factor (ε_k) of MZA fast reactor benchmark problem with iteration step and computation time in Krylov iteration (at top) and conventional iteration (at bottom).

Table- 5.1: Optimization of iteration scheme for obtaining minimum computation time in conventional and Krylov iteration for MZA fast reactor benchmark problem

Type of iteration	Iteration scheme	Computation time (s)	No of outer iteration	k_{eff} calculated by MOC code
Conventional	1 inner/outer	1524	506	1.1774788
	5 inner/outer	1323	101	1.1774779
	15 inner/outer	1247	34	1.1774773
	20 inner/outer	1243	27	1.1774775
	30 inner/outer	1325	20	1.1774780
Krylov	1 BICG/outer	436	58	1.1774789
	2 BICG/outer	370	29	1.1774789
	3 BICG/outer	392	23	1.1774789

Table- 5.2: k_{eff} values obtained by MOC code using various number of meshes, source assumption and iteration scheme for MZA fast reactor benchmark problem and their comparison with reference results

Core configuration	No. of mesh	Source assumption	Type of iteration	Iteration scheme	Computation time (s)	No. of outer iteration	k_{eff} calculated by MOC code	k_{eff} calculated by other codes (Name of code)
Ref. core	19,782	Flat	Conventional	20 inner/outer	1243	27	1.1774775	1.18649 (GMVP), 1.18654 (MOCUM)
	19,782	Flat	Krylov	2 BICG/outer	370	29	1.1774789	
	800	Linear	Conventional	5 inner/outer	622	108	1.1852796	
	800	Linear	Krylov	2 BICG/outer	125	29	1.1852804	
	3,100	Linear	Conventional	5 inner/outer	1406	109	1.1861610	
	3,100	Linear	Krylov	2 BICG/outer	326	30	1.1861619	
Void at centre	19,782	Flat	Conventional	20 inner/outer	1194	28	1.1860272	1.19489 (GMVP), 1.19502 (MOCUM)
	19,782	Flat	Krylov	2 BICG/outer	358	29	1.1860285	
	808	Linear	Conventional	5 inner/outer	634	108	1.1936735	
	808	Linear	Krylov	2 BICG/outer	125	29	1.1936743	
	3,120	Linear	Conventional	5 inner/outer	1375	109	1.1945209	
	3,120	Linear	Krylov	2 BICG/outer	309	33	1.1945216	
Void at horizontal boundaries	19,794	Flat	Conventional	20 inner/outer	1225	26	1.1711410	1.17990 (GMVP), 1.17995 (MOCUM)
	19,794	Flat	Krylov	2 BICG/outer	372	29	1.1711424	
	818	Linear	Conventional	5 inner/outer	637	108	1.1788262	
	818	Linear	Krylov	2 BICG/outer	143	32	1.1788270	
	3,114	Linear	Conventional	5 inner/outer	1351	109	1.1795822	
	3,114	Linear	Krylov	2 BICG/outer	272	29	1.1795830	
Void at vertical boundaries	19,782	Flat	Conventional	20 inner/outer	1224	26	1.1711403	1.17988 (GMVP), 1.17995 (MOCUM)
	19,782	Flat	Krylov	2 BICG/outer	408	33	1.1711416	
	818	Linear	Conventional	5 inner/outer	647	108	1.1788314	
	818	Linear	Krylov	2 BICG/outer	122	28	1.1788323	
	3,104	Linear	Conventional	5 inner/outer	1352	109	1.1795861	
	3,104	Linear	Krylov	2 BICG/outer	308	34	1.1795869	

5.4.2 KNK-II benchmark problem with non-reentrant boundary

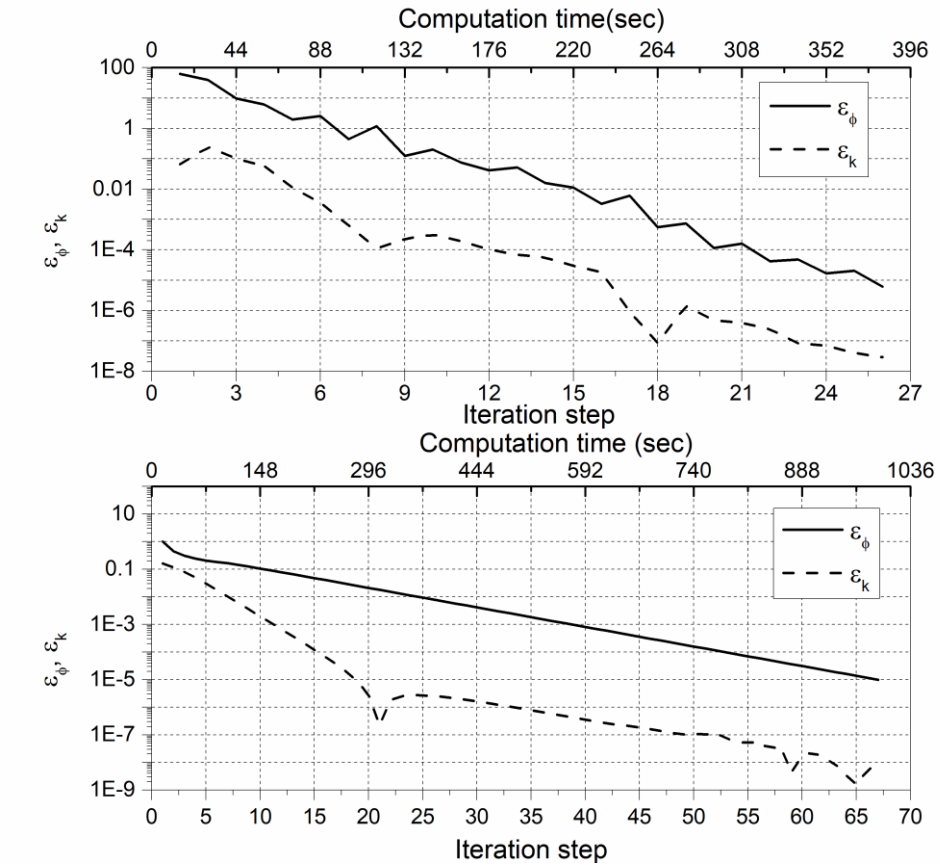

Fig. 5.5: Convergence of neutron flux (ε_ϕ) and multiplication factor (ε_k) of KNK-II benchmark problem with iteration step and computation time in Krylov iteration (at top) and conventional iteration (at bottom).

With flat and linear source assumption, k_{eff} is calculated using both conventional and Krylov iteration. For flat source assumption, the entire hexagonal core is divided into 11,622 meshes so that the size of mesh becomes small enough for the assumption to be hold. In order to achieve same order of accuracy in k_{eff}, conventional iteration takes 1034 sec with 5 inner per outer iteration scheme while Krylov iteration takes only 388 sec with 2 BICG per outer iteration scheme. Like previous problem, iteration schemes, quoted above, are optimized to get the minimum computation time. Trend of flux and multiplication factor convergence with iteration step and computation time in both the iteration techniques are shown in Fig. 5.5. Further it is attempted to reduce the

time by taking linear source assumption which makes possible to consider only 468 meshes i.e. lesser than the earlier assumption without compromising the accuracy. With this assumption too, Krylov iteration takes lesser time than the conventional iteration to arrive at the converged k_{eff} values. To be precise, conventional iteration takes 604 sec with 5 inner per outer iteration scheme while Krylov iteration takes 168 sec with 2 BICG per outer iteration scheme. k_{eff} values calculated by MOC code are compared with the values calculated by TWOHEX [68], SPANDOM-TA [67] and DIAMANT2 [69]. For flat source assumption, Krylov iteration takes 2.7 times lesser computation time than the conventional iteration. This computation time is reduced further by a factor of 2.3 due to Krylov iteration for linear source assumption. It is clear from the comparison that the values calculated for linear source assumption are closer (Flat source assumption: $\Delta\rho_{max} \sim 3.8$ mk, Linear source assumption: $\Delta\rho_{max} \sim 0.4$ mk) to the values calculated by the above mentioned codes. The values calculated for flat source assumption would have been closer to the values calculated by other codes if more number of meshes are considered. However, this will increase the computation time further and hence, is not shown here. k_{eff} values calculated for all the cases are summarized in Table- 5.3.

Table- 5.3: k_{eff} values obtained by MOC code using various number of meshes, source assumption and iteration scheme for KNK-II benchmark problem and their comparison with reference results

No. of mesh	Source assump-tion	Type of iteration	Iteration scheme	Compu-tation time (s)	No. of outer iteration	k_{eff} calculated by MOC code	k_{eff} calculated by other codes (Name of code)
11,622	Flat	Conventional	5 inner/outer	1034	67	1.4836043	1.49208 (TWOHEX),
11,622	Flat	Krylov	2 BICG/outer	388	26	1.4836046	1.49132
468	Linear	Conventional	5 inner/outer	604	68	1.4911046	(SPANDOM-TA),
468	Linear	Krylov	2 BICG/outer	168	30	1.4911048	1.49205 (DIAMANT2)

5.4.3 IAEA benchmark problem

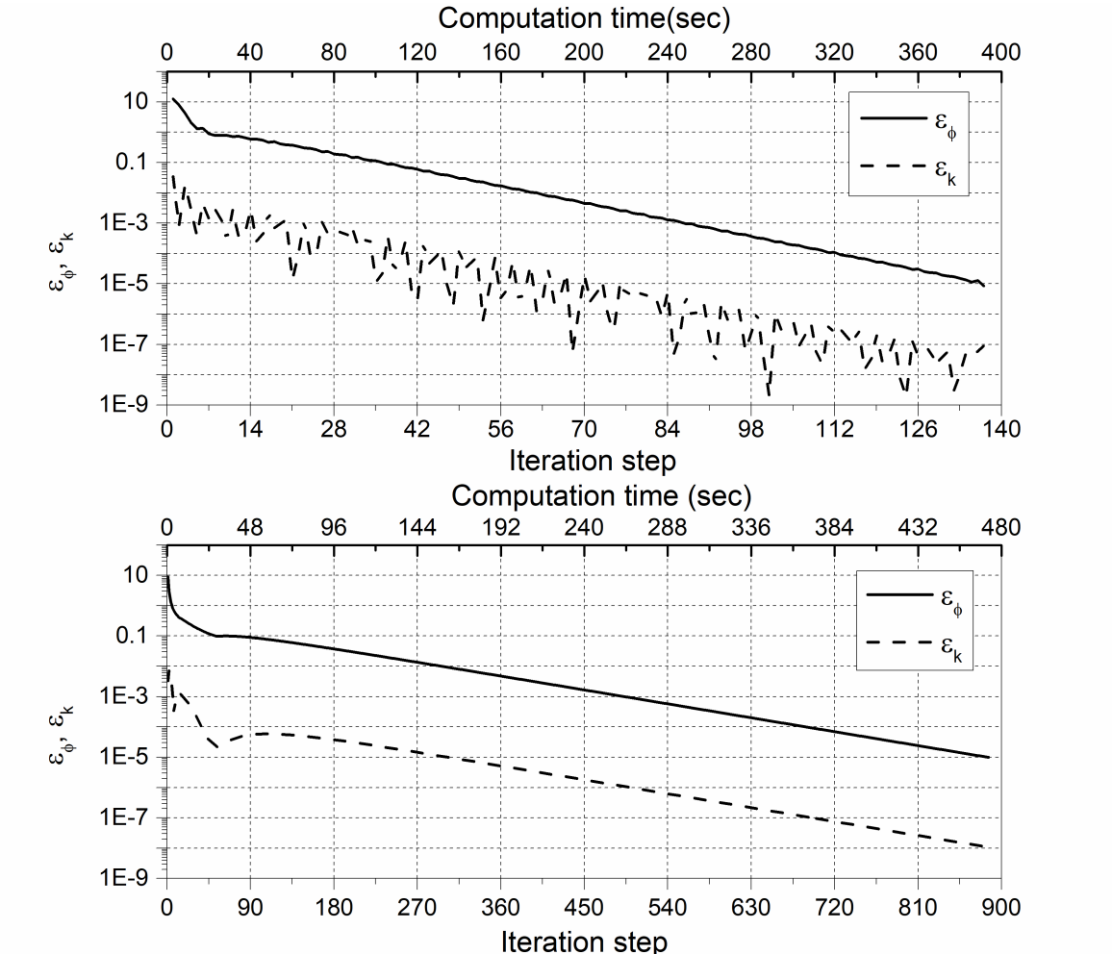

Fig. 5.6: Convergence of neutron flux (ε_ϕ) and multiplication factor (ε_k) of IAEA benchmark problem with iteration step and computation time in Krylov iteration (at top) and conventional iteration (at bottom)

The core is divided into 22,294 meshes in order to apply flat source assumption. Both conventional and Krylov iteration techniques are used to calculate k_{eff} with this assumption. After an optimization study, it is found that minimum computation time is achieved in case of 1 inner per outer iteration scheme for conventional iteration and 2 BICG per outer iteration scheme for Krylov iteration. As shown in the profile of flux and multiplication factor convergence with iteration step and computation time in Fig. 5.6, it is evident that conventional iteration takes 471 sec and Krylov iteration takes 389

sec to reach the converged k_{eff} value. So, there is a reduction in computation time by a factor of 1.2. To reduce the computation time further, linear source assumption is attempted. In this case, the core is divided into only 1,854 meshes, which is lesser than the number of meshes used by flat source assumption. With this assumption, conventional iteration takes 299 sec with 1 inner per outer iteration scheme while Krylov iteration takes 195 sec with 2 BICG per outer iteration scheme. As compared to Krylov iteration in flat source assumption, computation time is now reduced by a factor of 2 for Krylov iteration in linear source assumption. In Table- 5.4, k_{eff} values calculated by MOC code are compared with the values available in literature [SURCU [25], TEPFEM [74] and TPTRI [30]]. It is clear from the comparison that the values calculated for linear source assumption are closer (Flat source assumption: $\Delta\rho_{max} \sim 3.4$ mk, Linear source assumption: $\Delta\rho_{max} \sim 1.5$ mk) to the literature values. Error in k_{eff} value for linear source assumption, though higher than the earlier problems, is acceptable since there is a difference of 1 mk amongst the values reported in literature. It is necessary to verify whether the reduction in computation time, due to the use of Krylov iteration, has an effect on the accuracy of local parameter like neutron flux. Therefore, region averaged scalar flux is calculated for all five regions of the problem. Normalization criterion, given in Eq.4.41, is used to scale the one group flux. In Table-5.5, the scaled flux values are compared with the values calculated by TEPFEM and TPTRI. The comparison clearly indicates that Krylov iteration is as accurate as conventional iteration in both flat as well as linear source assumption.

Table- 5.4: k_{eff} values obtained by MOC code using various number of meshes, source assumption and iteration scheme for IAEA benchmark problem and their comparison with reference results

No. of mesh	Source assump-tion	Type of iteration	Iteration scheme	Compu-tation time (s)	No. of outer iteration	k_{eff} calculated by MOC code	k_{eff} calculated by other codes (Name of code)
22,294	Flat	Conventional	1 inner/outer	471	886	1.0044704	1.0069 (SURCU), 1.0079 (TEPFEM), 1.0070 (TPTRI)
22,294	Flat	Krylov	2 BICG/outer	389	137	1.0044712	
1,854	Linear	Conventional	1 inner/outer	299	908	1.0084456	
1,854	Linear	Krylov	2 BICG/outer	195	139	1.0084464	

Table- 5.5: Region averaged neutron flux values obtained by MOC code using various source assumption and iteration scheme for IAEA benchmark problem and their comparison with reference results

Region	Average flux calculated by		Type of iteration	Flat source assumption			Linear source assumption		
	TEPFEM	TPTRI		Avg. flux calculated by MOC code	Relative % error with respect to		Avg. flux calculated by MOC code	Relative % error with respect to	
					TEPFEM	TPTRI		TEPFEM	TPTRI
Zone 1	0.016860	0.016850	Conventional	0.016856	0.02	0.04	0.016858	0.01	0.05
			Krylov	0.016856	0.02	0.04	0.016858	0.01	0.05
Zone 2	0.000125	0.000130	Conventional	0.000129	3.58	0.40	0.000125	0.26	4.10
			Krylov	0.000129	3.58	0.40	0.000125	0.26	4.10
Zone 3	0.000035	0.000026	Conventional	0.000040	13.66	53.01	0.000037	5.45	41.95
			Krylov	0.000039	12.63	51.62	0.000037	4.44	40.59
Zone 4	0.000297	0.000282	Conventional	0.000306	3.02	8.50	0.000295	0.55	4.74
			Krylov	0.000306	3.02	8.49	0.000295	0.55	4.74
Zone 5	0.000786	0.000766	Conventional	0.000812	3.36	6.05	0.000791	0.69	3.32
			Krylov	0.000812	3.36	6.05	0.000791	0.69	3.32

5.4.4 BWR benchmark problem

In case of reflective boundary condition, flux variation across the problem domain is less as compared to that in vacuum boundary condition. Hence, k_∞ of the BWR lattice is calculated with flat source assumption only. For the said purpose, the lattice is divided into 2,600 meshes. Both conventional and Krylov iteration techniques are used to calculate k_∞. As shown in Table- 5.6 and Fig. 5.7, conventional iteration takes 50 sec with 15 inner per outer iteration scheme while Krylov iteration takes 23 sec with 6 BICG per outer iteration scheme, which suggests 2.2 times reduction in computation time. k_∞ values calculated by MOC code are in good agreement with the values calculated by three codes SURCU, TEPFEM and TPTRI.

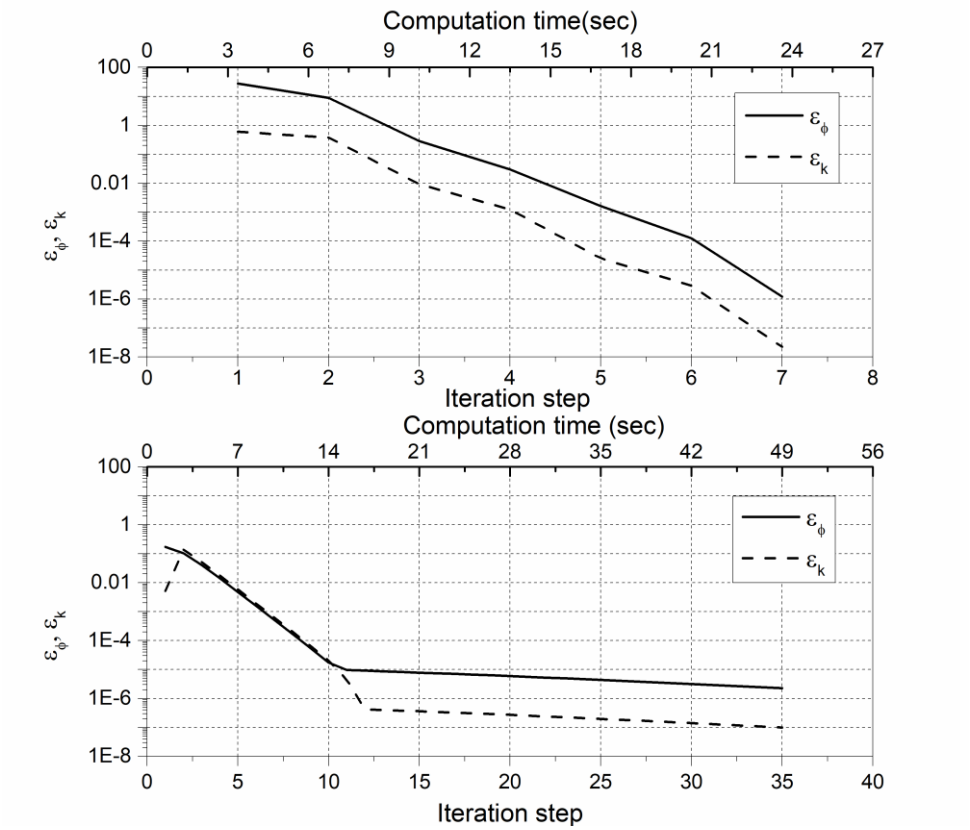

Fig. 5.7: Convergence of neutron flux (ε_ϕ) and multiplication factor (ε_k) of BWR benchmark problem with iteration step and computation time in Krylov iteration (at top) and conventional iteration (at bottom)

Table- 5.6: k_∞ values obtained by MOC code using conventional and Krylov iteration scheme for BWR benchmark problem and their comparison with reference results

No. of mesh	Source assump-tion	Type of iteration	Iteration scheme	Compu-tation time (s)	No. of outer iteration	k_∞ calculated by MOC code	k_∞ calculated by other codes (Name of code)
2,600	Flat	Conventional	15 inner/outer	50	35	1.2121066	1.2127 (SURCU), 1.2136 (TEPFEM), 1.2128 (TPTRI)
2,600	Flat	Krylov	6 BICG/outer	23	7	1.2121080	

5.4.5 LWR benchmark problem with burnable poison

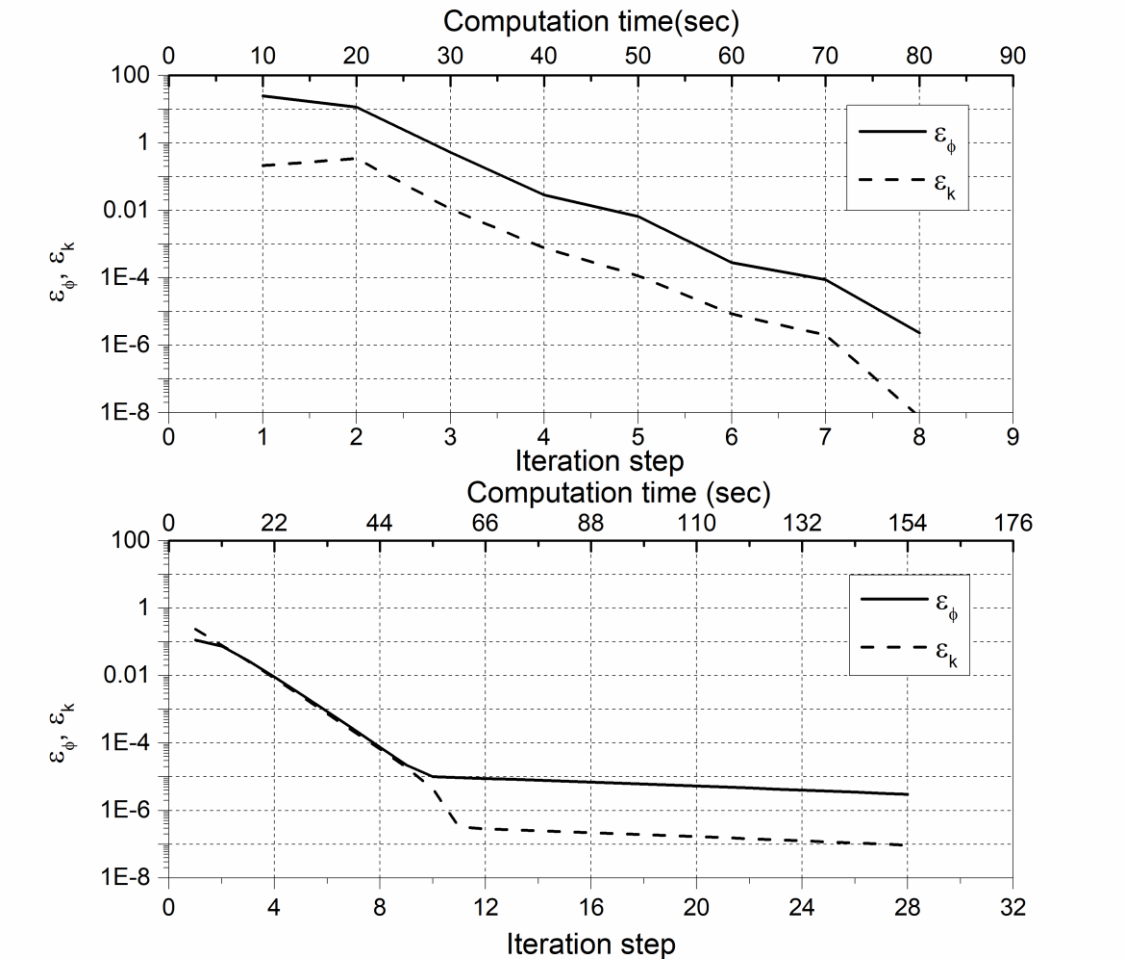

Fig. 5.8: Convergence of neutron flux (ε_ϕ) and multiplication factor (ε_k) of LWR benchmark problem with iteration step and computation time in Krylov iteration (at top) and conventional iteration (at bottom).

With flat source assumption, k_∞ is calculated using both conventional and Krylov iteration. In order to apply the flat source assumption, the LWR lattice is divided into 8,160 meshes. Almost identical k_∞ is obtained in 151 sec for conventional iteration with 15 inner per outer iteration scheme and in 76 sec for Krylov iteration with 5 BICG per outer iteration scheme. The respective profiles of flux and multiplication factor convergence are shown in Fig. 5.8. In Table- 5.7, k_∞ values calculated by MOC code are compared with the values calculated by SURCU and TPTRI. Linear source

assumption does not bring significant reduction in computation time since burnable poison, which has potential to perturb the flux profile across the lattice, is present within a very small region as compared to the lattice size.

5.4.6 BWR benchmark problem with gadolinium pins

The BWR lattice is divided into 7,296 meshes to calculate k_∞ for flat source assumption. As shown in Table- 5.8, conventional iteration takes 114 sec with 15 inner per outer iteration scheme while Krylov iteration takes 48 sec with 4 BICG per outer iteration scheme. Hence, the computation time is reduced by a factor of 2.4. Fig. 5.9 shows the convergence of flux and multiplication factor values with iteration step and computation time in both the iteration techniques. k_∞ values calculated by MOC code differ only at third decimal place from the values calculated by DRAGON and MOCUM. For the present problem, linear source assumption is not attempted since flux variation across the lattice is insignificant.

Table- 5.7: k_∞ values obtained by MOC code using conventional and Krylov iteration scheme for LWR benchmark problem with burnable poison and their comparison with reference results

No. of mesh	Source assump-tion	Type of iteration	Iteration scheme	Compu-tation time (s)	No. of outer iteration	k_∞ calculated by MOC code	k_∞ calculated by other codes (Name of code)
8,160	Flat	Conventional	15 inner/outer	151	28	0.8789466	0.8805 (SURCU),
8,160	Flat	Krylov	5 BICG/outer	76	8	0.8789478	0.8828 (TPTRI)

Table- 5.8: k_∞ values obtained by MOC code using conventional and Krylov iteration scheme for BWR benchmark problem with gadolinium pins and their comparison with reference results

No. of mesh	Source assump-tion	Type of iteration	Iteration scheme	Compu-tation time (s)	No. of outer iteration	k_∞ calculated by MOC code	k_∞ calculated by other codes (Name of code)
7,296	Flat	Conventional	15 inner/outer	114	33	0.9886809	0.986561 (DRAGON),
7,296	Flat	Krylov	4 BICG/outer	48	10	0.9886823	0.987785 (MOCUM)

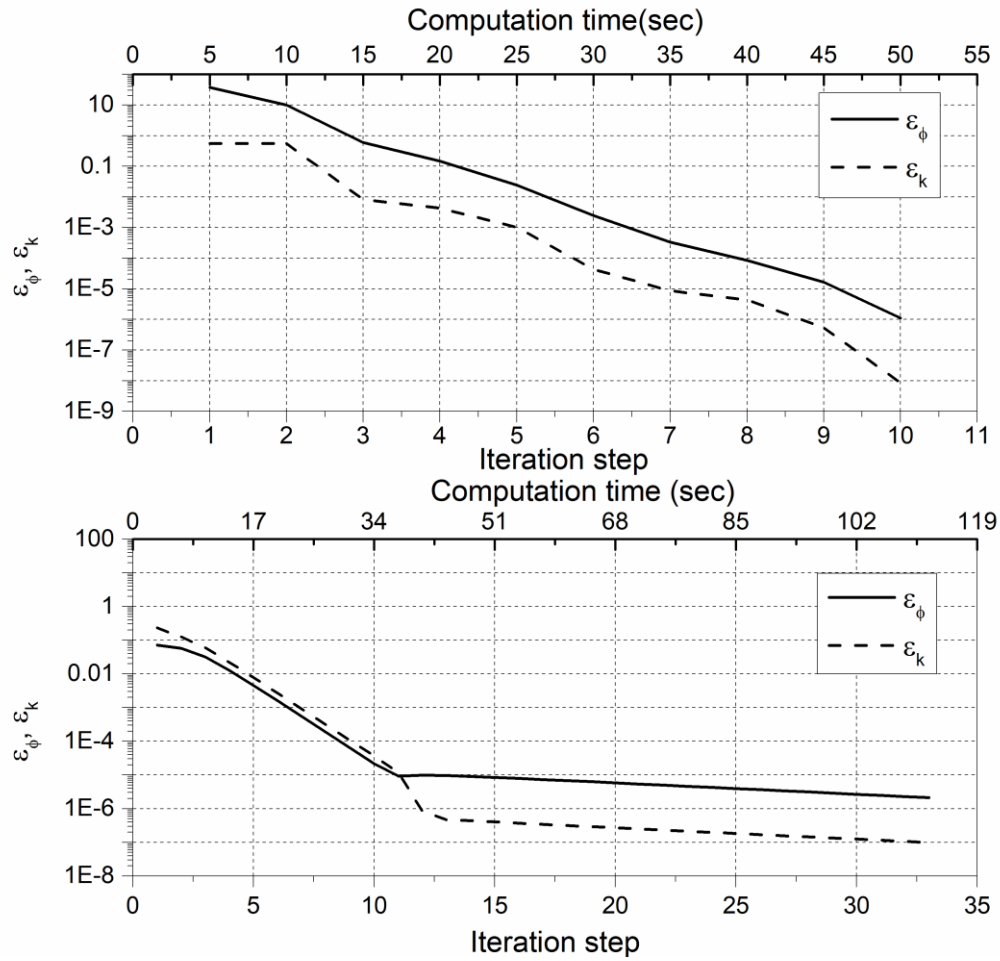

Fig. 5.9: Convergence of neutron flux (ε_ϕ) and multiplication factor (ε_k) of BWR benchmark problem (with gadolinium pins) with iteration step and computation time in Krylov iteration (at top) and conventional iteration (at bottom).

5.4.7 Hexagonal cell benchmark problem with central breeding pin

In order to bring out the advantage of Krylov iteration over the conventional iteration, k_∞ is calculated for the hexagonal cell using both the iteration techniques. Since, the source in a mesh is assumed to be flat, finer meshes are required for the calculation. Accordingly, the hexagonal cell is divided into 4,163 meshes. In order to obtain almost identical k_∞, conventional iteration takes 56 sec with 15 inner per outer iteration scheme while Krylov iteration takes only 29 sec with 7 BICG per outer iteration scheme. In Fig. 5.10, convergence of flux and multiplication factor values are plotted with iteration step and computation time for both the iteration techniques. In

Table- 5.9, k_∞ values calculated by MOC code are compared with the values calculated by MG-MCNP3B [75], TEPFEM and TPTRI. It is to be noted that the results from MOC code are closer to the Monte Carlo method based code MG-MCNP3B.

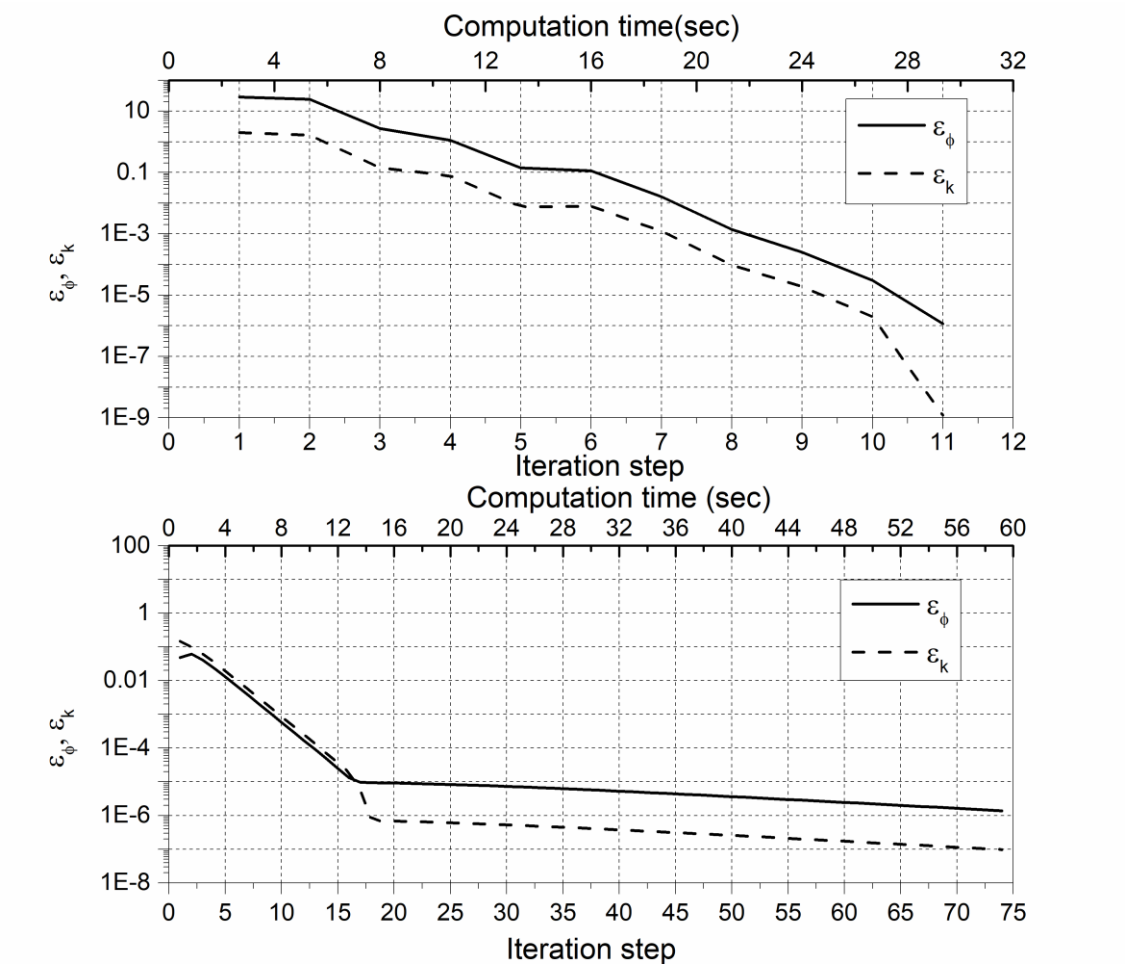

Fig. 5.10: Convergence of neutron flux (ε_ϕ) and multiplication factor (ε_k) of hexagonal cell benchmark problem with iteration step and computation time in Krylov iteration (at top) and conventional iteration (at bottom).

Table- 5.9: k_∞ values obtained by MOC code using conventional and Krylov iteration scheme for hexagonal cell benchmark problem with central breeding pin and their comparison with reference results

No. of mesh	Source assump -tion	Type of iteration	Iteration scheme	Compu -tation time (s)	No. of outer iteration	k_∞ calculated by MOC code	k_∞ calculated by other codes (Name of code)
4,163	Flat	Conventional	15 inner/outer	56	74	1.0901199	1.090803 (MG-MCNP3B), 1.086598 (TEPFEM), 1.085775 (TRTRI)
4,163	Flat	Krylov	7 BICG/outer	29	11	1.0901223	

5.4.8 CANDU-6 annular cell benchmark problem

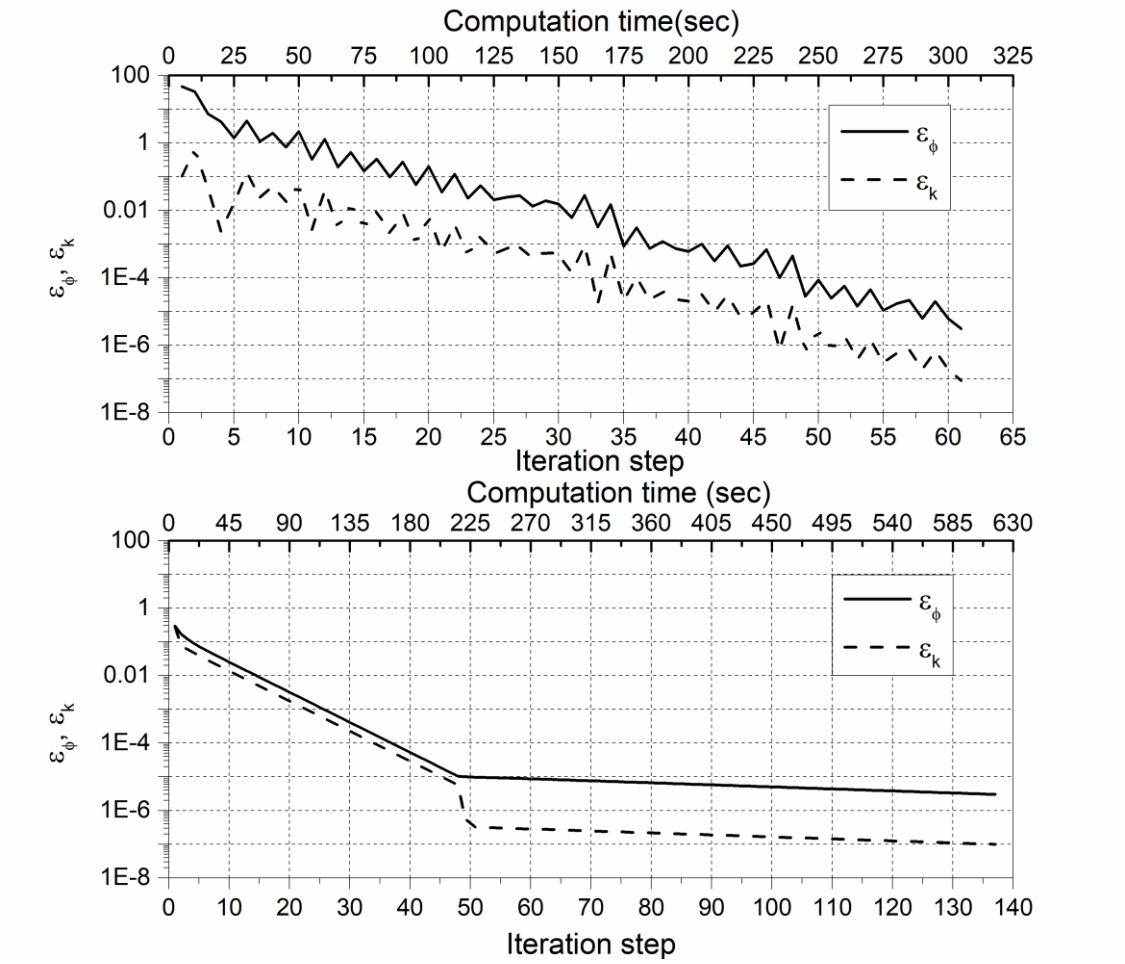

Fig. 5.11: Convergence of neutron flux (ε_ϕ) and multiplication factor (ε_k) of CANDU-6 annular cell benchmark problem with iteration step and computation time in Krylov iteration (at top) and conventional iteration (at bottom).

CANDU lattice is divided into 7,270 meshes to calculate k_∞ with flat source assumption. Conventional and Krylov iteration techniques are used for the calculation. Iteration scheme for each technique is optimized to obtain minimum computation time. Conventional iteration takes 603 sec with 15 inner per outer iteration scheme while Krylov iteration takes only 306 sec with 2 BICG per outer iteration scheme. In both the iteration techniques, the way flux and multiplication factor values reach the targeted convergence limit are plotted with iteration step and computation time in Fig. 5.11. In

Table- 5.10, k_∞ values calculated by MOC code are compared with the values calculated by DRAGON and MOCUM. It is clear from the comparison that for flat source assumption, Krylov iteration takes nearly half the time the conventional iteration takes.

5.4.9 HTTR benchmark problem

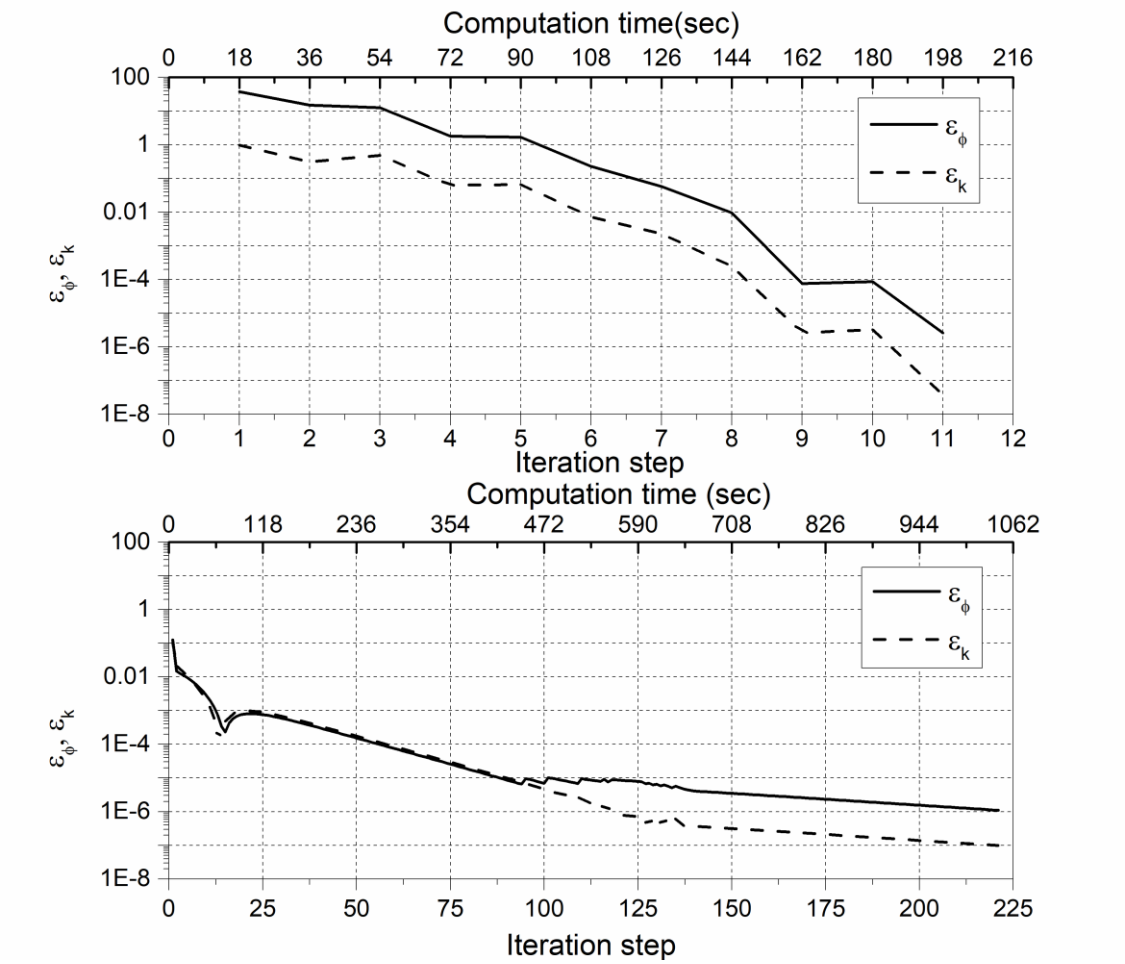

Fig. 5.12: Convergence of neutron flux (ε_ϕ) and multiplication factor (ε_k) of HTTR benchmark problem with iteration step and computation time in Krylov iteration (at top) and conventional iteration (at bottom).

For flat source assumption, the HTTR fuel block is divided into 5,078 meshes. Like all other problems, an optimization study is carried out to find out suitable iteration schemes for obtaining minimum computation time. Conventional iteration, with 5 inner per outer iteration scheme, takes 1059 sec to get converged k_∞ whereas Krylov iteration,

with 6 BICG per outer iteration scheme, takes only 199 sec to get the same. So, there is almost 1/5-th reduction of computation time, while maintaining the accuracy of solution, because of Krylov iteration. It is interesting to see in Fig.23 how the initial guess values of flux and multiplication factor progress towards the converged solution with iteration step as well as computation time. Table- 5.11 shows good agreement between k_∞ values calculated by MOC code, MCNP5 [62] and MOCUM. It is imperative to investigate whether Krylov iteration technique is accurate in predicting local parameter too like fission density, which is calculated for all the fuel pins of the fuel block. Fission density f_k of k-th fuel pin has already been defined in Eq.4.42. This fractional quantity is normalized to 33, which is the total number of fuel pins. Comparison of fission density distribution calculated by MOC code and MCNP5 in Table- 5.12 confirms that the accuracy of Krylov iteration is at par with conventional iteration.

Table- 5.10: k_∞ values obtained by MOC code using conventional and Krylov iteration scheme for CANDU-6 annular cell benchmark problem and their comparison with reference results

No. of mesh	Source assump-tion	Type of iteration	Iteration scheme	Compu-tation time (s)	No. of outer iteration	k_∞ calculated by MOC code	k_∞ calculated by other codes (Name of code)
7,270	Flat	Conventional	15 inner/outer	603	137	1.1113526	1.112837 (DRAGON),
7,270	Flat	Krylov	2 BICG/outer	306	61	1.1113608	1.111358 (MOCUM)

Table- 5.11: k_∞ values obtained by MOC code using conventional and Krylov iteration scheme for HTTR benchmark problem and their comparison with reference results

No. of mesh	Source assump-tion	Type of iteration	Iteration scheme	Compu-tation time (s)	No. of outer iteration	k_∞ calculated by MOC code	k_∞ calculated by other codes (Name of code)
5,078	Flat	Conventional	5 inner/outer	1059	221	1.0402054	1.04119 (MCNP5),
5,078	Flat	Krylov	6 BICG/outer	199	11	1.0402116	1.04084 (MOCUM)

Table- 5.12: Fission density distribution obtained by MOC code using conventional and Krylov iteration scheme for HTTR benchmark problem and their comparison with reference results

Pin	Reference value	Conventional iteration		Krylov iteration	
		Fission density	Relative % error with respect to ref. value	Fission density	Relative % error with respect to ref. value
1	0.99862	0.99865	0.00	0.99865	0.00
2	1.01562	1.01534	0.03	1.01534	0.03
3	1.03231	1.03188	0.04	1.03189	0.04
4	0.99872	0.99872	0.00	0.99871	0.00
5	0.98256	0.98291	0.04	0.98291	0.04
6	0.99015	0.99016	0.00	0.99017	0.00
7	0.99967	0.99982	0.02	0.99982	0.02
8	1.01565	1.01533	0.03	1.01533	0.03
9	1.01566	1.01537	0.03	1.01537	0.03
10	0.9902	0.99018	0.00	0.99018	0.00
11	0.98733	0.98736	0.00	0.98736	0.00
12	0.98944	0.98956	0.01	0.98955	0.01
13	0.99013	0.99017	0.00	0.99017	0.00
14	0.9986	0.99861	0.00	0.99861	0.00
15	1.03227	1.03187	0.04	1.03187	0.04
16	0.99969	0.99983	0.01	0.99983	0.01
17	0.98936	0.98956	0.02	0.98956	0.02
18	0.98725	0.98737	0.01	0.98737	0.01
19	0.98252	0.98292	0.04	0.98292	0.04
20	1.01562	1.01535	0.03	1.01535	0.03
21	0.99016	0.99018	0.00	0.99018	0.00
22	0.98723	0.98738	0.02	0.98738	0.02
23	0.98937	0.98959	0.02	0.98959	0.02
24	0.99008	0.99021	0.01	0.99021	0.01
25	0.99856	0.99868	0.01	0.99868	0.01
26	0.99859	0.99862	0.00	0.99862	0.00
27	0.98252	0.98292	0.04	0.98292	0.04
28	0.99021	0.99021	0.00	0.99021	0.00
29	0.99971	0.99987	0.02	0.99987	0.02
30	1.01561	1.01541	0.02	1.01541	0.02
31	0.99862	0.99865	0.00	0.99865	0.00
32	1.01564	1.01539	0.02	1.01539	0.02
33	1.03234	1.03192	0.04	1.03192	0.04

5.5 Results and discussion

In this chapter, application of Biconjugate Gradient stabilized (BiCGSTAB) method, which is one kind of Krylov subspace iteration method, is discussed in the context of acceleration of the MOC solution of neutron transport equation. Formulation of Krylov method requires the conventional MOC solution to be converted into a matrix equation. However, explicit formation of matrix by storing its non-zero elements, even in CSR format, requires huge computer memory, which hinders the MOC code to run in a personal computer. Therefore, we propose a numerical recipe, which takes the effect of the matrix on any vector without storing the matrix elements. This approach is first of its kind in the domain of Krylov accelerated MOC.

Performance of both Krylov iteration and conventional iteration is studied for several benchmark problems involving different geometry and boundary conditions. It has been found that number of inner or BICG iterations per outer iteration influences the computation time, as shown in Table- 5.1 for the MZA fast reactor benchmark problem. Hence, selection of suitable iteration scheme is an important aspect to be looked into, while minimizing the computation time.

All the results show that Krylov iteration takes lesser time than conventional iteration to obtain the solution of comparable accuracy. However, the extent up to which the computation time is reduced by Krylov iteration mainly depends on scattering of neutrons from one group to the other which again depends on number of energy groups involved in the problem. Neutron scattering from one group to the other group is predominant in 4 energy group problems of MZA, KNK-II and 6 energy group problem of HTTR, discussed in previous section. Therefore, the computation time, while solving these problems, is reduced by a factor of 3 to 5 or even more. Rest of the benchmark

problems involve 1 or 2 energy groups and the computation time is reduced only by a factor of 2 or less. However, solving a many energy group problem with simple geometry (e.g. a square filled with single homogeneous material) hardly reflects any advantage of Krylov iteration over the conventional iteration.

The concept of linear source expansion is useful in reducing the computation time for vacuum boundary problems where large flux gradient exists. With a view to gain more advantage than combining with flat source assumption, Krylov iteration is applied in the theoretical framework of linear source expansion. This shows further reduction in computation time by factor of 2 to 3.

CHAPTER 6

Conclusion and Future Scope of Work

Present thesis describes the Method of Characteristics (MOC) based formalism to solve neutron transport equation for mainly assembly level lattice calculation with reflective and periodic boundary conditions and to some extent core level calculation with vacuum boundary condition. The mesh generation procedure, used in this code, is based on Delaunay triangulation technique and Bower-Watson algorithm, which are considered to be quite general and hence, can be applied to triangulate any geometry. However, the code is currently capable of modeling any geometry consisting of a combination of circles, rectangles and hexagons subject to the outer boundary being square or hexagonal. In the first phase of development, flat source (flux) is assumed while solving the transport equation by MOC. With this assumption, performance of the code is tested by comparing the calculated multiplication factor with other international codes for selected benchmark problems. This, in general, shows good agreement with the reference values and indicates basic soundness of the method and the coding. Later an improved formulation of MOC is introduced aiming to obtain accurate results with fewer meshes as compared to earlier. This is achieved by representing the variation of the source in a mesh as a linear function of two spatial coordinates x and y. This helps in reducing the computation time. The analysis of a number of benchmark problems with this linear source representation clearly brings out its advantage over the flat source used earlier. However, the advantage is more manifest

in case of vacuum boundary condition or high neutron absorbing material being present in the system. Apart from this, the use of fewer triangular meshes cannot represent geometries with circular elements accurately enough. Hence, we work on the triangulation process so that the 'triangular' meshes are in conformity with the circular elements of the geometry. It is thus ensured that the advantage of using coarse meshes is not lost. Benchmark problems having cylindrical bodies (with circular projections) are studied to establish both these aspects i.e. the problem and its solution. Finally, an attempt is made to develop an efficient numerical recipe based on Krylov subspace iteration method, which requires matrix-vector multiplication. For real problems, size of the matrix is huge and storing its non-zero elements, even in most sophisticated and compact formats, requires extremely large computer memory. Therefore, the effect of matrix "A" on vector "ψ", instead of forming the matrix explicitly, is found out. We have attempted the matrix based approach first in order to understand how the matrix operates on the vector and then devised a numerical recipe, which, staying within the framework of conventional mesh-angle sweep of MOC, mimics the way the matrix is multiplied with the vector. This matrix-free approach is used in combination with the existing formalism of flat as well as linear source approximation to solve a number of benchmark problems. Results show significant improvement in terms of faster convergence of solution over the conventional iteration without compromising the accuracy.

In future, we have plan to couple the MOC code to a multi-group library and to include treatment of anisotropic scattering and burn up to convert it into a full-fledged lattice code. Further reduction in computation time is possible by simulating half or quarter core by permitting different boundary conditions at various surfaces instead of

the same boundary condition at each of the surfaces. Though efforts were made in this direction for one or two problems, there are scopes to implement this capability in a more general way. As a topic of future study, bringing the effect of preconditioning into the matrix free approach could be explored for getting computational incentive. We can also use direct methods like Orthomin etc. to replace the existing conventional iteration technique. It is possible to solve the neutron transport equation for full core calculation without doing any homogenization. For doing so, we would like to extend our transport code from lattice level to full core level using parallel computer, which is very much required to obtain the solution in reasonable time.

REFERNCES

[1] Huria, H. C., 1978. MURLI-Multigroup Integral Transport-Theory Code for thermal reactor lattice studies. Atomkernenergie, 31(2), 87-90.

[2] Huria, H. C., 1984. Incore Fuel Management Programs for Nuclear Power Reactors (p. 41). IAEA-TECDOC-314.

[3] Thilagam, L., Sunil Sunny, C., Jagannathan, V., Subbaiah, K. V., 2009. A VVER-1000 LEU and MOX assembly computational benchmark analysis using the lattice burnup code EXCEL. Annals of Nuclear Energy, 36(4), 505-519.

[4] Fujita, E. K., Driscoll, M. J., Lanning, D. D., 1978. Design and fuel management of PWR cores to optimize the once-through fuel cycle. (No. MIT-EL-78-017) MIT Energy Laboratory.

[5] Tsuchihashi, K., 1970. CLUP-77. A FORTRAN program of collision probabilities for square clustered assembly, JAERI-1196.

[6] Fayers, F. J., Davison, W., George, C. H., Halsall, M. J., 1972. LWR-WIMS: A modular computer code for the evaluation of light water reactor lattices. Part 1. Description of methods (No. AEEW-R--785 (Pt. 1)). Atomic Energy Establishment, Winfrith (England).

[7] Janssen, A. J., Caspers, C. M., 1973. THERMOGENE, an integral transport program for arbitrary X--Y geometry systems. Part 1. Basic theory of program and calculated results. Atomkernenergie. 21(l), 27-35.

[8] Janssen, A. J., Brand, P., Caspers, L.M., 1974. THERMOGENE, an integral transport program for arbitrary x--y geometry systems. III. Calculation of actual light water reactor lattices Atomkernenergie, 22(1), pp. 14-23.

[9] Degweker, S. B., 1985. BOXER-3: a three dimensional integral transport code for PHWR supercell. Report BARC-1295. Bhabha Atomic Research Centre, Bombay (India).

[10] Kosaka, S., Saji, E., 2000. Transport theory calculation for a heterogeneous multi-assembly problem by characteristics method with direct neutron path linking technique. Journal of nuclear science and technology, 37(12), pp.1015-1023.

[11] Lee, G. S., Cho, N. Z., Hong, S. G., 2000. Acceleration and parallelization of the method of characteristics for lattice and whole-core heterogeneous calculations. Proc. of PHYSOR 2000, Session II-C, Pittsburgh, USA, May 7-11.

[12] Joo, H. G., Cho, J. Y., Kim, K. S., Lee, C. C., Zee, S. Q., 2004. Methods and performance of a three-dimensional whole-core transport code DeCART. PHYSOR 2004 -The Physics of Fuel Cycles and Advanced Nuclear Systems: Global Developments, Chicago, Illinois, April 25-29, American Nuclear Society, Lagrange Park, IL.

[13] Jung, Y. S., Joo, H. G., 2009. Decoupled planar MOC solution for dynamic group constant generation in direct three-dimensional core calculations. In: Proc. of Int. Conf. on Mathematics and Computational Methods Applied to Nuclear Science and Engineering (M&C2009), May 3–7, 2009, Saratoga Springs, NY, USA.

[14] Kosaka, S., Takeda, T., 2004. Verification of 3D heterogeneous core transport calculation utilizing non-linear iteration technique. Journal of nuclear science and technology, 41(6), pp.645-654.

[15] Cho, N. Z., Lee, G. S., Park, C.J., 2002. A fusion technique of 2-D/1-D methods for three-dimensional whole-core transport calculations. Proc. of the Korean Nuclear Society Spring Meeting, Kwangju.

[16] Cho, N. Z., Lee, G. S., Park, C.J., 2002. Fusion of Method of Characteristics and Nodal Method for 3-D Whole-Core Transport Calculation, Trans. Am. Nucl. Soc., 86, pp.322.

[17] Askew, J. R., 1972. A characteristics formulation of the neutron transport equation in complicated geometries. Report AEEW-M, 1108.

[18] Hong, S. G., Cho, N. Z., 1998. CRX: a code for rectangular and hexagonal lattices based on the method of characteristics. Ann. Nucl. Energy 25 (8), 547–565.

[19] Smith, K., Rhodes, J., 2000. CASMO-4 characteristics methods for two-dimensional PWR and BWR core calculations, ANS/ENS International Winter Meeting and Embedded Topical Meetings, Washington D.C., November.

[20] Chen, Q., Wu, H., Cao, L., 2008. Auto MOC – a 2D neutron transport code for arbitrary geometry based on the method of characteristics and customization of AutoCAD. Nucl. Eng. Des. 238 (10), 2828–2833.

[21] Marleau, G., Hébert, A., Roy, R., 2008. A user guide for DRAGON 3.06. Report IGE-174 Rev, 7.

[22] Yang, X., Satvat, N., 2012. MOCUM: A two-dimensional method of characteristics code based on constructive solid geometry and unstructured meshing for general geometries. Annals of Nuclear Energy, 46, 20-28.

[23] Boyd, W., Shaner, S., Li, L., Forget, B., Smith, K., 2014. The OpenMOC method of characteristics neutral particle transport code. Annals of Nuclear Energy, 68, pp.43-52.

[24] Ray, S., Degweker, S. B., Rai, R., Singh, K.P., 2016. A collision probability and MOC based lattice and burnup code for analysis of LWR fuel assemblies. Nucl. Sci. Eng. 184 (4).

[25] Stepanek, J., Auerbach, T., Haelg, W., 1983. Calculation of four thermal reactor benchmark problems in X–Y geometry. EPRI NP-2855.

[26] Häggblom, H., Ahlin, Å., Nakamura, T., 1975. Transmission probability method of integral neutron transport calculation for two-dimensional rectangular cells. Nuclear Science and engineering, 56(4), pp.411-422.

[27] Mader, C., 1975. A transport program in X–Y geometry based on function expansion in angular and space. EIR-Bericht Nr 290.

[28] Wasastjerna, F., 1979. An Application of the Transmission Probability Method to the Calculation of Neutron Flux Distributions in Hexagonal Geometry. Nuclear Science and Engineering, 72(1), pp.9-18.

[29] Zhang, Y., 2000. Numerical Solution of Two-dimensional Neutron Integral Transport Equation for Hexagonal Geometry and Development of Software Package for Calculation of PWR Hexagonal Assembly [Dissertation for doctoral degree in Chinese]. Xi'an Jiaotong Univerity, Xi'an.

[30] Hongchun, W., Pingping, L., Yongqiang, Z., Liangzhi, C., 2007. Transmission probability method based on triangle meshes for solving unstructured geometry neutron transport problem. Nuclear engineering and design, 237(1), 28-37.

[31] Sugimura, N., Yamamoto, A., Ushio, T., Mori, M., Tabuchi, M., Endo, T., 2007. Neutron transport models of AEGIS: an advanced next-generation neutronics design system. Nuclear science and engineering, 155(2), pp.276-289.

[32] Jevremovic, T., Vujic, J. and Tsuda, K., 2001. ANEMONA—a neutron transport code for general geometry reactor assemblies based on the method of characteristics and R-function solid modeler. Annals of Nuclear Energy, 28(2), pp.125-152.

[33] Halsall, M. J., 1980. CACTUS, a characteristics solution to the neutron transport equations in complicated geometries (No. AEEW-R--1291). UKAEA Atomic Energy Establishment.

[34] Goldberg, L., Vujic, J., Leonard, A., Stachowski, R., 1995. Trans. Am. Nucl. Soc., 73, 173-174.

[35] Cho, J. Y., Joo, H. G., Kim, K. S., 2002. Cell based CMFD formulation for acceleration of whole-core method of characteristics calculations. Journal of the Korean Nuclear Society, 34(3), pp.250-258.

[36] Mazumdar, T., Degweker, S. B., 2015. Solution of neutron transport equation by Method of Characteristics. Annals of Nuclear Energy, 77, pp.522-535.

[37] Masiello, E., Clemente, R., Santandrea, S., 2009, May. High Order Method of Characteristics for 2-D Unstructured Meshes. In Proc. Int. Conf. on Mathematics, Computational Methods & Reactor Physics (M&C 2009) (pp. 3-7).

[38] Ferrer, R. M., Rhodes III, J. D., 2016. A Linear Source Approximation Scheme for the Method of Characteristics. Nuclear Science and Engineering, 182(2), pp.151-165.

[39] Mazumdar, T., Degweker, S. B., 2017. Solution of the neutron transport equation by the Method of Characteristics using a linear representation of the source within a mesh. Annals of Nuclear Energy, 108, pp.132-150.

[40] Joo, H. G., Cho, J. Y., 2002. Dynamic implementation of the equivalence theory in the heterogeneous whole core transport calculation. In: Proceedings of the PHYSOR 2002, Paper 13A-02 on CD-ROM, October 7–10, Seoul, Korea.

[41] Cho, J. Y., Kim, K. S., Shim, H. J., Song, J. S., Lee, C. C., Joo, H. G., 2008. Whole core transport calculation employing hexagonal modular ray tracing and CMFD formulation. Journal of nuclear science and technology, 45(8), pp.740-751.

[42] Tang, C., Zhang, S., 2009. Development and verification of an MOC code employing assembly modular ray tracing and efficient acceleration techniques. Ann. Nucl. Energy 36, 1013–1020.

[43] Yamamoto, A., 2005. Generalized coarse-mesh rebalance method for acceleration of neutron transport calculations. Nucl. Sci. Eng. 151, 274–282.

[44] Chai, X.M., Yao, D., Wang, K., Wang, L. and Yu, Y., 2010. Generalized coarse-mesh finite difference acceleration for method of characteristics in unstructured meshes. Chinese Joural of Computational Physics, 27(4), pp.541-547.

[45] Halsall, M.J., 1992. The CACTUS transport method in WIMS (No. IAEA-TECDOC--678).

[46] Wu, G. J., Roy, R., 2003. Acceleration techniques for trajectory-based deterministic 3D transport solvers. Annals of Nuclear Energy, 30(5), pp.567-583.

[47] Zika, M. R., Adams, M. L., 2000. Transport synthetic acceleration for long-characteristics assembly-level problems. Nucl. Sci. Eng. 134, 135–158.

[48] Sanchez, R., Chetaine, A., 2000. A synthetic acceleration for a two-dimensional characteristic method in unstructured meshes. Nucl. Sci. Eng. 136, 122–139.

[49] Santandrea, S., Sanchez, R., 2002. Acceleration techniques for the characteristic method in unstructured meshes. Annals of Nuclear Energy, 29(4), pp.323-352.

[50] Hong, S. G., Cho, N. Z., 1999. Angular depend rebalancing (ADR) iteration for discrete-ordinate transport problems in equilateral triangle meshes. In: Proc. of the Int. Conf. on Mathematics and Computation, Reactor Physics and Environmental Analysis in Nuclear Applications, Madrid.

[51] Patton, B. W., Holloway, J. P., 2002. Application of preconditioned GMRES to the numerical solution of the neutron transport equation. Annals of Nuclear Energy, 29(2), pp.109-136.

[52] Gupta, A., Modak, R. S., 2004. Krylov sub-space methods for K-eigenvalue problem in 3-D neutron transport. Annals of Nuclear Energy, 31(18), pp.2113-2125.

[53] Saad, Y., 1996. Iterative Methods for Sparse Linear System. PWS, Boston, MA.

[54] Dahmani, M., Le Tellier, R., Roy, R., Hébert, A., 2005. An efficient preconditioning technique using Krylov subspace methods for 3D characteristics solvers. Annals of Nuclear Energy, 32(8), pp.876-896.

[55] Zhang, H., Wu, H., Cao, L., 2011. An acceleration technique for 2D MOC based on Krylov subspace and domain decomposition methods. Annals of Nuclear Energy, 38(12), pp.2742-2751.

[56] Zheng, Y., Peng, M., Xia, G., 2017. Development and verification of MOC code based on Krylov subspace and efficient preconditioning techniques. Annals of Nuclear Energy, 99, pp.427-433.

[57] Mazumdar, T., Gupta, A., 2018. Application of Krylov Acceleration Technique in Method of Characteristics based Neutron Transport Code, Submitted to Nuclear Science and Engineering.

[58] Bell, G.I. and Glasstone, S., 1970. Nuclear reactor theory (No. TID-25606). US Atomic Energy Commission, Washington, DC (United States).

[59] Lewis, E.E. and Miller, W.F., 1984. Computational methods of neutron transport.

[60] Sood, A., Forster, R. A., Kent Parsons, D., 2003. Analytical benchmark test set for criticality code verification. Prog. Nucl. Energy 42 (1), 55–106.

[61] Wood, J., Williams, M. M. R., 1984. Recent progress in the application of the finite element method to the neutron transport equation. Ann. Nucl. Energy 11 (1), 21–40.

[62] Zhang, Z., Rahnema, F., Zhang, D., Pounders, J. M., Ougouag, A.M., 2011. Simplified two and three dimensional HTTR benchmark problems. Ann. Nucl. Energy 38 (5), 1172–1185.

[63] Brantley, P. S., Larsen, E. W., 2000. The simplified P3 approximation. Nucl. Sci. Eng. 134, 1–20.

[64] Kaise, Y., Osada, H., 2003. Investigation of MOZART experimental data and analysis of MOZART experiment using JFS-3-J3. 2R group constant (No. JNC-TJ–9400-2003-009). Japan Nuclear Cycle Development Inst.

[65] Chiba, G., Numata, K., 2007. Neutron transport benchmark problem proposal for fast critical assembly without homogenizations. Ann. Nucl. Energy 34 (6), 443–448.

[66] Takeda, T., Ikeda, H., 1991. 3-D neutron transport benchmarks. J. Nucl. Sci. Technol. 28 (7), 656–669.

[67] Kim, T.H., Cho, N.Z., 1996. Extension of source projection analytic nodal Sn method for analysis of hexagonal assembly cores. J. Korean Nucl. Soc. 28 (5), 488–499.

[68] Walters W.F., Brinkley F.W., Marr D.R., 1989, Users Guide for TWOHEX: A Code Package for Two-Dimensional, Neutral-Particle, Transport in Equilateral Triangular Meshes, LA-10258-M, Los Alamos National Laboratory.

[69] Del Valle, E., Mund, E. H., 2004. RTk/S N solutions of the two-dimensional multigroup transport equations in hexagonal geometry. Nucl. Sci. Eng. 148 (1), 172–185.

[70] Petkov, P.T., Takeda, T., 1998. Transport calculations of MOX and UO2 pin cells by the method of characteristics. J. Nucl. Sci. Technol. 35 (12), 874–885.

[71] Menon, S.V.G., Khandekar, D.C., 1984. Convergence of the inner–outer iteration scheme. Zeitschrift für angewandte Mathematik und Physik (ZAMP) 35 (3), 321–331.

[72] Rebay, S., 1993. Efficient unstructured mesh generation by means of Delaunay triangulation and Bowyer–Watson algorithm. J. Comput. Phys. 106 (1), 125–138.

[73] Michels, H., 1997. DISLIN manual. Max-Planck-Institut fur Aeronomie, Katlenburg-Lindau, 288. URL: http://www.dislin.de/ (last accessed on 24th December, 2019).

[74] Cao, L.Z., 2004. Spherical harmonics method for neutron transport equation based on unstructured-meshes. Nucl. Sci. Technol. 15 (6), 335–339.

[75] Deng, L., Xie, Z.S., 2000. A 3-d multigroup P3 MC code and its benchmarks. J. Nucl. Sci. Technol. 37 (7).

[76] Nagaya, Y., Okumura, K., Mori, T., Nakagawa, M., 2005. MVP/GMVP 2: general purpose Monte Carlo codes for neutron and photon transport calculations based on continuous energy and multigroup methods.

[77] Van der Vorst, H.A., 1992. Bi-CGSTAB: A fast and smoothly converging variant of BiCG for the solution of nonsymmetric linear systems. SIAM Journal on scientific and Statistical Computing, 13(2), pp.631-644.

[78] Gupta, A. and Modak, R.S., 2002. On the use of the conjugate gradient method for the solution of the neutron transport equation. Annals of Nuclear Energy, 29(16), pp.1933-1951.