DEVELOPMENT OF INDIGENOUS MONTE CARLO CODE FOR ESTIMATION OF k_{eff} OF NUCLEAR CYCLE FACILITIES

BY
K.V.SUBBAIAH

MANIPAL CENTRE FOR NATURAL SCIENCES CENTRE OF EXCELLENCE, MANIPAL ACADEMY OF HIGHER EDUCATION MADHAV NAGAR, MANIPAL- 576104, KARNATAKA

PROGRESS REPORT FINAL (2016-2020)

DAE BRNS Project No: 36(4)/14/40/2015

Contents

1. Project Overview

1. Name and Address of the institute: **Manipal Centre for Natural Sciences Centre of Excellence, Manipal Academy of Higher Education, Manipal 576104, Karnataka**

2. Title of the Project : **Development of Indigenous Monte Carlo Code for Estimation of k_{eff} of Nuclear Cycle Facilities**

Reference No. : **DAE BRNS/Project No. 36(4)/14/40/2015**

Date of first sanction : **21.12.2015**

3. Name, Designation and Full Address including E-mail address, Phone No and Fax No. of the Principal investigator of the project :

Dr. Kamatam Venkata Subbaiah
Consultant Professor and RSO
Manipal Centre for Natural Sciences Centre of Excellence
Dr. T.M.A. Pai Planetarium Building
Manipal Academy of Higher Education
Manipal 576104, Karnataka
e-mail : kv.subbaiah@manipal.edu
phone: +91 8202923575

4. Date of Commencement of actual work on the project: **April 1, 2016**

5. Details of the grant (name and designation of staff, name and cost of equipment, consumables (give heads) received during the tenure of the scheme:

Name	Designation	Date of joining	Date of resignation
Mr. Sachin Seth	**JRF**	**01-12-2016**	**31-03-2020**

Details of Grant

Budget Head	Sanctioned			
	YEAR I (2015-2016)	YEAR II (2016-2017)	YEAR III (2017-2018)	YEAR IV (2018-2019)
Equipment	10,50,000/-	0	0	0
Staff Salary -RA(1)	4,32,000/-	4,32,000/-	4,32,000/-	
Consumables	30,000/-	20,000/-	30,000/-	
Travel - PI	60,000/-	60,000/-	60,000/-	
Contingencies	50,000/-	50,000/-	50,000/-	
Overheads	1,17,900/-	38,400/-	38,400/-	
Total (INR)	17,39,900/-	6,00,400/-	6,00,400/-	

Equipment procured for the project

1) Computer Node Node of HP (16 cores with Hyper threading) for Parallel Computer existing at MCNS, MAHE, Manipal
2) HP Laptops- 2 Nos.
3) Multi-function computer peripheral-Lajer printer and scanner

6. Detailed Technical Report on the entire work done under the project

1. Scientific background of the project

With the closure of many experimental facilities, the nuclear criticality safety analyst increasingly is required to rely on computer calculations to identify safe limits for the handling and storage of fissile materials. However, in many cases, the analyst has little experience with the

specific codes available at his/her facility. Uranium and plutonium are the elements that undergo neutron induced fission and are commonly used as nuclear fuels for power production. Uranium is naturally occurring element in the earth crust whereas the plutonium is manmade element and is formed continuously as a by-product during power generation in uranium fuelled reactors by transmutation process. These elements are processed in nuclear fuel cycle facilities to obtain the required purity of isotope, desired shape and size before deploying as fuel bundle in reactors. There are two distinct differences between reactors and nuclear fuel cycle facilities that must be taken care of in the estimation of keff of a system. In reactors, the fuel is always in the form of solid bundles whereas in nuclear fuel cycle facilities the fuels can exist in the form of solids (Fuel fabrication units, storage bays), in the form of powders (Bird Cages) or in the form of solutions (Reprocessing labs, dissolver, storage tanks) and the second difference is the keff, which is near to unity in the operating reactors while in nuclear cycle facilities it is always less than unity. Further, the reactor contains repeated structures of fuel sub-assemblies whereas nuclear fuel cycle facilities contain bodies of different shapes and sizes and arranged in arbitrary manner. In the development of new Monte Carlo code for estimating keff for nuclear fuel cycle facilities, these facts have to be kept in mind and one should adopt suitable algorithms to estimate precise value of keff of a system.

U-235 and Pu-239 isotopes of uranium and plutonium are called fissile materials. During fission process, these isotopes release two or more neutrons and a large amount of energy. These neutrons while moving in the system may interact with fissile atoms causing further fissions, which may lead to multiplication of neutrons, resulting in uncontrolled chain reaction. Estimation of neutron multiplication factor a fissile system is of paramount importance in the nuclear industry. The neutron multiplication factor of a fissile system is denoted as keff, which is defined as the ratio of fission neutrons produced in a current generation to the number of fission neutrons started in the previous generation. Here, the generation refers to tracking of all neutrons in a cycle from its birth (fission source points) till its death (absorption or escape from the system). Based on the magnitude of keff, the fissile systems are called as subcritical system (keff<1), critical system (keff =1), and super critical system (keff>1). Nuclear reactors, fuel enrichment plants, spent fuel reprocessing plants, fuel storage bays, fuel fabrication plants are some of the areas in a nuclear industry where a large amount of fissile materials are used/handled. Improper storage or inadvertent movement of these materials may lead to neutron multiplication factor greater than one resulting in adverse consequences. Thus, the keff plays a crucial role in the control and safety of fissile systems.

The keff of a fissile system can be estimated by solving the Radiation Transport Equation(RTE), which is nothing but the neutron balance equation in phase-space variables involving neutron flux. Since the analytical solution is difficult, the solution of the equation can be obtained numerically either by deterministic or Monte Carlo simulation method. Owing to ease in handling variety of neutron interactions and ease in representation of arbitrary configuration of materials of any fissile system, Mont Carlo methods are preferred for precise estimation of keff. However, the main disadvantage with this method is the requirement of a huge amount of computer time for problems involving a large number of simulations. This difficulty can be circumvented by

1. the availability faster Computers (Multiple CPUs),
2. using modern Softwares/ operating systems (Parallel Programming Compilers)
3. Employing latest and efficient programming techniques.

The existing codes are being modified to account for the latest changes in the neutron interaction data and modern programming techniques and on the other hand, new Monte Carlo codes are being developed not only to take care of latest neutron data but also the rapid developments happening in the computer hardware, software and programming techniques(object oriented programming). These newer versions of the codes denied to India and indigenous codes are not available currently.

However, there are projections in India to increase the nuclear power production by importing advanced reactors as well as developing FBRs in India for power generation and intensifying the research on utilization of thorium. While most of present day reactors are U fuelled, the development of advanced U-Th or Pu-Th fuelled reactors is important to our future technology growth. Criticality estimations of these systems are expected to be more difficult and need to be addressed. This will be explored while developing this code. Hence, there is a need to develop reliable, faster and efficient Monte Carlo codes in India to handle variety of fissile configurations.

The current proposed project will be a sincere attempt to develop an indigenous MC code for estimation of keff a fissile system incorporating the latest available international nuclear data base and employing the state of the art programming and computational techniques. The code will be tailor made to address the geometrical situations that are encountered in nuclear fuel cycle facilities (coupled/decoupled systems containing fuels in solid/liquid phages). The outcome of the project will be the development of a new criticality Monte Carlo Code capable of simulating 3D neutron transport in complex configuration of materials but also fast due to deployment of advanced computing techniques. The developed code will be tested against benchmark problem results and also against the results of other Monte Carlo codes. Few thorium fuel benchmarks exist and the subject has not been extensively studied. We would like to consider undertaking some theoretical and experimental benchmarks for the thorium fuel in this study. Further, the source code will be available in house for making changes to incorporate latest developments in input data base as well as changes happening in the format of data bases. Besides, it can be easily extended to simulate transport of other particles and other tallies of practical relevance. Because of the attractive features available in the new code, it will be of immense use in various DAE organizations. Besides, this project will enable universities to develop expertise in the field of nuclear energy in places other than the DAE facilities.

PARTII-PROJECTOBJECTIVES,RESEARCHPLAN and DELIVERABLES

200. List of Objectives proposed in the project

1. Setting up of latest evaluated neutron interaction cross section data for commonly encountered isotopes in the specific format(ACE) suitable for general purpose Monte Carlo Codes
2. Assessment of algorithms for generation of pseudo random number generators with suitable properties, in particular with a stride which is essential for parallel computing.
3. Development of efficient neutron tracking method in complex geometry specified in the extended form of Combinatorial geometry package(Addition of Arrays and HEXAGON Body)
4. Designing input specification in most advanced and verbose manner (XML-extended mark up language, input) and inclusion of other desired tallies such as reaction rates, neutron flux in addition to k_{eff} in reporting of results in the output file. Further, the results will be written in convenient format for subsequent graphical visualization.

5. Programming the code with the state of the art prevailing techniques with geometry trace and Debug facility. Provision to plot interactively in 2D geometry view, inspection of neutron cross sections, both micro and macro, for the isotopes specified in the input file.

6. Validation of the results of the new code with the recent reported data as well as against the results of standard international Monte Carlo codes.

7. Detailed documentation through reports, operating manuals and publications

210. **Describe the yearly ResearchPlan and identify deliverables***(Please see Instruction - Sr. No. 14):*

A. At PI/ CI's Institution

Ist Year:

1. Literature Survey, collection of latest information on evaluated neutron interaction cross sections data and on object oriented programming techniques with data encapsulation ability.
2. Assessment of new developments in solving the Radiation Transport Equation by Monte Carlo Methods in particular sampling theory/methods of scattered/secondary neutron energy angular distribution and also correlated distributions.
3. Preparation of technical design reports, flow charts, data formats and others related to the above tasks.

IInd Year:

1. Creation of proper neutron interaction data libraries in the required format for easy and fast access.
2. Combinatorial geometry package with extension of array specification and hexagonal body. Development and testing of this geometry tracking object module.
3. Development and programming modules of essential components of Monte Carlo Code for criticality.
4. Development of Graphical user interfaces for I/O planned for this code.
5. Integration of all required objects (modules) and building up of single executable file.

IIIrd Year

1. Test of Executable module for sample problems.
2. Testing of code against Bench marked criticality problems. Corrections and modifications if required.
3. Preparation of complete documentation on the developed code.

B. At PC's Institution.

Ist Year:

1. Formal exposure to the neutron data bases and training of the researcher/JRF of the project from the PI institution on the details of the neutron interactions and the format of existing data bases.
2. Development of program for extraction of particular neutron interaction data from the master data base file.
3. Survey and selection suitable algorithm of pseudo random number generators with a stride. Programming, testing and assessing suitability for parallel computing.

IInd Year:

1. Specification and arriving at easy and attractive I/O operations for the new code.Designing of the front-end screens of the user interfaces and programming of XML reader and parser.
2. List directed display of the contents in the input typing and context sensitive help features.
3. Testing and verification of I/O procedures and corrections, if any and final flowcharts preparation on the input.

IIIrd Year:

1. Merging and testing of the code modules developed at PI and PC institutions.
2. Incorporating required changes in the code for parallel computing.
3. Testing of the new code with simple and other benchmark problems.
4. Feedback/suggestions to evolving the manual.

210. C Deliverables

1. Indigenously developed Monte Carlo code executable both on standalone and parallel computers along with source code will be delivered.
2. CD containing latest evaluated neutron interaction cross section data library for commonly encountered isotopes in the specific format(ACE) suitable for newly developed Monte Carlo Code will be deposited with BRNS.
3. Full Documentation of the program, User Manual, Sample Problems and indications for future enhancements/developments will also contain in the above CD.
4. Documentation will include algorithms, mathematical formulations, sampling theories and other new developments in the code. Designing input specification in most advanced and verbose manner (XML-extended markup language, input) and inclusion of other desired tallies such as reaction rates, neutron flux in addition to k_{eff} in reporting of results in the output file. Programming the code with the state of the art prevailing techniques with geometry trace and Debug facility. Provision to plot interactively in 2D geometry view & reaction CXS. Validation of the results of the new code with the recent reported data as well as against the results of standard international Monte Carlo codes.
5. Technical reports and Publications.

2. Installation and Input Guide

- **COPY DVD contents (all folders and Sub folders) to a Hard Drive**

MCkeff.EXE file is provided and works for WINDOWS OS 7 or higher. If it does not work, you may build exe file from source code.

2.1. Compiling MCkeff from source

The Fortran source code and the makefile are present in the directory "source_code". To build the executable various compiler "flags" and compiler "defines" are used. The used compiler flags are,

```
-fcheck=all       <standard compiler checks>
-fcheck=bounds <Checks array bounds while executing>
-O3               <Optimiz exe for speed>
-w                <inhibit warning messages>
-fopenmp          <Enable OpenMP compile directives>
-fbackslash       <To erase the string in command window>
```

Used compiler defines are, WINDOWS, GFORTRAN and LINUX. These are written in both windows and linux makefiles.

2.1.1. For Windows Operating Systems follow the steps below

1. Go to Hard Drive (> C/D/…:)
2. CD/compilers
3. Dbl Click on "Compilerintaller"

The computer copies required compiler folders/subfolders and modifies the system path to access GFORTRAN compiler.

To create executable of MCkeff.exe

- CD/source_code
- Identify makefile.windows and give the following command.

```
make -f makefile.windows
```

After successful execution, MCkeff.exe file is created in the same folder.
This procedure tested under windows 10 with 64bit Operating System. It is recommended to use operating system windows 7 or later.

2.1.2 For Linux operating systems

The compilation procedure has been carried out under UBUNTU 19.04 LTS Linux operating system. For the compilation purpose it is recommended to use operating system from UBUNTU 17 onwards. The "gfortran compiler" and "make" can be installed with the following command.

sudo apt install gfortran make

2. Installation and Input Guide

Open a UBUNTU terminal by prssing ALT+CTRL+T. And change the directory to MCK/source_code. To create Linux executables run the command

make -f makefile.linux

in the current directory, which will create Linux mckeff executables. Additionally, you can set system path to this folder such that the mckeff can be used throughout the system. Ubuntu 19.04 LTS is used for testing these Linux executables.

2.2. Cross-section configurations

The required cross section files are given in the folder cross_sections. Extract the zip files to a folder (endf_cxs).
Or
ENDF 7.1 ACE files can be downloaded from the site.
https://www.nndc.bnl.gov/endf/b7.1/aceFiles/ENDF-B-VII.1-tsl.tar.gz
Extract to the folder (endf_cxs) , and transfer XSDIR file to the MCkeff.exe directory.

Next, edit first line of ..**xsdir** file as

datapath=path_to_cross_sections\endf_cxs

MCkeff uses this information to locate cross section data for the nuclides.

2.3. Running the MCkeff.exe

Running the code and various command line options are identical both in WINDOWS and LINUX OS. To get options available type

>MCKEFF –h

The following are displayed.

Command line option	Description
i=<InputFile>	To specify user input file name (default ‘inp’)
o=<OutputFile>	To specify user Output file name. (default ‘out*’)
-p	To get Geometry plot of the input problem.
-c <#n>	To run with user specific number of threads (#n = No of threads).
-h	To print help message.

To run with input file as ‘inputfile’ and output file as ‘outputfile’, the following command need to be given.

>mckeff i=inputfile o=outputfile

Note: In the near future, MCkeff will be made to run on parallel computers. In that case, the command line will be

>mpirun –n xx –machinefile nodefile MCkeff [options]

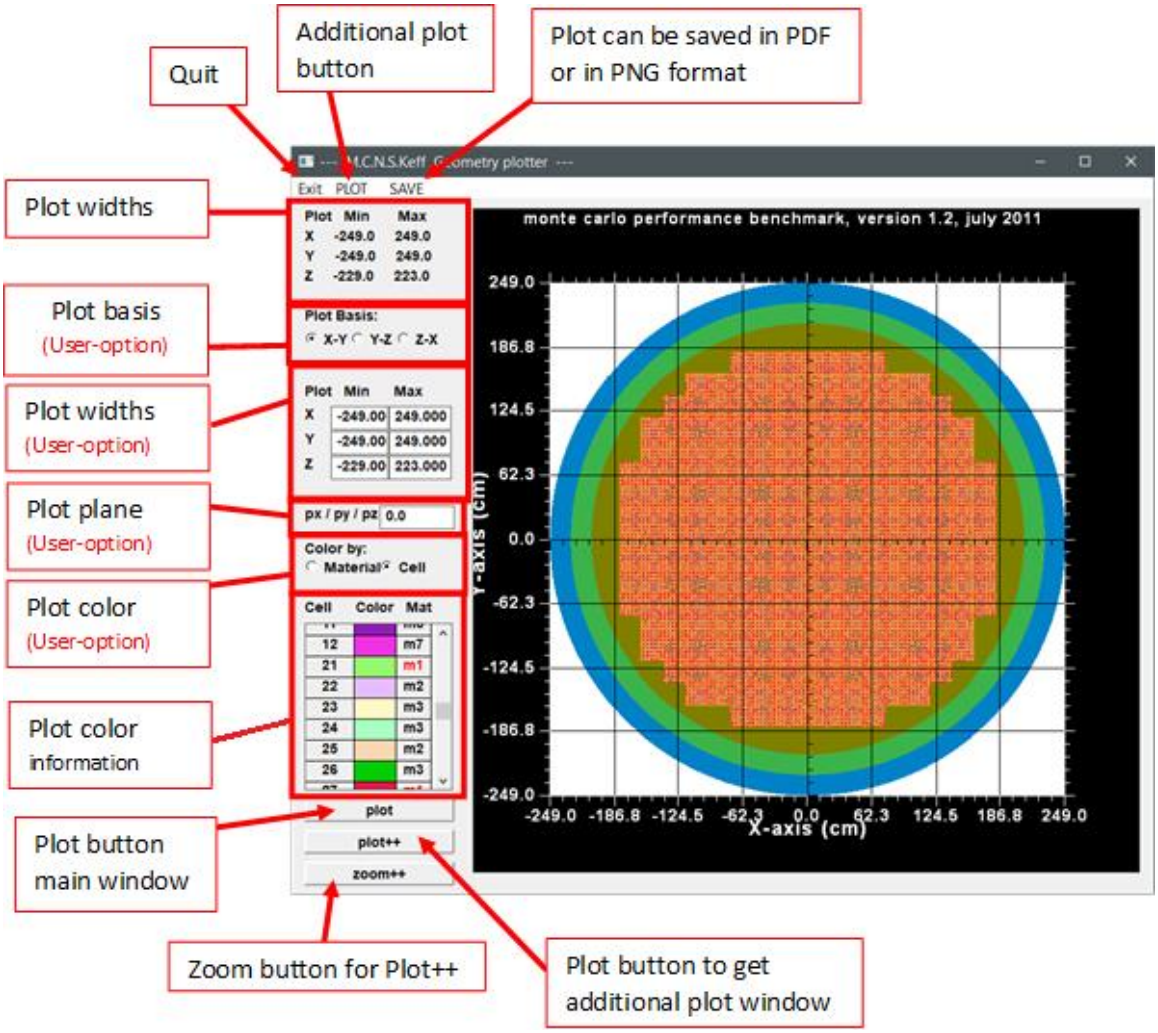
Where xx stands for number of nodes in cluster (eg. 1/2/3…)

2.4. The geometry plotter

To plot the geometry of the input problem (in windows OS), pass the argument "**-p**" in the command line as

> **mckeff -p** *Or* > **mckeff i=inputfile o=outx -p**

The geometry plotter available for windows operating systems and can be invoked by passing the argument **"-p"** during the execution. One can use command "–po" in place of "-p" in case the option '-p' does not produce plots. The plotter has various user-options like changing plot widths, plot basis plane and plotting colors. By changing the option, the geometry can be re-re-plotted on clicking plot button. This is the integrated plot window; this window will not keep aspect ratio accordingly with the plot widths. Additional plot window can be obtained by plot++ button, this window keeps aspect ratio according to the plot widths, and this plot in the plot++ window can be saved in PDF and PNG format using the buttons at menu bar. Figure 2 shows zoomed portion of the plot which is given in Fig.2.1.



NOTE: Few additional details on geometry plotter are given in User Manual. More plots are given in Chapter-6 of the document file.

Fig. 2.1. Geometry Plotter Main

2.5. Input file Format

Input file serve as the medium of communication between the user and the program. Input is the intelligible information passed on to a computer code to perform specified task. The information can be passed on either through interactive mode (Terminal) or through a text file (File saved on a storage device). In general, many of the computer codes may require a large amount of input information and hence it is preferred to provide the required data in the form of text files. In addition, these codes may require additional standard data (cross section data), which often may be voluminous and are specified as a stored text files or binary files. After completion of the task, the results generated are written as textual output files.

The acceptable format of input file in MCkeff code is sub set of data format that is allowed in MCNP code. This format is adopted for MCkeff code such that the users can easily run MCNP code with out altering the input file for comparison of results. The input is divided into 3 blocks, namely, Cell block, Surface block, and Data block. MCkeff required all these 3 blocks of information. Each block is separated by a blank line (Line without any character) and some other features enabled are outlined below.

1. First line of cell block must be title line
2. Each line in the input line can be up to 200 characters. A single line can be span across several lines by indicating an ampersand **"&"** at the end of the line or by specifying 5 spaces in the continued line.
3. The line which starts with the character "c" in the first 5 places following space is treated as a comment line. These comment lines can appear in any place of the input file.
4. End of input line comments must be followed by a **"$"** character. The first line of the input is treated as the title of the input problem being solved.
5. The data can be given in free format separated by a blank space

Note: Tab character in the input file will be replaced by single spaces.

A typical input file of highly enriched uranium (HEU) sphere of radius 8.7407 cm is given below with descriptive comment for each line in table 2.1.

Table 2.1: A typical input file for Godiva Reactor

lno	Input data	remark
1	C cell block lines follow	C – indicates Comment
2	Bare HEU critical sphere (Godiva)	Title line
3	1 1 0.04798424 -1 imp:n=1	Cell line-1
4	2 0 1 imp:n=0	Cell line-2
5		Blank line—block separator
6	C surface block cards follow	C – indicates Comment
7	1 so 8.7407	Surface line
8		Blank line—block separator
9	C Data Block lines follow	C – indicates Comment
10	c Godiva sphere at 18.74 g/cm3	C – indicates Comment
11	m1 92234.72c 4.9184e-4	Material line-1

12	92235.72c 4.4994e-2	5 balnks beginning refer to continuation line
13	92238.72c 2.4984e-3 $ U-234	$ end of line comment
14	kcode 10000 1.0 50 100	Data card
15	ksrc 0 0 0	Data card
16		End of input

The input file data blank line numbers 5,8 and 16 represent the block separator. For each block follows different format and whose explanation follows in sequel. Data that is required following cell number,(1-block) surface number(2-block) and keyword (3-block) has a definite format.

The first line of the input is the title line. The 3 blocks of data are separated by a blank line. The cell or surface number or data card keyword must begin within the first five columns. Card entries must be separated by one or more spaces. Input lines cannot exceed 200 columns. The MCkeff input file describes the problem geometry in blocks 1 and 2, in block-3 specifies the materials and kcode control information. The geometry is constructed by defining cells that are bounded by one or more surfaces. Cells can be filled with a material or be void or with universe. Universe is the collection of 2 or more cells.

2.5.1. CELL BLOCK DATA FORMAT

Cells are homogenous material regions bounded by surfaces. Surfaces are analytical linear or quadratic equations identified with a integers. Any complex geometry of the system is defined in terms of the cells. Cell card data of geometry portion contain surfaces with prefix, '+' or '-' sign indicating + ve half space (space right of this surface) and –ve (space left of this surface) . For instance, consider elliptical surface of id=1, -1 in cell card indicates the space with in the elliptical region and 1 or +1 refers to the entire space outside the ellipse and is shown in Fig.2.1.

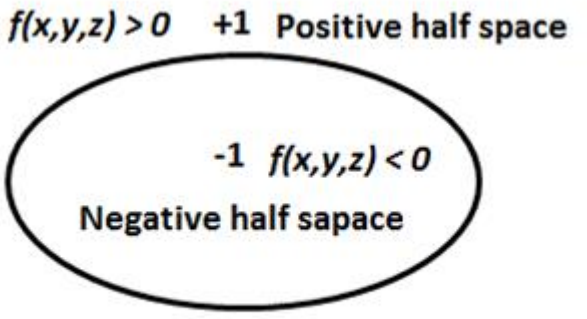


Fig.2.1 Showing negative and positive half spaces

In forming a cell, surfaces are combined with the Boolean operators (Intersection ' ', union ':' complement '#'). The cell number must begin within the first five columns of a line and it must be positive integer < 9999. The specific format for a cell card is:

ID MATERIAL DENSITY GEOMETRY PARAMETERS

ID = Cell ID (+ve integer) starting in column 1 to 5 and < 999.
MATERIAL = Material ID (+ve integer) from which the cell is filled. (if void=0)
DENSITY = Density of the material (prefix - value is g/cm^3 , + value is atoms/barn-cm)
= Absent if the material is "0".

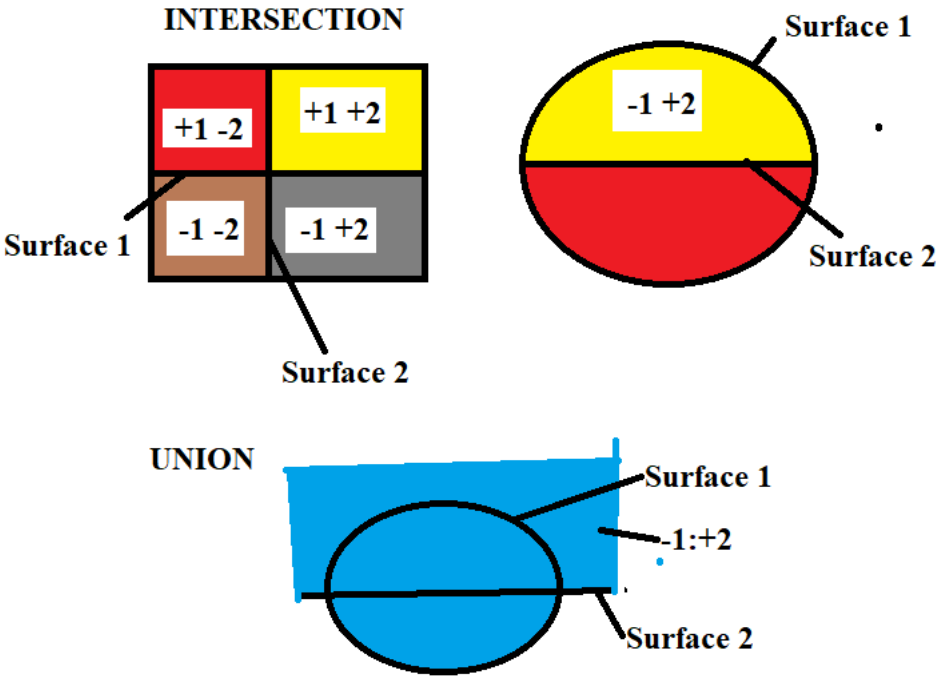
GEOMETRY = List of surface ID with prefix of signs followed by Boolean operators (Intersection space ' ', Union ':' and complement operator '#') defining the cell.

PARAMETERS = other cell parameters such as U, FILL, LAT, TRCL, IMP:N, MAT and RHO

Open and close parentheses can be used in the geometry specification. The geometry specification always must cover the entire space. Undefined regions and overlaps must not be present. Cell also can be defined in short hand form using construct 'LIKE n BUT' with parameters.

Cell Formation with surfaces using Boolean Operators

Typical formation of 2D cells is sown in the figure below. The LHS contains 4 regions which are formed with planes 1 and 2 using the intersection operators (blank) along with signed surfaces. The RHS is a closed elliptical figure and the yellow region is specified as -1 +2. The bottom figure demonstrate the union operator ':' and the region formed is ' -1:+2' .The space is the entire elliptical region coloured with sky blue extending to infinity above surface 2.



PARAMETERS

(i) U and FILL

A cell can be filled with a material or universe, which is a collection of cells called. Cells can be added to a particular universe by specifying the parameter U=ID(+ve integer) , Where ID is the universe ID to which the cell belongs to. Any number of cells can be present in a universe and the collection of cells must span over the entire space. A cell can be filled with a universe and can be specified in the cell line as FILL=ID(+ve integer), where ID is the filling universe ID. The cells of the universe are truncated by the filling cell boundary.

(ii) LAT

A cell can be repeated in the region of space by specifying the Lattice parameter LAT. These repeated lattice units can be filled with same or different universe or material specified in that cell. LAT=1 defines the square lattice structure, where the cell can span over 2 or 3 dimensions. Geometry specification of the Square lattice can have only 4 (2D lattice) or 6 (3D lattice) surfaces with intersection operators. Lattice cell must have FILL parameter which defines the specific region of the lattice cell. FILL can be a single universe ID, where the entire region of the lattice filling cell will be of same universe. FILL can be a specified structure of collection of universes. The syntax of the Fill pattern is given as follows:

FILL= n (n is universe id which is infinite lattice truncated by the boundary of filling cell)

FILL=-1:1 -1:1 0:0 3 4 3 4 3 4 1 2 3 (2D lattice)

FILL=0:1 0:1 0:1 3 4 3 4 3 4 1 2 (3D lattice)

2D square lattice

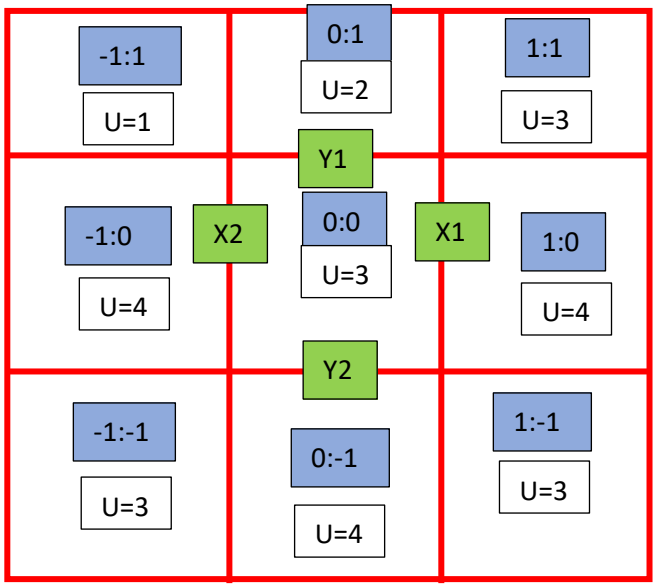
The cell line is:

ID material density X1 X2 Y1 Y2 FILL=IXL:IXU IYL:IYU list of universe IDs u=ID

Where,

X1 = First surface listed (must be x-plane)
X2 = Second surface listed (must be x-plane)
Y1 = Third surface listed (must be y-plane)
Y2 = Second surface listed (must be y-plane)
IXL, IXU = Lower and Upper range of x-dimension index (Eg., -1:1)
IYL, IYU = Lower and Upper range of y-dimension index (Eg., -1:1)

In this specified range, the defined cell corresponds to the 0:0 element as shown in figure 1. X-index increases on first surface listed X1, and decreases on the second surface listed X2, similarly on Y index.



FILL=-1:1 -1:1 0:0 3 4 3 4 3 4 1 2 3 (2D lattice)

(iii) IMP:N

IMP:N corresponds to Neutron importance of the cell. This parameter can have the value 0 or n (+ve integer). If the neutron importance of a cell is zero, then the neutron will not be tracked in that cell. Usually, a single cell which bounds the whole geometry given zero importance. If n >1 , refers to splitting of 1 track to n tracks and reducing the weight of the particle correspondingly. This is one of the variance reduction techniques and not implemented in the MCkeff code at present.

(iv) TRCL

This key word is useful for relocating the cell earlier defined at a different location. A cell can be translated using the TRCL parameter. Translation can be given as:
TRCL=(x,y,z), co-ordinates to which cell must be translated
Or
TRCL=n (TR number specified in the data block)

(v) LIKE n BUT

This phrase is used for relocating the earlier defined cell at different location with change in material (MAT=n) and density (RHO=xxx).

2.5.2. SURFACE DATA BLOCK FORMAT

As pointed earlier, cells are formed by surfaces and combining them with binary operators. In MCkeff, we limit to the use of 21 different type analytical surfaces of first and second degree. Surfaces can be defined with the following syntax in the surface block,

ID PNEUMONIC COEFFICIENTS

Where,

ID	= Surface ID(+ve integer < 999) in columns 1 to 5
MNEUMONIC	= Surface type
CO-EFFICIENTS	= Surface co-efficient

The surfaces used in the MCkeff code , the data required and analytical equation is given in table 2.2.

Table-2.2. List of analytical surfaces allowed in MCkeff and the data entries required

Sno.	Mnemonic	Type	Description	Equation f(x,y,z)=0	Data entries
1	P	Plane	General	$Ax + By + cZ - D = 0$	A,B,C,D
2	PX		Normal to X-axis	$x - D = 0$	D
3	PY		Normal to Y-axis	$y - D = 0$	D
4	PZ		Normal to Z-axis	$z - D = 0$	D
5	So	Sphere	Centered at origin	$x^2 + y^2 + z^2 - R^2 = 0$	R
6	S		General	$(x - x_0)^2 + (y - y_0)^2 + (z - z_0)^2 - R^2 = 0$	x_0, y_0, z_0, R
7	Sx		Centre on X-axis	$(x - x_0)^2 + y^2 + z^2 - R^2 = 0$	x_0, R
8	Sy		Centre on Y-axis	$x^2 + (y - y_0)^2 + z^2 - R^2 = 0$	y_0, R
9	Sz		Centre on Z-axis	$x^2 + y^2 + (z - z_0)^2 - R^2 = 0$	z_0, R
10	C/X	Cylinder	Parallel to X-axis	$(y - y_0)^2 + (z - z_0)^2 - R^2 = 0$	y_0, z_0, R
11	C/Y		Parallel to Y-axis	$(x - x_0)^2 + (z - z_0)^2 - R^2 = 0$	x_0, z_0, R
12	C/Z		Parallel to Z-axis	$(x - x_0)^2 + (y - y_0)^2 - R^2 = 0$	x_0, y_0, R
13	CX		On X-axis	$y^2 + z^2 - R^2 = 0$	R
14	CY		On Y-axis	$x^2 + z^2 - R^2 = 0$	R
15	CZ		On Z-axis	$x^2 + y^2 - R^2 = 0$	R
16	K/X	Cone	Parallel to X-axis	$\sqrt{(y - y_0)^2 + (z - z_0)^2} + t(x - x_0) = 0$	$x_0, y_0, z_0, t^2 \pm 1$
17	K/Y		Parallel to Y-axis	$\sqrt{(x - x_0)^2 + (z - z_0)^2} + t(y - y_0) = 0$	$x_0, y_0, z_0, t^2 \pm 1$
18	K/Z		Parallel to Z-axis	$\sqrt{(x - x_0)^2 + (y - y_0)^2} + t(z - z_0) = 0$	$x_0, y_0, z_0, t^2 \pm 1$
19	KX		On X-axis	$\sqrt{y^2 + z^2} + t(x - x_0) = 0$	x_0, t^2
20	KY		On Y-axis	$\sqrt{x^2 + z^2} + t(y - y_0) = 0$	y_0, t^2
21	KZ		On Z-axis	$\sqrt{x^2 + y^2} + t(z - z_0) = 0$	z_0, t^2 ± 1 *is used only for one shhet co*

2.5.3. DATA BLOCK FORMAT

The Data block contains information on materials and the data required to control keff calculations. All of the cards begin with keyword. For instance 'm' stands for material, 'ksrc' stands for initial source points and 'ksrc' contain control information on the calculation of keff calculations. These keywords and the data required are given below.

(i) MATERIALS(M keyword)

Material card contains the information on the material identification and composition of nuclides.

ID nuclide1 fraction1 nuclide2 fraction2 ……. Nuclide n fraction n

ID = Material ID (Eg., m100)
Nuclide with fractions = ZAID.XXc^1 fraction^1 ZAID.XXc^2 fraction^2
(Fraction with positive sign = Atom fraction)
(Fraction with negative sign = Weight fraction)
In a given material all nuclides must be specified either with atom fraction or with weight fraction.

(ii) S(a,b) tables (MT keyword)

Special treatment for kinematics of elastic scattering is required for some nuclides at around thermal energies (< 4 ev). They are moderating materials such as H_2O, BeO, graphite etc.

SAB table corresponds to a material must be specified with the same ID as:
M100 some material
MT100 name of the nuclide in the material 100….
MT line must be followed by the corresponding material specifications.

(iii) CELL IMPORTANCE (IMP:N keyword)

Cell importance can also be specified in the data block as: IMP:N list of cell importance (0 or 1). (Number repeater "r" can also be used here as IMP:N 1 5r 0)
Number of entries must match with the number of cells defined

(iv) SOURCE POINTS (KSRC keyword)

Source points for the initial batch must be specified with the KSRC parameter in the data block. The points can be specified as:
KSRC x1 y1 z1 x2 y2 z2 x3 y3 z3 x4 y4 z4
If the card is absent, the default value is (0,0,0)

(v) KCODE

Number particles that need to be simulated, the total number of batches and the number of batches that needs to be skipped before tallying must be specified with KCODE in data block as;

KCODE PARTICLES initial-Keff SKIP TOTAL

(vi) PRINT

In the data block if the parameter print is present, then the output file additionally does stochastic volume calculation results (Only if a single base universe is present) and first 50 source points information.

(vii) Cell or Surface translation (TR keyword)

Translation co-ordinates can also be given in the data block as:

TRn x y z
Where n = translation ID.

These ID can be used while specifying cell as TRCL=ID

2.6. Output file Format

Output file is divided into several tables. The output file contains the MCNS title and input file name. The following tables of information are present in the output file. A sample output file is given at the end of this document.

2.6.1. User input

A same copy of the user input is present after the identifier “USER INPUT”, and the path in which xsdir found will be written on output.

2.6.2. CELLS

This table contains all the cells information, such as cell id, material, cell universe, filling universe number, cell importance and cell translation co-ordinates.

2.6.3. SURFACE

This table contains the surface information such as surface id, type of the surface and the surface coefficient.

2.6.4. MATERIALS

This table contains the material specified with their nuclide and atom fractions.

2.6.5. Stochastic volume calculation table

This table contains cell volumes in units of a cubic centimetre. This table is present only if PRINT parameter is present in the data block and all the cells belong to the Base universe.

2.6.6. Source particles

This table contains information on the first 50 source points. Source point coordinates, direction, and particle energy will be printed in this table.

2.6.7. Batch Keff and Results

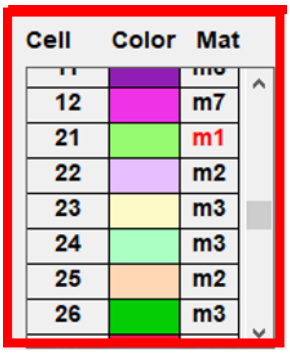
Finally, Batch wise track length keff is given for all the batches. For the Active batches, the average track length is given in addition.
The result contains final, Absorption, Collision, Track length, and combined k-effective.

2.7.1. The Geometry plotter

Here additional details of Geometry plotter is given. Please refer to installation guide to understand how to get geometry plots.

2.7.2. Colour Table in geometry plotter:

The numbers in the cell column are the cell IDs, and the "colour", "mat" are corresponding plot colour and the material from which cell is filled. If the Material ID is given a red colour, it corresponds to fissionable material.



2.7.3. Geometry Checking:

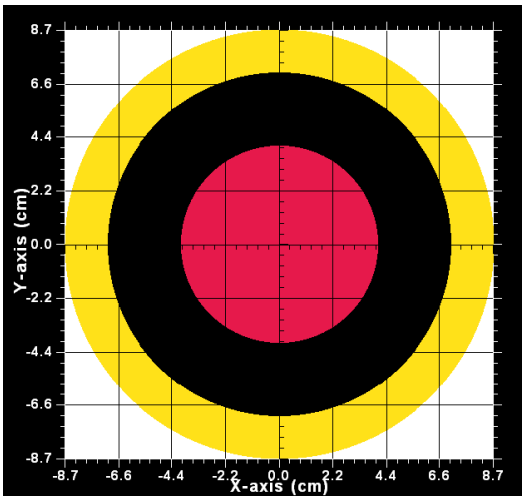
The geometry plotter catches the user geometry errors. Make sure to check the geometry plot of the problem before the simulation. The problem of geometry must span over all the space and no overlapping of cells must occur. If the **geometry plotter finds any undefined region or Overlapping region it gives BLACK colour to that region of space.** If you find such occurrences of Black colour in geometry make sure to correct the geometry before the simulation. If you find any difficulties getting the plot, try "**-po**" argument in place of "**-p**".

2.7.4. Undefined region

Geometry error – undefined region

```
1   1 -18.74     -1   imp:n = 1
2   1 -18.74   2 -3   imp:n = 1
3   0                3   imp:n = 0

1    so    4.0
2    so    7.0
3    so    8.741
```



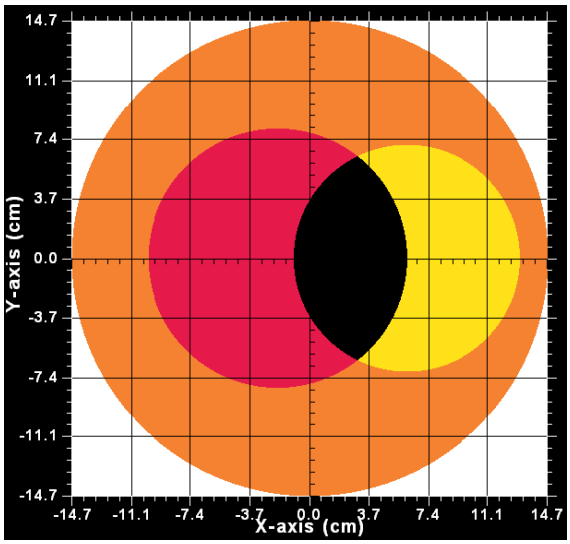
It is clear from the input that, the spherical region from 4 to 7 cm is not defined in the geometry lines, hence the region has been given black colour.

2.7.5. **Overlapping regions**

Geometry error – overlapped region

```
1   1 -18.74   -1          imp:n = 1
2   1 -18.74   -2          imp:n = 1
3   0           -3 #1 #2   imp:n = 1
4   0            3         imp:n = 0

1   sx   -2.0   8.0
2   sx    6.0   7.0
3   so    14.741
```



From the above input it is clear that the spherical portion of the cells 1 and 2 overlaps in the space from -1.0 to 6 cm, hence the black colour in the geometry

3. Radiation Transport – Monte Carlo Method

This chapter presents the basic theoretical frame work required for estimation of neutron multiplication factor (k_{eff}) of fissile systems. We start with the time dependent neutron transport equation that governs the distribution of neutron population (called flux) in phase space element. The phase space element consists of space (3 coordinates), direction (2 coordinates), energy(one coordinate) and time (one coordinate). Various methods of solving the equation in brief and Monte Carlo Method of solving in detail are presented. Further, casting of steady state transport equation in to eigen value form and the power iteration method of solution is discussed. The converged eigen value which represents the neutron multiplication of the fissile system (k_{eff}), which corresponds to the steady state solution of the equation. Further, it is possible to estimate eigen values by different ways and its interpretation is discussed towards the end of this chapter. Often, particle or neutron may be used in the following text for radiation.

3.1. Introduction to Transport equation

Basic aim of radiation transport theory is to predict the particle distribution N (r,E,Ω,t) in phase space variables(space(r), energy(E) , angle(Ω) and time (t)) . To get the particle distribution, it is required to solve the radiation transport equation. The equation that governs the of time evolution of neutral particle population or flux N(r,E,Ω,t) in phase space (r,E,Ω,t) element and is given by[1]

$$\frac{1}{v}\frac{\partial N}{\partial t} + \Omega * \nabla N + \Sigma_t * N = \iint [< \nu > * \Sigma_f + \Sigma_s] P(\mathrm{r}, \mathrm{E}', \Omega' \rightarrow \mathrm{E}, \Omega) N \, dE' \, d\Omega' + \mathrm{S} \quad (3.1)$$

N (r,E,Ω,t)	- neutron flux
S (r,E,Ω,t)	- flux independent source
Σ_t (r,E,Ω)	- total cross section
Σ_f (r,E,)	- fission cross section
Σs(r,E,)	- scatter cross section
<ν(r,E)>	- average number of neutrons emitted per fission
P (r,E', Ω' -> E, Ω)	- probability of transfer from (E', Ω') to (E, Ω)
v	- neutron speed (not velocity)

Here the production term $< \nu > * \Sigma_f$ actually represents all fission neutron production, i.e., it includes (n,2n), (n,3n), etc. This is the fundamental integro-differential equation and solving it one can deduce many parameters are of practical interest. In case of neutrons, some of these parameters are energy deposition rate, neutron multiplication factor, nuclear reaction rates etc. In general, most often the solution of steady state equation is sought, which can be obtained by setting *dN/dt =0 as*

$$\Omega * \nabla N + \Sigma_t * N = \iint [< \nu > *] \Sigma_f P(\mathrm{r}, \mathrm{E}', \Omega' \rightarrow \mathrm{E}, \Omega) N \, dE' \, d\Omega' + \iint \Sigma_s \; P(\mathrm{r}, \mathrm{E}', \Omega' \rightarrow \mathrm{E}, \Omega) N \, dE' \, d\Omega' + \mathrm{S} \quad (3.2)$$

The equation (3.2) is a steady state neutron balance equation between the loss and gain. The LHS terms represents neutron loss due to leakage and absorption (geometry and material properties) and on the other hand RHS terms represent gain due to production of fission neutrons, scattering from other phase space (r,E',Ω') in to this phase space (r,E,Ω) and constant external source (S) respectively. The implicit assumptions made in deriving the equation are:

- Static, homogeneous medium
- Time-independent
- Markovian – next event depends only on current (r,v,E), not on previous events
- Particles do not interact with each other

- Neglect relativistic effects
- No long-range forces (particles fly in straight lines between events)
- Material properties are not affected by particle reactions
- Etc.

3.2. Solution methods

For steady state radiation transport equation (2.2), solution is sought in many areas such as nuclear reactors, spent fuel pools, fuel reprocessing facilities, nuclear fuel storage, nuclear weapons and in radiation shielding. Analytical solution is not possible to the equation (2.2) since the functional forms in E are very complex for scattering and fission cross sections. Therefore numerical solutions are attempted. There are two distinct computational methods of solution that are in vogue. They are:

(i) Deterministic methods, where the transport equation is discretized in space(r), energy(E), and angle(Ω) either to solve directly or iteratively. Different types of discretization give rise to different deterministic methods[2,3]], such as discrete ordinates (S_N), spherical harmonics (P_N), collision probabilities, nodal methods, and others. The solution obtained from this method is for flux and the parameters of interest can be deduced from it.

(ii) Monte Carlo methods are well suited for the problems whose evolution is influenced by the random factors. The equilibrium neutron population distribution is formed after the many different type of collisions between the neutron and medium, which are stochastic in nature. Hence, Monte Carlo methods are preferred over the others. Besides, it is capable of solving the transport equation continuous in space (r), energy (E), and angle (Ω) while other methods cannot handle. The disadvantage in this method is that it does not solve transport equation directly for neutron flux from which physical parameters are deduced. Instead, it estimates the expected value of a physical quantity of interest by drawing independent samples randomly while the neutron passes through the medium. A good description of Monte Carlo method can be found in a book by Carter and Cashwell[4] on " *Particle-Transport Simulation with the Monte Carlo Method*" and in a more recent book is by Lux and Koblinger[5] on "*Monte Carlo Particle Transport Methods: Neutron and Photon Calculations".* Methods of sampling from standard probability densities are discussed in the Monte Carlo samplers by Everett and Cashwell[6]. Monte Carlo methods are also employed to get the solution of deterministic equations by constructing a stochastic model of the equation artificially.

3.3. Monte Carlo Method

Since the current project is on obtaining a solution to neutron transport equation by Monte Carlo method, some details of various functional modules of Monte Carlo method are given in sequel. They are:

1. Pseudo random number generator-uniform pdf (0,1)
2. Probability density functions (cross sections) and sampling theory
3. geometry of the system and ray tracing method
4. Tallies-scoring parameters of interest
5. Error estimation
6. Variance reduction techniques, and
7. Parallel computers

A brief description of each functional module is described below.

3.3.1.Pseudo random number generator (PRNG)

Pseudo random numbers are the back bone of all the Monte Carlo methods and are used in the selection of stochastic variables in the problem. This is done by equating cumulative distribution of non-uniform stochastic variables to cumulative distribution of uniformly distributed random numbers in the interval (0,1). Most computer programming languages include functions or library routines that generate pseudo random numbers. The minimum desirable properties of random numbers are:

- Uniformity in the interval [0,1) shown in Fig.2.1
- Long period (depends on the largest integer on the Computer)
- Minimum correlation and do not have any easily discernible patterns.
- Reproducible (same sequence helps in debugging the program)
- Less CPU time

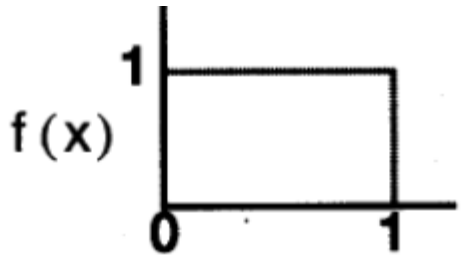


Fig.2.1 Uniform distribution function

The widely used standard generator of random numbers is the prime modulus multiplicative linear congruential generator[7]. The method of generating sequence of integers Z_n in the range 1 to N and a corresponding sequence of floating point numbers ξ_n in (0,1) is by means of the following formulas:

$$z_{i+1} = gz_i + c \ MOD\ N \ , and \ \xi_{i+1} = z_{i+1}/N \qquad (3.3)$$

Where

N is a large prime integer,
z_{i+1} is the $i+1^{th}$ number in the sequence,
z_i is the previous number (called seed), and
g is a constant multiplier.

The choices of g and N are critical in achieving a desired goal. For example a=6 and N=13 gives the full period sequence 1,6,10,8,9,2,12,7,3,5,4,11. However, a=5 and N=13 gives the repeating sequence 1,5,12,8 with cycle length of 4 (Expected 12). Lehmer recommended that N be a Mersenne prime, of the form 2^p-1, valid for p=2,3,5,7,13,17,19,31… Park and Miller[8] suggested to use N= 2^{31}-1 =2147,4836,47 and a= 7^5 = 16807 which gives random numbers having full period(~ 10^9) and is demonstrably very nearly random. Often in radiation transport of deep penetration problems the cycle length may not be enough and hence some programs use random number generators based on 2 seeds to get a cycle length of 10^{18}. The constants for the linear congruential generator for various machines is given hand book by L'Ecuyer[9] used by default for 64 bit computers are g =

2806196910506780709, c = 1, and N = 2^{63} (see L'Ecuyer). Chaptre-7 of this document provides a list of random numbers and their properties.

PRNG for Parallel computers

For running Monte Carlo code on machines with number of CPUs, where the work (total number of particles) is divided among the available processors, and to reproduce the results of a single processor, the PRNG must be able to generate seed for each particle by skipping a predetermined numbers(152,917 in MCNP). Without this capability, it would be very difficult to maintain reproducibility in a parallel calculation. Fortunately, algorithms have been developed that allow us to skip ahead in (log2 N) operations instead of (N) operations. One algorithm to do so is described in a paper by Brown[10]. This algorithm relies on the following relationship:

$$z_{i+k} = g^k z_i + c\frac{g^k-1}{g-1} \; MOD\; N \;, and \; \xi_{i+1} = z_{i+1} \,/\, N \tag{3.4}$$

Note that the above has the same general form as equation(2.3), so the idea is to determine the new multiplicative and additive constants in O(log2 N) operations. The period (k) is chosen to be a large integer enough to run simulation of a single particle.

3.3.2. Probability density functions(cross sections) and sampling theory

The key to Monte Carlo methods is the notion of random sampling of stochastic variables. These variables are generally non-uniform in distribution and are sampled by equating them uniform distribution function in the interval (0,1). As is well known that neutron undergoes different types of interactions with nuclei and are dependent on neutron energy. The important interactions are elastic, inelastic, fission, capture and other particle emission reactions. Since these interaction cross sections are not smooth functions of energy, they are tabulated in close intervals of energy and linearly interpolatable between the nodes. Such data files deduced from basic ENDF/B files using NJOY[11] code in to a suitable format. One such format is A Compact ENDF file or generally known as ACE formatted data files. In addition, these files contain secondary neutron energy-angular distributions. Monte Carlo codes make use of these data files for sampling of interaction events randomly using the pseudo random numbers. All programs read these and build the required probability and cumulative distribution tables before sampling. The type of random sampling technique, getting non-uniform variate from uniform random number ξ, depends on the form input data available and they can be broadly classified as follows:

i. Direct Inversion Method or cumulative probability distribution (cpd) Method,
ii. Table Lookup + Interpolation Method
iii. Rejection Method
iv. Composition Method, and
v. Composition-Rejection Method

Details of these methods are discussed in detail in chapter-3, which deals with the physics of sampling.

3.3.3. Geometry of the system and Ray tracing method

Complex 3D geometry of the system can be constructed based on either few of the known 3D shapes or few basic surfaces enclosing these volumes. The popular Monte Carlo codes such as KENO[12], TART[13], SERPENT[14] built on 3D shapes while MCNP[15], OPENMC[16] use surfaces

to build the geometry of the system. In MCkeff code, we adopt the surfaces to build geometry of the system similar to that of MCNP code. The details of geometry specification and particle tracking algorithms are given in chapter 4.

3.3.4.Tallies and scoring

This is important module which computes the results of Monte Carlo simulation. The results depends on the purpose for which Monte Carlo simulations are carried out. Most common results sought after are neutron flux, energy deposition rate, reaction rates and neutron multiplication factor (keff) in case of fissile systems. MCkeff developed is for the estimation of keff in fissile systems, the result is obtained by three different ways namely absorption estimator , tracklength estimator and collision estimator. The algorithms are described in this chapter under section 3.6 on eigen value estimators.

3.3.5. Error Estimation

In Monte Carlo simulation of histories of particle are independent and the error associated with estimators can be easily estimated. An *estimator (K)* is a specific function of the random samples, of a random variable, that statistically represents a true unknown mean. If *k* is a random variable with an associated distribution and an unknown mean, then the function $K(k_1, k_2, k_3,...,k_n)$ is an estimator of the unknown mean. The set $\{k_1, k_2, k_3,...,k_n\}$ consists of n independent random samples selected from the probability density distribution of *k*. An estimator is *unbiased* if its expected value equals the true mean (μ). Since each simulation is independent, the error or variance associated with the expected value is proportional to the sample size. More description is provided in the section 3.8 under statistical interval estimates.

3.3.6. Variance reduction techniques

The estimated expectation value accuracy (precision) mainly depends on the sample size. However, it is often experienced for problems of large dimensions having many independent variables, the desired accuracy of the solution can not be accomplished by just increasing the sample size. To circumvent this difficulty, one has to resort to advanced Monte Carlo methods, which employ variance reduction techniques. Some of the commonly known variance reduction techniques are (i) implicit capture (ii) exponential biasing (iii) importance sampling (iv) weight window generation and (v) Russian roulette. The detailed description is found in MCNP manual[15]. Suitability of a particular technique largely depends on the problem and judicious selection of it is an art which demands considerable experience of the user. In MCkeff, we employ implicit capture and Russian roulette variance reduction methods as optional.

3.3.7. Parallel computers

Monte Carlo particle transport algorithms are inherently parallel because each particle history can be simulated independently and concurrently on separate processors. In Monte Carlo methods hundreds of thousands or millions of particle histories are simulated using random numbers, highly accurate representations of particle reaction probabilities and exact models for 3-D geometries. The principal limitations for Monte Carlo methods are the requirement to simulate many particles to achieve an acceptable statistical uncertainty, which may take much of CPU time. To reduce CPU time, Monte Carlo method provides ample incentive to utilize the computational power of modern vector (OpenMP[17-19])and parallel supercomputers. The Message Passing Interface (MPI[18]) specification has become a standard for message-passing libraries for parallel computations. There exist more than a dozen implementations on a variety of computing platforms. If enough memory

is available to each processor, the Monte Carlo code and data can be replicated, and each processor can independently follow a portion of the total particles. This is possible due to the statistical independence of the particle histories. Many Monte Carlo particle transport codes such as MCNP[15], OpenMC[16] are being used extensively on parallel computers. MCkeff adopts both openMP and MPI in programming on linux parallel computer.

3.4. Steady state Transport equation in the eigen value form

In a non-multiplying medium with a steady state source, there exists a steady state flux distribution(*N*) that satisfies this equation. In fissile systems with out external source, it is difficult to find the solution of steady state equation, which demands the combination of cross sections and geometry that will allow equation to be satisfied. Therefore for multiplying systems, the transport equation solution is studied by introducing the concept of criticality. Physically, a system containing fissionable material is critical if a self-sustaining time independent chain reaction in the absence of external neutron sources can be maintained. The transport equation is casted in the form of an eigenvalue problem[20] by dividing the average number of fission neutrons emitted by a constant K and it can be written as

$$\Omega * \nabla N + \Sigma_t * N - \iint \Sigma_s P(\mathrm{r}, \mathrm{E}', \Omega' \rightarrow \mathrm{E}, \Omega) N \, dE' \, d\Omega' = \iint \frac{<\nu>}{K} * \Sigma_f P(\mathrm{r}, \mathrm{E}', \Omega' \rightarrow \mathrm{E}, \Omega) N \, dE' \, d\Omega' \quad (3.5)$$

This is called an eigen value equation because on both sides neutron flux occurs with a constant multiplier[20]. The eigenvalue *K* provides a measure of the criticality of the system. It can be thought of that K can be defined as the quantity by which ν must be divided to keep a system critical, i.e., ν can be adjusted to obtain a time independent solution to equation (3.5). The system is critical if this largest value of K is equal to unity. If K < 1 the system is subcritical, then <ν>, the actual number of neutrons per fission emitted is less than necessary to maintain criticality and <ν>/K, provides the required number of neutrons per fission which can maintain the criticality. Similarly for K > 1 the system is supercritical, the actual number of neutrons per fission must be less than actual <ν>, and <ν>/K, gives the number of neutrons per fission required to maintain the system critical. The multiplication eigenvalue problem, equation (3.5), can be written in operator notation as

$$H\,N = \frac{\nu}{K} F N \qquad (3.6)$$

Where *H* is a transport operator and *F* is a fission source density and is defined

$$H = \Omega * \nabla N + \Sigma_t * N - \iint \Sigma_s F(\mathrm{r}, \mathrm{E}', \Omega' -> \mathrm{E}, \Omega) N \, dE' \, d\Omega' \qquad (3.7)$$

$$F = \iint \Sigma_f P(\mathrm{r}, \mathrm{E}', \Omega' -> \mathrm{E}, \Omega) N \, dE' \, d\Omega' \qquad (3.8)$$

Inverting *H* and operating on both sides by *F*, we obtain the following form of the eigenvalue problem:

$$K = \frac{[F H^{-1} \nu] F N}{F N} \qquad (3.9)$$

And the physical meaning associated with the above equation is

$$K = \frac{Number\ of\ fission\ neutrons\ produced\ in\ the\ current\ generation}{Number\ of\ neutrons\ from\ the\ source} \qquad (3.10)$$

Here *FN* represents the fission source in the system. This is an Eigen value equation and the solution is obtained by guessing initial source points and guess estimate of k by power iteration method[19,21].

3.5 Eigenvalue (k) Calculations

A largest eigenvalue of the system (converged solution) is calculated by Monte Carlo simulation from the initial guess values in fissile systems. In Monte Carlo the recording of events done while neutron traversing the medium and it is called a "history" of a particle. A typical history for two particle histories are shown in Fig.3.1.

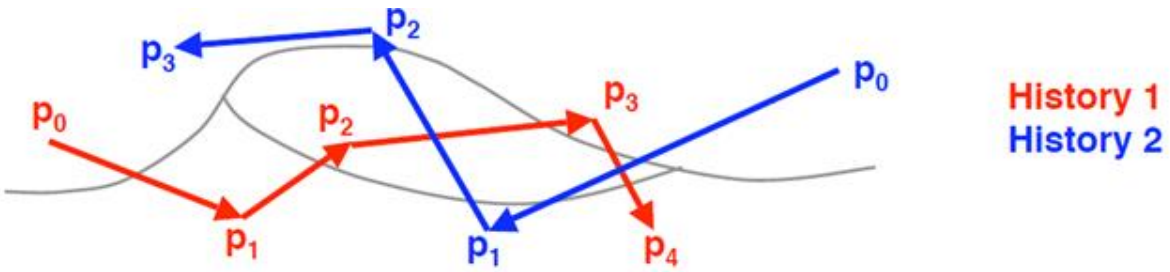


Fig.3.1. traces history , sequence of states (p1,p2,p3,...) which are used for tallying purposes in a given region

Although each history may not have any physical meaning, however average over a large number of histories represents a physical observable quantity. The events of a neutron sampled randomly are the physical processes such as free flight distance, type of collision such as fission, scattering, transmission, leakage etc. Since, the initial source points are not known, we start with the guess distribution and iterate till the convergence16. In an eigenvalue calculation usually method of successive generations/cycles is followed to estimate neutron multiplication factor, k with N neutrons (thousands) for a batch over several batches or cycles or batches (N^c). The user will start simulation either with a guess of uniform spatial distribution over some region of the geometry or simply a point source and with a guess value of estimator *k* is unity. Fortunately, regardless of the choice of initial source distribution, the method is guaranteed to converge to the true source distribution. At the end of each fission generation, fission sites occurred (*M*) may not be *N* for use in the next generation. One way is stored fission sites may be randomly sampled from the *M* fission sites such that for each generation only *N* particles are started. The other way is weight (W) of source particles is adjusted (W*N/M) in such way that always same estimator is simulated. Starting source distribution (guess) may not be characteristic distribution of the fissile system, hence few generation estimators (until source distribution converges) are not used for final estimators.

The following information is provided to a Monte Carlo eigenvalue simulation algorithm:

- the nominal number of source histories per batch;
- the initial guess of K;
- the number of initial batches, I_i, to be discarded before tally accumulation;
- the total number of batches, I_t, for the problem; and an initial spatial distribution for the source

In criticality calculation, three types of estimators are deduced from Monte Carlo simulations. We use the superscripts C, A and TL for the collision, absorption and track length estimators respectively.

3.5.1. Collision Estimator (K^c):

The collision estimate of K for an active batch is:

$$K^c = \frac{1}{N}\sum_j \sum_i W_i \frac{< \nu > \Sigma_f}{\Sigma_t} \qquad (3.11)$$

where,
i = all collisions for a particle in regions where fission is possible;
j = all source particles for a batch;
N = number of source particles for a batch; and
W_i = weight of particle entering collision.
Then the quantity $W_i \frac{<\nu>\Sigma_f}{\Sigma_t}$ in equation (2.9) is the expected number of neutrons to be produced from a fission process in collision i. Hence K^c is the mean number of fission neutrons produced per generation (or batch), representing the Eigen value (K)of a critical system.

3.5.2. Absorption Estimator(K^A)

The absorption estimator of K^A for an active batch:

$$K^A = \frac{1}{N}\sum_j \sum_i W_i \frac{<\nu>\Sigma_f}{\Sigma_a} \qquad (3.12)$$

where,
i = an analog capture event; and
Σ_a= macroscopic absorption cross section.

3.5.3. Track Length Estimator(K^{TL})

The track length estimator for K^{TL} is given by,

$$K^{TL} = \frac{1}{N}\sum_j \sum_i d\, W_i \frac{< \nu > \Sigma_f}{\Sigma_t} \qquad (3.13)$$

where,
d = track length segment;
i = all track length segments ,d, for a particle; and
j = all source particles for a batch.

3.6. Statistics for Monte Carlo K Calculation

As mentioned previously, the Eigen values estimated do not include contribution of few batches (N_{ina}) before source converges over a total number of batches (N_t). Then, the number of

active batches, N_a that are used to evaluate the mean and the standard deviation for K is then given by,

$$N_a = N_t - N_{ina} \tag{3.14}$$

A Monte Carlo eigenvalue simulation provides the following estimate of the average eigenvalue,

$$K = \frac{1}{N_a}\sum_{i=1}^{N_a} K_i \tag{3.15}$$

where K_i is the batch eigenvalue. The *sample variance* in this estimate of K is

$$\sigma_s^2 = \frac{\sum_{i=1}^{N_a} K_i^2}{N^a - 1} - \frac{(\sum_{i=1}^{N_a} K_i)^2}{N^a(N^a - 1)}$$

The *standard deviation* is then

$$\sigma = \frac{\sigma_s}{\sqrt{N^a}}$$

which is provided along with the average eigenvalue estimate. The expressions for the variance and standard deviations assume that the batch eigenvalues are independent, hence no batch-to-batch correlations exist after the fundamental eigen mode is reached.

In criticality calculations at the end of each cycle or batch (or fission generations) an estimate of K is calculated by each of these three estimators. The final K estimator of each type is the average over many batch K estimates. Recent studies[22] have suggested that a combination of the three estimators to be the best K estimate available. It should be emphasized that for Monte Carlo criticality calculations, the final result is not a point estimate of K, but rather a confidence interval. The confidence interval based on the three-statistically-combined K estimator is the recommended result to use for all final K confidence intervals because all of the available information has been used in the final result. This estimator often has a lower estimated standard deviation than any of the three individual estimators and therefore provides the smallest valid confidence interval as well.

3.7.Issues Related to Monte Carlo K Calculation

- The first issue that needs to be addressed regarding Monte Carlo K calculations is that of source convergence. Before accumulating any K tally data, enough batches must be performed and discarded to allow the source neutron distribution to attain the fundamental mode.
- The second issue deals with the assumption that the batch eigenvalues are independent. For systems batch-to-batch correlations among fission neutron distribution exist, and this assumption is invalid.
- The third issue involves the choice of the optimum number of batches versus the optimum number of neutrons per batch. The user must obtain the correct number by trail and error.

References

1. G. I. Bell and S. Glasstone, Nuclear Reactor Theory, Van Nostrand Rienhold, New York (1970).

2. J. J. Duderstadt and W. R. Martin, Transport Theory, John Wiley & Sons, New York (1979).
3. E. E. Lewis and W. F. Miller, Jr. , Computational Methods of Neutron Transport, American Nuclear Society, Inc., La Grange Park, Illinois, (1993).
4. L. L. Carter and E. D. Cashwell, Particle Transport Simulation with the Monte Carlo Method, ERDA Critical Review Series, TID-26607 (1975).
5. I.Lux and L. Koblinger, Monte Carlo Particle Transport Methods: Neutron and Photon Calculations, CRC Press, Florida (1991).
6. C.J. Everet and C.D. cashwell, A third Mont Carlo sampler,La-9721-MS, March, (1983)
7. D.H. *Lehmer, "Mathematical methods in large-scale computing units"*. Proceedings of a Second Symposium on Large-Scale Digital Calculating Machinery. *pp. 141–146, (1951)*
8. P*ark, Stephen K.; Miller, Keith W. (1988).* "Random Number Generators: Good Ones Are Hard To Find" (PDF). Communications of the ACM. ***31** (10): 1192–1201*. doi:10.1145/63039.63042.
9. P. L'Ecuyer - Handbook of computational statistics, 2012
10. F.B. BROWN, "Random Number Generation with Arbitrary Strides," Trans. Am. Nucl. Soc. (Nov., 1994)
11. R. E. MacFarlane and D. W. Muir. The NJOY Nuclear Data Processing System. LA-12740-M. Los Alamos National Laboratory, 1994.
12. L. M. Petrie, N. F. Landers, KENO V.a: AN IMPROVED MONTE CARLO CRITICALITY, PROGRAM WITH SUPERGROUPING, ORNL/NUREG/CSD-2/R6,September,(1998)
13. Dermott E. Cullen , "TART 2002: A Coupled Neutron-Photon, 3-D, Combinatorial Geometry, Time Dependent Monte Carlo Transport Code", UCRL-ID-126455, Rev. 4, (June 2003), Lawrence Livermore National Laboratory.
14. Jaakko Leppänen, " Serpent – a Continuous-energy Monte Carlo Reactor Physics Burnup Calculation Code ",User's Manual, March 6, 2013
15. S. M. Girard, editor. MCNP – A General Monte Carlo N-Particle Transport Code, Version 5 Volume I: Overview and Theory. LA-UR-03-1987. Los Alamos National, Laboratory, 2003.
16. Paul K. Romano, OpenMC OpenMC: A State-of-the-Art Monte Carlo Code for Research and Development, https://www.openmp.org
17. https://www.openmp.org,"OpenMP Fortran Application Program Interface"
18. MPI: The Complete Reference (2-volume set) 2nd Edition by Marc Snir etal.
19. S. MATSURA, F. B. BROWN, R. N. BLOMQUIST, "Parallel Monte Carlo Eigenvalue Calculations," Trans. Am. Nucl. Soc. (Nov. 1994).
20. https://www.sdsc.edu/~majumdar/thesis
21. Lieberoth, J., 1968. A Monte Carlo technique to solve the static eigenvalue problem of the Boltzmann transport equation. Nukleonik 11 (5), 213–219.
22. Todd J. Urbatsch, R. Arthur Forster, Richard E. Prael & Richard J. Beckman,(1995) Estimation and Interpretation of keff Confidence Intervals in MCNP, Nuclear Technology, 111:2, 169-182, DOI: 10.13182/NT95-A35128

4. Physics and Sampling Theory

4.1. Introduction

This chapter presents the sampling theory which is the key component of all Monte Carlo codes. For neutron transport, sampling of free-flight-distance, type of collision, energy-angle distributions of secondary neutrons is required. The sampling depends on the type of data is present (Tabular or functional form). The sampling is done by equating cumulative probability of uniformly distributed function (Random number) to that of cumulative distribution of non-uniform distributed function. Since the tally results are averaged over outcomes accrued over a large number of initial particles, and the outcomes may be scattered widely around the mean value. The measure of scattering is the variance. In order to reduce the variance, Monte Carlo employ scoring of particle weights rather than the number and details are given in section 4.1. The random sampling methods depends on the type of data and it is described under neutron interaction data file format in section 4.2. For simplicity, energy-angle distribution is given in center of mass coordinate(CM) system for some reactions. The transformation of coordinates from CM system to Laboratory coordinate system is given towards the end of this chapter.

4.2. Particle Weight

In Monte Carlo simulations if the results (tallies) are scored based on particle count, the resulting variance in the mean value may be unacceptably high because of each outcome may be 'TRUE=1' or 'FLASE=0'. For instance, Monte Carlo simulation of number of particles transmitting a large thickness would result in the high value of variance. The variance can be reduced preserving the statistical mean value, by altering the transmission probability of the particle and reducing the weight of the particle accordingly and scoring the weight for the tally. The particle weight is unity if simulations are done directly without deviating natural processes. The Weight of a particle is an adjustment for deviating from a direct physical simulation of the transport process. The variance reduction methods use weighting schemes to produce the same mean as the natural transport process, but with lower calculational variance than the natural variance of the transport process. In short, particle weight is a number carried along with each particle, representing that particle's relative contribution to the final tallies. Its magnitude is determined to ensure that whenever Monte Carlo deviates from an exact simulation of the physics, the expected physical result nonetheless is preserved in the sense of statistical averages, and therefore in the limit of large particle numbers. For example, if an event is made 2 times as likely to occur (as it would occur without biasing), the tally ought to be multiplied by ½ so that the expected average tally is unaffected. In order to reduce variance associated with the mean value of Monte Carlo simulation, concept of particle weight is introduced such that the outcomes are somewhat evenly distributed.

4.3. Neutron Interaction Data File Format

Sampling theory is the key of all Monte Carlo codes. It samples a variable of a non-uniform distribution function from uniformly distributed function in the interval (0,1). Before applying a sampling theory, one must make sure that the given function is always positive and it is bounded in the interval of sampling interest. In the neutron transport theory, the outgoing energy and scattered angle of a neutron are sampled from the neutron-nuclear interactions. Neutron undergoes different types of interactions with nuclei while traversing the medium and are dependent on incident neutron energy and the type of target nuclide. The important interactions are

(a) elastic,
(b) inelastic,
(c) fission,
(d) capture or absorption, and

(e) other particle emission reactions.

Since these interaction cross sections are not smooth functions of energy but non-negative, they are tabulated in close intervals of energy. Such data files deduced from basic ENDF/B (Evaluated Nuclear Data File) files using 'NJOY' code[1] in to a suitable format. One such format is A Compact ENDF file format or generally known as ACE formatted data files. In addition, these files contain secondary neutron energy-angular distributions and ENDF laws of interpolation formulas. Monte Carlo codes make use of these data files for sampling of interaction events randomly using the pseudo random numbers. All programs read these and build the required probability and cumulative distribution tables before sampling. A typical edited cross section output file depicting reaction type and outgoing energy-angle distribution laws for u-235 isotope is given table 4-1.

Table 4-1: outgoing Energy-angle distribution ENDF LAWS for u-235 isotope (Edited)

sno	reaction	MT	Law	mult	Nei(angd)	Q-val	Thresh_E
2	(n,2n)	16	44	2	0	-5.298E+00	5.321E+00
3	(n,3n)	17	44	3	0	-1.214E+01	1.219E+01
4	(n,fission)	18	4	19	0	1.934E+02	1.000E-11
5	(n,4n)	37	44	4	0	-1.789E+01	1.796E+01
6	(n,n1)	51	3	1	24	-7.680E-05	2.250E-03
7	(n,n2)	52	3	1	21	-1.304E-02	1.310E-02
		-----	do	----			
45	(n,n40)	90	3	1	10	-3.909E+00	3.926E+00
46	**(n,nc)**	**91**	**44**	**1**	**0**	**-4.979E-01**	**5.000E-01**

Few points must be noted while understanding the columns of the table 3-1. Serial numbers 8 to 44 represent same inelastic scattering only with different Q-values. The column 2 (MT) refers ENDF Reaction Number and column 3 (LAW) refers to secondary neutron energy distribution law and column 5, (mult) refers to multiplicity of neutrons exiting from the reaction and for fission it is given as 19, which conveys that the code has to calculate the number neutrons emitted from the fission reaction. Further, energy distribution law 44 refers to outgoing energy-angle correlated Kallbach-Mann formalism and law 3 refers to discrete formula. For angular distribution, the tabular data is given in the center of mass(CM) coordinate system. After arriving at a proper value, it is necessary to convert to target at rest or laboratory (LAB) coordinate system.

4.4. Sampling Methods

The outgoing neutron energy-angle distributions are correlated or uncorrelated depending on the type of reaction. For elastic scattering the outgoing neutron energy and angle are correlated, only the angular distributions is provided in CM system at selected points of input energies. After sampling the angle, the resulting neutron energy is computed in the Lab system. For inelastic scattering, the angular distribution is isotropic in CM system and hence only outgoing neutron energies are sampled. The type of random sampling technique applicable depends on the form of input data available and they can be broadly classified as follows:

i. Direct Inversion Method
ii. Table Lookup Method (discrete pdf method)
iii. Rejection Method
iv. Composition Method, and
v. Composition-Rejection Method

4. Physics and Sampling Theory

Some physical processes can be described in a simple functional form. For instance, angular distribution of scattered neutron in CM system is an isotropic (constant function) at low incident neutron energies. These functions are integrable and directly invertible. A schematic representation of probability density function f(x) (PDF), cumulative probability function F(x) (CDF) and selection process ($x \leftarrow F^{-1}(\xi)$) is shown in Fig.4-1.

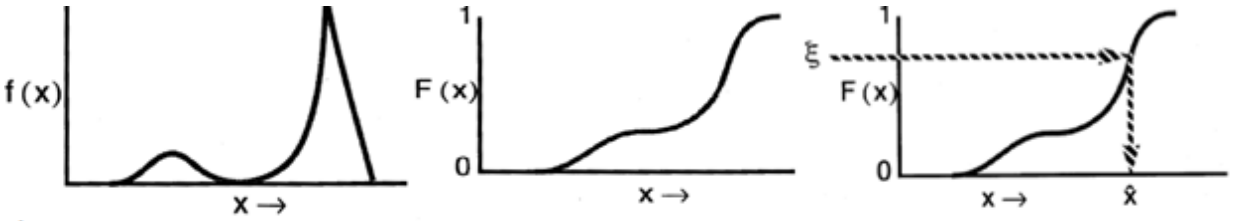


Fig.4.1. (a) PDF function (b) CDF function (c) Direct inversion

Some Commonly encountered functions in neutron transport Monte Carlo codes are given in table 4-2. The theory behind this can be found from reference[2].

Table 4-2: Directly invertible fumnctions[2] in Neutron Transport Monte Carlo Codes

Probability Density Function	Direct Sampling Method
Linear: f(x)=2x, 0 < x < 1	$X \leftarrow \sqrt{\xi}$
Exponential: $f(x)=e^{-x}$, 0 < x	$X \leftarrow -\log\xi$
2D isotropic: $f(\rho) = \frac{1}{2\pi}$, ρ(u,v)	$U \leftarrow \cos 2\pi\xi_1$ $v \leftarrow \sin 2\pi\xi_1$
3D isotropic: $f(\Omega) = \frac{1}{4\pi}, \quad \Omega = (u,v,w)$	$U \leftarrow 2\xi_1 - 1$ $V \leftarrow \sqrt{1-u^2}\ cos2\pi\xi_2$ $w\ \sqrt{1-u^2}\ sin2\pi\xi_2$
Maxwellian: $f(x) = \frac{2}{T\sqrt{\pi}}\sqrt{\frac{x}{T}}\ e^{\frac{-x}{T}}$ 0< x	$X \leftarrow T(\log\xi_1 - \log\xi_2 \cos^2\frac{\pi\xi 3}{2})$
Watt Spectrum: $f(x) = \frac{2e^{-ab/4}}{\sqrt{\pi a^3 b}}\ e^{-x/a} sinh\sqrt{bx}$, 0 <x	$W \leftarrow a(\log\xi_1 - \log\xi_2 \cos^2\frac{\pi\xi 3}{2})$ $X \leftarrow w + \frac{a^2 b}{4} + (2\xi_4 - 1)\sqrt{a^2 bw}$
Normal: $f(x) = \frac{1}{\sigma\sqrt{2\pi}}\ e^{\frac{-1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$ 0 < x	$X \leftarrow \mu + \sigma\sqrt{-2log\xi_1}\cos 2\pi\xi_2$

Other sampling schemes used in physics of random walk are presented or referenced at appropriate places in this chapter.

4.5. Sampling Distance to Next Collision (Direct Inversion method)

In Monte Carlo simulation of particle history, at first the distance to next collision is sampled from the source(collision) point. Some times this distance will be referred as a track length or a free flight distance between two collisions of a particle. Within a given cell of fixed composition, the method of sampling a collision along the track is determined using the following theory. The probability of a collision for a particle between *l* and *l* + *dl* along its line of flight is given by

$$p(l)dl = e^{-\Sigma_t l}\Sigma_t dl \qquad (4.1)$$

where Σ_t is the macroscopic total cross section of the medium and is interpreted as the probability per unit length of a collision. Setting ξ the random number on [0,1), to be

$$\xi = \int_0^l e^{-\Sigma_t s}\Sigma_t ds = 1\text{-}e^{-\Sigma_t l} \quad (4.2)$$

$$l = -\frac{1}{\Sigma_t}\ln(1-\xi) \quad (4.3)$$

But, because 1 – ξ is distributed in the same manner as ξ and hence may be replaced by ξ, we obtain the well-known expression for the distance to collision

$$l = -\frac{\ln(\xi)}{\Sigma_t} \quad (4.4)$$

Track length tallies use the length of a track in a given cell to determine a quantity of interest, such as fluence, flux, energy deposition, or multiplication factor (k^{tra} track length estimator) in criticality calculations. After transporting the estimated distance, particle undergoes collision and it used in criticality calculation to estimate multiplication factor of the system (k^{col} collision estimators).

4.6. Sampling of Neutron collisios

After transporting the neutron to the estimated distance and if found in the same material, neutron undergoes collision with a nuclide. The following sequence is followed:

1. *the collision nuclide is sampled (Discrete PDF method);*
2. *either the S(α, β) treatment is used or the velocity of the target nucleus is sampled for low energy neutrons (~below about 4 eV);*
3. *neutron capture (that is, neutron disappearance by any process) is modeled;*
4. *unless the S(α, β) treatment is used, either elastic scattering or an inelastic reaction (including fission) is selected, and the new energy and direction of the outgoing track(s) are determined;*
5. *if the energy of the neutron is low enough and an appropriate S(α, β) table is present, the collision is modeled by the S(α, β) treatment instead of by step 4.*

In criticality calculations, collision events are used for estimation of neutron multiplication factor and is known as collision estimator (k^{col}). Details of collision sampling steps are given below.

4.6.1. Sampling of Collision Nuclide (Discrete pdf method)

The schematic drawing of selecting a nuclide which have discrete probabilities in a material with composition of nuclides $x_1, x_2, x_3 \ldots$ is shown in Fig. 4.2(a,b,c).

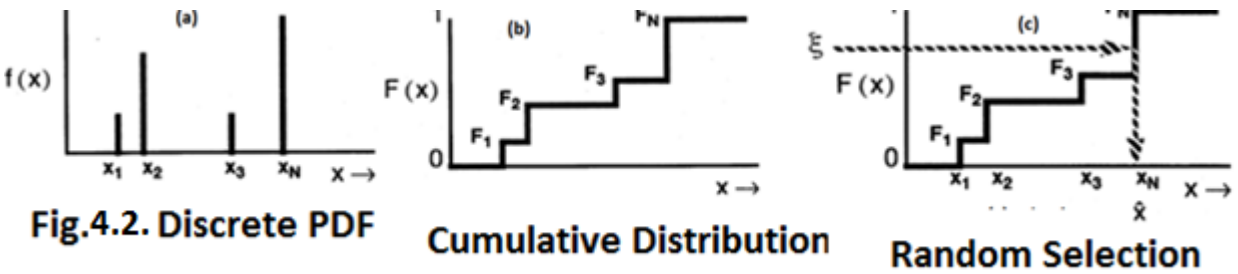


Fig.4.2. Discrete PDF Cumulative Distribution Random Selection

In Monte Carlo codes, sampling of a colliding nuclide out of *n* different nuclides forming the material is done in the following way. The total macroscopic cross section (Σ) at the incoming neutron energy for all nuclides is calculated by the suitable interpolation and a random number ξ on the unit interval [0,1) is generated. Then the k^{th} nuclide is chosen based on the equation (3.5)

$$\sum_{l=1}^{k-1} \Sigma_{tl} < \xi \sum_{l=1}^{n} \Sigma_{tl} < \sum_{l=1}^{k} \Sigma_{tl} \quad (4.5)$$

Where $\sum_{tl}$ is the macroscopic total cross section of a nuclide *l*. If the energy of the neutron is low enough (below about 4 eV) and if the appropriate S(α, β) table is present, the total cross section is the sum of the capture cross section from the regular cross-section table and the elastic and inelastic scattering cross sections from the S(α, β) table. Otherwise, the total cross section is taken from the regular cross-section table and is adjusted for thermal effects as described below.

4.6.2. sampling of Absorption or Disappearance reaction(Table lookup method)

After sampling the target nuclide in the material, next it is looked for neutron disappearance reaction, since it is easy to implement. The absorption cross section is specially defined as the sum of all (*n*, *x*) cross sections, where *x* is anything except neutrons. Thus σ_a is the sum of $\sigma_{n,g}$, $\sigma_{n,a}$, $\sigma_{n,d}$, σ_f, etc. For fissile nuclides, "absorption" includes both capture and fission reactions. This is done by sampling a random number ξ on the interval [0, 1) and checking whether

$$\xi\sigma_t(E) < \sigma_a(E) - \sigma_f(E) \quad (4.6)$$

where σ_t is the total cross section, σ_a is the absorption cross section (this includes fission), and σ_f is the total fission cross section of the nuclide. Absorption is treated in one of two ways: analog or implicit.

Analog Absorption: In analog absorption, the particle is killed with probability σ_a/σ_t. where σ_a and σ_t are the absorption and total cross sections of the collision nuclide at the incoming neutron energy. For all particles killed by analog absorption, the entire particle energy multiplied with weight is deposited in the collision cell. This particle is killed and the simulation is carried out by drawing a new particle from the bank.

Implicit Absorption: In this case particle is not killed in the absorption process instead the weight of the particle is changed. For implicit absorption, also called survival biasing, the neutron weight *Wn* is reduced to W'*n* as follows:

$$W_n' = \left(1 - \frac{\sigma_a}{\sigma_t}\right) * W_n \quad (4.7)$$

If the new weight W*n* is below the problem weight cutoff (default = 0.25), Russian roulette is played, resulting overall in fewer particles with larger weight. This is important for deep penetration shielding problems.

4.6.3. Sampling of Elastic Scattering

If the particle does not undergo disappearance reaction, then it may undergo scattering. There are two types of scattering phenomena viz. elastic and inelastic of neutron with nucleus. In elastic scattering neutron does not loose energy in the center-of-mass system and the target nucleus will not

be left in an excited state. However, in the lab coordinates, the neutron does indeed lose energy. The selection of an elastic collision is made with the probability

$$\frac{\sigma_{el}}{\sigma_{el}+\sigma_{in}} = \frac{\sigma_{el}}{\sigma_t - \sigma_a} \qquad (4.8)$$

where σ_{el} is the elastic scattering cross section, σ_{in} is the inelastic cross section; includes any neutron-out processes–(*n,n'*), (*n,f*), (*n,np*), etc., σ_a is the absorption cross section, that is, all neutron disappearing reactions. σ_t is the total cross section comprises of, $\sigma_t = \sigma_{el} + \sigma_{in} + \sigma_a$. Both σ_{el} and σ_t are adjusted for the free gas thermal treatment at thermal energies (< 4 eV, velocity of target nuclide has to be sampled). Directions and energies of all outgoing particles from neutron collisions are determined by sampling data from the appropriate cross-section table.

4.6.4. Sampling of Inelastic Scattering (Table lookup method)

Number of inelastic reactions are possible between the neutron and nuclide (ENDF MT=50 to 91). The selection of an inelastic collision is made with the remaining scattering probability

$$\frac{\sigma_{in}}{\sigma_t - \sigma_a} \qquad (4.9)$$

If the collision is determined to be inelastic, the type of inelastic reaction, *n*, is sampled from

$$\sum_{i=1}^{n-1}\sigma_i < \xi \sum_{i=1}^{N}\sigma_i < \sum_{i=1}^{n}\sigma_i \qquad (4.10)$$

where ξ is a random number on the interval [0,1), *N* is the number of inelastic reactions, and σ_i is the i^{th} inelastic reaction cross section at the incident neutron energy. If more than one neutron results in the exit channel, the neutron weight is multiplied with the number of neutrons exiting the reaction. Directions and energies of all outgoing particles from neutron collisions are determined by sampling data from the appropriate cross-section table.

4.6.5. Sampling of Fission

While fission is normally considered an absorption reaction, as far as it concerned in Monte Carlo simulation. It actually bears more similarities to inelastic scattering since fission results in secondary neutrons in the exit channel. Other absorption reactions like (n, γ) or (n, α), on the contrary, produce no neutrons. In an eigenvalue calculation, secondary neutrons from fission are only "banked" for use in the next generation rather than being tracked as secondary neutrons from elastic and inelastic scattering would be. On top of this, fission is sometimes broken into first-chance fission, second-chance fission, etc. The nuclear data file either lists the partial fission reactions with secondary energy distributions for each one, or a total fission reaction with a single secondary energy distribution.

When a fission reaction is sampled, the following algorithm is used to create and store fission sites for the following generation. First, the average number of prompt and delayed neutrons must be determined to decide whether the secondary neutrons will be prompt or delayed. This is important because delayed neutrons have a markedly different energy spectrum from prompt neutrons, one that has a lower average energy of emission. The total number of neutrons emitted (ν_t) or prompt neutrons emitted (ν_p) and delayed neutrons (ν_d) are given as a function of incident energy in the ENDF cross section file. Two representations exist for ν_t. The first is a polynomial of order N with coefficients $c_0, c_1, \ldots, c_N$. If ν_t has this form, we can evaluate it at incoming energy E by using the equation

$$\nu_t(E) = \sum_{i=0}^{N} C_i E^i \qquad (4.11)$$

The other representation is just a tabulated function with a specified interpolation law. Once these (ν_t and ν_d) have been determined, we can calculated the delayed neutron fraction

$$\beta = \frac{\nu_t}{\nu_d} \qquad (4.12)$$

We then need to determine how many total neutrons should be emitted from fission. If no survival biasing is being used, then the number of neutrons emitted is

$$\nu = \frac{W\nu_t}{k_{eff}} \quad (4.13)$$

where w is the statistical weight and k_{eff} is the effective multiplication factor from the previous generation. The number of neutrons produced is biased in this manner so that the expected number of fission neutrons produced is the number of source particles that we started with in the generation. Since ν is not an integer, we use the following procedure to obtain an integral number of fission neutrons to produce. If $\xi > \nu - \lfloor\nu\rfloor$, then we produce $\lfloor\nu\rfloor$ neutrons. Otherwise, we produce $\lfloor\nu\rfloor$+1 neutrons. Then, for each fission site produced, we sample the outgoing angle and energy according to the algorithms given in Sampling Angular Distributions and Sampling Energy Distributions respectively.

.

If the neutron is to be born delayed, then there is an extra step of sampling a delayed neutron precursor group since they each have an associated secondary energy distribution. The sampled outgoing angle and energy of fission neutrons along with the position of the collision site are stored in an array called the fission bank. In a subsequent generation, these fission bank sites are used as starting source sites.

4.7. Sampling of Secondary neutron Angle-Energy Distributions

For a reaction with secondary products (neutron only in criticality calculations), it is necessary to determine the outgoing angle and energy of the products, in particular neutron. For any reaction other than elastic and level inelastic scattering, the outgoing energy must be determined based on tabulated or parameterized data. The ACE format specifies its own representations based loosely on the formats given in ENDF-6 as (i) the outgoing energy and outgoing angle are sampled independently of each other (uncorrelated) , or (ii) the outgoing energy and outgoing angle are correlated. In the latter case, the outgoing energy may be specified as a function of the sampled outgoing angle, or the outgoing angle may be specified as a function of the sampled outgoing energy. Details of the possible data representations and sampling schemes are provided in the following sections. We will describe how the outgoing angle and energy of secondary particles are sampled.

One of the subtleties in the nuclear data format is the fact that a single reaction product can have multiple angle-energy distributions. This is mainly useful for reactions with multiple products of the same type in the exit channel such as $(n, 2n)$ or $(n, 3n)$. In these types of reactions, each neutron is emitted corresponding to a different excitation level of the compound nucleus, and thus in general the neutrons will originate from different energy distributions. If multiple angle-energy distributions are present, they are assigned incoming-energy-dependent probabilities that can then be used to randomly select one. Once a distribution has been selected, the procedure for determining the outgoing angle and energy will depend on the type of the distribution.

4.7.1. Sampling Angular Distributions

The cosine of the angle between incident and exiting particle directions, is sampled from angular distribution tables in the collision nuclide's cross-section library. The cosines are either in the center-of-mass or target-at-rest system, depending on the type of reaction. Data are provided at a number of incident neutron energies. Two types of angular distributions are:

- An isotropic angular distribution,
- A tabular distribution

Sampling Isotropic Angular Distribution

Secondary neutron emission is isotropic for fission and inelastic scattering interactions. In this case, no data is present in the nuclear data file, and the cosine of the scattering angle is simply calculated as

$$\mu = 2\xi - 1 \quad (4.14)$$

where μ is the cosine of the scattering angle and ξ is a random number sampled uniformly on [0, 1)

Sampling of non-isotropic Angular Distribution (Table Lookup method)

There are two ways of representing and sampling a non-isotropic scattering cosine in a tabular form. The first method involves the use of 32 equally probable cosine bins. When this method with 32 equiprobable cosine bins is employed, a random number on the interval [0,1) is used to select the i^{th} cosine bin. The value of μ is interpolated in the intervals between i and i+1 using same random number ξ, then

$$i = int(32 * \xi)$$
$$\mu = \mu_i + (32\,\xi - i)(\mu_{i+1} - \mu_i) \quad (4.15)$$

The method of 32 equiprobable cosine bins accurately represents high-probability regions of the scattering probability; however, it can be a very crude approximation in low-probability regions.

In the second method, the angular distribution data tables (cosines of angle, probabilities, cumulative probabilities) are provided at discrete values for a number of incident neutron energies. For each incoming neutron energy E_i, let us call $p_{i,j}$ the j[th] value in the probability distribution function and $c_{i,j}$ the j[th] value in the cumulative distribution function. We first find the interpolation factor f on the incoming energy grid:

$$f = \frac{E - E_i}{E_{i+1} - E_i} \quad (4.16)$$

where E is the incoming energy of the particle. Then, statistical interpolation is performed to choose between energy E_i and E_{i+1}. Let ℓ be the chosen table where $\ell = i$, if $\xi_1 > f$ and $\ell = i + 1$ otherwise, where ξ_1 is a random number. Another random number ξ_2 is used to sample a scattering cosine bin j using the cumulative distribution function:

$$c_{l,j} < \xi_2 < c_{l,j+1} \quad (4.17)$$

The final scattering cosine will depend on whether histogram or linear-linear interpolation is recommended in the ACE table. In general, we can write the cumulative distribution function as

$$c(\mu) = \int_{-1}^{\mu} p(\mu')\, d\mu' \quad (4.18)$$

where $c(\mu)$ is the cumulative distribution function and $p(\mu)$ is the probability distribution function. Since tabulated values are at discrete points, denoting cumulative probability till $\mu_{\ell,j,} = c_{\ell,}$j, and if $\mu > \mu_{\ell,j}$, equation (3.20) can be written as

$$\mathrm{c}(\mu) = c_{l,j} + \int_{\mu_{l,j}}^{\mu} p(\mu')\, d\mu' \quad (4.19)$$

For histogram interpolation, we have that $p(\mu') = p_{\ell}$, for $\mu_{\ell,j} \leq \mu' < \mu_{\ell,j+1}$. Thus, after integrating, we have that

$$\mathrm{c}(\mu) = c_{l,j} + (\mu - \mu_{l,j}) p_{l,j} = \xi_2$$

$$\mu = \mu_{l,j} + \frac{\xi_2 - c_{l,j}}{p_{l,j}} \quad (4.20)$$

For linear-linear interpolation of probability distribution function (p), and if we interpolate between successive values on the probability distribution function, then

$$\mathrm{p}(\mu') - p_{l,j} = \frac{p_{l,j+1} - p_{l,j}}{\mu_{l,j+1} - \mu_{l,j}} (\mu' - \mu_{l,j}) \quad (4.21)$$

Inserting for $p(\mu')$ from the above, into equation (4.21), we obtain

$$\mathrm{c}(\mu) = c_{l,j} + \int_{\mu_{l,j}}^{\mu} \left[\frac{p_{l,j+1} - p_{l,j}}{\mu_{l,j+1} - \mu_{l,j}} (\mu' - \mu_{l,j}) + p_{l,j} \right] d\mu' \quad (4.22)$$

Let us now make a change of variables using

$$\eta = \frac{p_{l,j+1} - p_{l,j}}{\mu_{l,j+1} - \mu_{l,j}} (\mu' - \mu_{l,j}) + p_{l,j}$$

$$\mathrm{c}(\mu) = c_{l,j} + \frac{1}{m} \int_{\mu_{l,j}}^{m(\mu - \mu_{l,j}) + p_{l,j}} \eta \; d\eta \quad (4.23)$$

where we have used

$$\mathrm{m} = \frac{p_{l,j+1} - p_{l,j}}{\mu_{l,j+1} - \mu_{l,j}}$$

Integrating the equation (4.23) and equating to random number ξ_2, we have

$$\mathrm{c}(\mu) = c_{l,j} + \frac{1}{2m} \left(\left[m(\mu - \mu_{l,j}) + p_{l,j} \right]^2 + p_{l,j}^2 \right) = \xi_2$$

Solving for μ, we have the final form for the scattering cosine using linear-linear interpolation:

$$\mu = \mu_{l,j} + \frac{1}{m} \left(\sqrt{p_{l,j}^2 + 2m(\xi_2 - c_{l,j})} - p_{l,j} \right) \quad (4.24)$$

After sampling cosine of the angle, the scattered neutron energy is then calculated from two-body kinematics, if it is correlated.

4.7.2. Sampling secondary neutron Energy Distributions

The outgoing neutron energy (E') distribution data given either in the parametric form, continuous tabular data or coupled angle-energy distribution form. Sampling of E' depends on the type of data available for reactions (elastic, inelastic, fission, other particulate reactions) which contain neutrons in the exit channels

4.7.2.1. Outgoing neutron Energy from Elastic Scattering:

For elastic scattering angular distribution data in CM system is given in ACE files. Elastic scattering is nearly isotropic for low energy incident neutron in the center of mass system and no need to give data. However, for higher energies, elastic scattering is anisotropic and generally given in the form of continuous tabular data in the CM system.

Once the particle direction is sampled from the appropriate angular distribution tables, then the exiting energy,E', is dictated by two-body kinematics.

$$E' = \frac{1}{2}E[(1-\alpha)\mu_{cm} + 1 + \alpha]$$
$$= E\left[\frac{1 + A^2 + 2A\mu_{cm}}{(1+A)^2}\right]$$

Where E = incident neutron energy, μ_{cm} = center-of-mass cosine of the angle between incident and exiting particle directions,

$$\alpha = \left[\frac{(A-1)^2}{(A+1)^2}\right]$$

and A = mass of collision nuclide in units of the mass of a neutron (atomic weight ratio),The μ_{cm} is converted to lab coordinate system and the resulting direction cosines calculations are given in the conversion of CM to lab system.

4.7.2.2. Outgoing neutron Energy from Inelastic Scattering and fission

Inelastic scattering is discrete level scattering, firstly level is sampled and Q-value, the energy distribution law is fetched from the data table and the outgoing neutron energy is calculated based on the particular ENDF law. Generally, the angular distribution is isotropic in CM system. Sampling of Important ENDF energy distribution laws given in the table are explained below.

i) Law=3 Inelastic Level Scattering

For inelastic level scattering secondary neutron angular distribution is isotropic in CM system, and the energy distribution is given mostly in the parametric form in the CM system (ENDF Law-3) and calculated as

$$E_{out}^{CM} = \left(\frac{A}{A+1}\right)^2 * \left(E - \left(\frac{A+1}{A}\right)Q\right) \quad (4.25)$$

Where E_{out}^{CM} = outgoing energy in the CM system
E = incident neutron energy
A = atomic weight ratio
Q = Q-Value

The outgoing energy in the laboratory system E_{out}^{lab} computed as

$$E_{out}^{lab} = E_{out}^{CM} + \frac{\left\{ E+2\mu_{cm}(A+1)\sqrt{EE_{out}^{CM}} \right\}}{(A+1)^2} \quad (4.26)$$

The cosine of the angle μ_{cm} is sampled from isotropic distribution. The μ_{lab} is computed to lab system using appropriate way explained elsewhere. The form of ACE data if not parametric, it is given as continuous tabular data either correlated with angle or independent.

ii) Law=4 Continuous Tabular Distribution-scaled Energy interpolation

In ACE formatted data files, a continuous tabulated secondary neutron energy distribution table is provided for inelastic and particle emission reactions for a set of incoming energies. If one performs simple interpolation between tables for neighboring incident energies, it is possible that the resulting energies would violate laws governing the kinematics, i.e., the outgoing energy may be outside the range of available energy in the reaction. To avoid this situation, the accepted practice is to use a process known as scaled interpolation. First, we find the tabulated incident energies which bound the actual incoming energy of the particle, i.e., find i such that $Ei < E < E_{i+1}$ and calculate the interpolation factor f

$$f = \frac{E - E_i}{E_{i+1} - E_i} \quad (4.27)$$

Then, we interpolate between the minimum and maximum energies of the outgoing energy distributions corresponding to E_i and E_{i+1}.

$$E_{min} = E_{i,1} + f(E_{i+1,1} - E_{i,1}) \quad (4.28)$$
$$E_{max} = E_{i,M} + f\left(E_{i+1,M} - E_{i,M}\right) \quad (4.29)$$

where E_{min} and E_{max} are the minimum and maximum outgoing energies of a scaled distribution for incident neutron energy bin i. Let ℓ be the chosen with $\ell = i$ if $\xi_1 > f$ and $\ell = i + 1$ otherwise, and ξ_1 is a random number. For each incoming neutron energy E_i, let us call $p_{i,j}$ the jth value in the probability distribution function, $c_{i,j}$ the jth value in the cumulative distribution function, and $E_{i,j}$ the jth outgoing energy. We then sample an outgoing energy bin j using the cumulative distribution function:

$$c_{l,j} < \xi_2 < c_{l,j+1} \quad (4.30)$$

where ξ_2 is a random number sampled uniformly on [0, 1). At this point, we need to interpolate between the successive values on the outgoing energy distribution using either histogram or linear-linear interpolation. The formulas for these can be derived along the same lines as those found in Tabular Angular Distribution (see section 3.3.1). For histogram interpolation, the interpolated outgoing energy on the ℓ^{th} energy bin is

$$E^{\wedge} = E_{l,j} + \frac{\xi_2 - c_{l,j}}{p_{l,j}} \quad (4.31)$$

If linear-linear interpolation is to be used in PDFs, the outgoing energy on the ℓ^{th} energy bin is

$$E^{\wedge} = E_{l,j} + \frac{E_{l,j+1} - E_{jl}}{p_{l,j+1} - p_{l,j}} \left(\sqrt{p_{l,j}^2 + 2\frac{p_{l,j+1} - p_{l,j}(\xi_2 - c_{l,j})}{E_{l,j+1} - E_{l,j}}} - p_{l,j} \right) \quad (4.32)$$

Since this outgoing energy may violate reaction kinematics, we then scale it to minimum and maximum energies calculated in equations (4.28 and 4.29) to get the final outgoing energy:

$$E' = E_{min} + \frac{E^{\wedge} - E_{l,1}}{E_{l.M} - E_{l,1}} (E_{max} - E_{min}) \quad (4.33)$$

where E_{min} and E_{max} are defined the same as in equations (4.28 and 4.29) for corresponding input energy l.

iii) Law=7 Maxwell Fission Spectrum (composition rejection method)

One representation of the secondary energies for neutrons from fission is the so-called Maxwell spectrum. A probability distribution for the Maxwell spectrum can be written in the form

$$p(E')dE' = c\sqrt{E'}\, e^{-\frac{E'}{T(E)}}\, dE' \quad (4.34)$$

where E is the incoming energy of the neutron and $T(\mathrm{E})$ is the so-called nuclear temperature, which is a function of the incoming energy of the neutron. The ENDF format contains a list of nuclear temperatures versus incoming energies. The nuclear temperature is interpolated between neighboring incoming energies using a specified interpolation law. Once the temperature T is determined, we then calculate a candidate outgoing energy based on rule C64 in the Monte Carlo Sampler[2]:

$$E' = -T\{log^{(\xi_1)} + log^{(\xi_2)} cos^2\left(\frac{\pi\xi_3}{2}\right)\} \quad (4.35)$$

where $\xi 1$, $\xi 2$, $\xi 3$ are random numbers sampled on the unit interval. The outgoing energy is only accepted if

$$0 \leq \mathrm{E}' \leq E - U \quad (4.36)$$

else rejected, and it is resampled using equation (4.35). Here U is restricted energy from the ACE table.

iv) Law=7 Energy-Dependent Watt Spectrum(composition rejection method)

The probability distribution for a Watt fission spectrum can be written in the form

$$p(E')dE' = c\, e^{-\frac{E'}{a(E)}}\, sinh\sqrt{b(E)E'}\; dE' \quad (4.36)$$

where a and b are parameters for the distribution and are given as tabulated functions of the incoming energy of the neutron. These two parameters are interpolated on the incoming energy grid using a specified interpolation law. Once the parameters have been determined, we sample a Maxwellian

spectrum with nuclear temperature 'a' using the algorithm described in Maxwell Fission Spectrum to get an energy W. Then, the outgoing energy is calculated as

$$E' = W + \frac{a^2 b}{4} 2(\xi - 1)\sqrt{a^2 b W} \qquad (4.37)$$

where ξ is a random number sampled on the interval [0, 1). The outgoing energy is only accepted according to a specified restriction energy U as defined $0 < E' < E - U$. The derivation of the algorithm described here can be found in a paper by Romano.[4]

v) Law=44 Kalbach-Mann Correlated Scattering

If the secondary distribution for a product was given with this law, the angle and energy are correlated with one another and cannot be sampled separately. Several representations exist in ENDF/ACE for correlated angle-energy distributions. This law is very similar to the uncorrelated continuous tabular energy distribution except now the outgoing angle of the neutron is correlated to the outgoing energy and is not sampled from a separate distribution. For each incident neutron energy E_i tabulated, there is an array of precompound factors $R_{i,j}$ and angular distribution slopes $A_{i,j}$ corresponding to each outgoing energy bin j in addition to the outgoing energies and distribution functions as in Continuous Tabular Distribution.

The calculation of the outgoing energy of the neutron proceeds exactly the same as in the algorithm described in Continuous Tabular Distribution. In that algorithm, we found an interpolation factor f, statistically sampled an incoming energy bin ℓ, and sampled an outgoing energy bin j based on the tabulated cumulative distribution function. Once the outgoing energy has been determined with equation (4.33 scaled interpolation), we then need to calculate the outgoing angle based on the tabulated Kalbach-Mann parameters. These parameters themselves are subject to either histogram or linear-linear interpolation on the outgoing energy grid. For histogram interpolation, the parameters are

$$R = R_{i,j} \quad (4.38)$$
$$A = A_{i,j} \quad (4.39)$$

If linear-linear interpolation is specified, the parameters are

$$R = R_{i,j} + \frac{E^{\wedge} - E_{l,j}}{E_{l,j+1} - E_{l,j}}(R_{l,j+1} - R_{l,j}) \quad (4.40)$$
$$A = A_{i,j} + \frac{E^{\wedge} - E_{l,j}}{E_{l,j+1} - E_{l,j}}(A_{l,j+1} - A_{l,j}) \quad (4.41)$$

where E^ is defined in equation (4.31). With the parameters determined, the probability distribution function for the cosine of the scattering angle is

$$p(\mu)d\mu = \frac{A}{2\sinh(A)}[\cosh(A\mu) + R\sinh(A\mu)]d\mu \quad (4.42)$$

The rules for sampling this probability distribution function can be derived based on rules C39 and C40 in the Monte Carlo Sampler[2]. First, we sample two random numbers ξ_3, ξ_4 on the unit interval. If $\xi_3 > R$ then the outgoing angle is

$$\mu = \frac{1}{A}\ln(T + \sqrt{T^2 + 1}) \quad (4.42)$$

where $T = (2\xi_4 - 1)\sinh(A)$. If $\xi_3 \leq R$, then the outgoing angle is

$$\mu = \frac{1}{A}\ln\left(\xi_4 e^A + (1-\xi_4)e^{-A}\right) \quad (4.43)$$

vi) LAW=66 N-Body Phase Space Distribution

Reactions in which there are more than two products of similar masses are sometimes best treated by using what's known as an N-body phase distribution. This distribution has the following probability density function for outgoing energy and angle of the i-th particle in the center-of-mass system:-

$$p_i(\mu, E')dE'd\mu = c_n\sqrt{E'}\,(E_i^{max} - E')^{\frac{3n}{2}-4}\,\mathrm{dE'd\mu} \quad (4.44)$$

where n is the number of outgoing particles, C_n is a normalization constant, E_i^{max} is the maximum center-of-mass energy for particle i, and E' is the outgoing energy. We see in equation (4.44) that the angle is simply isotropic in the center-of-mass system. The algorithm for sampling the outgoing energy is based on algorithms R28, C45, and C64 in the Monte Carlo Sampler. First we calculate the maximum energy in the center-of-mass using the following equation:

$$E_i^{max} = \frac{A_p - 1}{A_p}\left(\frac{A}{A+1}E_{in} + Q\right) \quad (4.45)$$

where A_p is the total mass of the outgoing particles in neutron masses, A is the mass of the original target nucleus in neutron masses, and Q is the Q-value of the reaction. Next we sample a value x from a Maxwell distribution with a nuclear temperature of one using the algorithm outlined in Maxwell Fission Spectrum. We then need to determine a value y that will depend on how many outgoing particles there are. For n = 3, we simply sample another Maxwell distribution with unity nuclear temperature. For n = 4, we use the equation

$$y = -\ln(\xi_1\xi_2\xi_3) \quad (4.46)$$

where ξ_i are random numbers sampled on the interval [0, 1). For n = 5, we use the equation

$$y = -\ln(\xi_1\xi_2\xi_3\xi_4) - \ln(\xi_5)\cos^2\left(\frac{\pi}{2}\xi_6\right) \quad (4.47)$$

After x and y have been determined, the outgoing energy is then calculated as

$$E' = \frac{x}{x+y}E_i^{max} \quad (4.48)$$

In the ENDF/B-VII.1 nuclear data evaluation, only one reaction uses an N-body phase space distribution at all, the $(n, 2n)$ reaction with H-2.

vi) Law=9 Evaporation Spectrum

Evaporation spectra are primarily used in compound nucleus processes where a secondary particle can "evaporate" from the compound nucleus if it has sufficient energy. The probability distribution for an evaporation spectrum can be written in the form

$$p(E')dE' = c\,E'\,e^{-\frac{E'}{T(E)}}\,dE' \quad (4.49)$$

where E is the incoming energy of the neutron and T(E) is the nuclear temperature, which is a function of the incoming energy of the neutron. The ENDF format contains a list of nuclear temperatures versus incoming energies. The nuclear temperature is interpolated between neighboring incoming energies using a specified interpolation law. Once the temperature T is determined, we then calculate a candidate outgoing energy based on the algorithm given in LA-UR-14-27694:

$$E' = -T \log ((1 - g\xi_1)(1 - g\xi_2)) \quad (4.50)$$

where $g = 1 - e^{-w}$, $w = (E - U)/T$, U is the restriction energy, and ξ_1, ξ_2 are random numbers sampled on the unit interval. The outgoing energy is only accepted according to the restriction energy $0 < E' < E-U$.

vii) Prompt and Delayed Neutron Emission:

If the ACE data for the collision isotope includes delayed-neutron spectra, and then each fission neutron is first determined by sampling, the type of emitted neutron is delayed or prompt is determined from the ratio of delayed $\nu_d(Ein)$ to total $\nu_t(Ein)$ as

if $\xi < \nu_d(Ein) / \nu_t(Ein)$, produce a delayed neutron, or
if $\xi > \nu_d(Ein) / \nu_t(Ein)$, produce a prompt neutron,

where ν_d is the expected number of delayed neutrons. If the neutron is determined to be a prompt fission neutron, it is emitted instantaneously, and the emission laws (angle and energy) specified for prompt fission are sampled (watt fission spectrum). If the neutron is determined to be a delayed fission neutron, then sampling for the decay group by using the specified abundances. Finally, delayed neutron energy is obtained from the appropriate emission. In general, the delayed neutron spectra tend to be softer than the prompt spectra. Overall, inclusion of delayed-neutron spectra can be expected to produce small positive reactivity changes for systems with significant fast neutron leakage and small negative changes for some systems in which a significant fraction of the fissions occurs in isotopes with an effective fission threshold (e.g., 238U and 240Pu).

4.8. Free Gas Thermal Treatment

A collision between a neutron (at low energies ~eV) and an atom is affected by the thermal motion of the atom, and in most cases, the collision is also affected by the presence of other atoms nearby. The free gas thermal treatment consists of adjusting the elastic cross section and taking into account the velocity of the target nucleus when the kinematics of a collision are being calculated. No adjustment is made if the elastic cross section in the data library was already processed to the room temperature since MCkeff uses only room temperature cross sections. MCkeff uses explicit S(α,β) data set that takes into account the effects of chemical binding and crystal structure for incident neutron energies below about 4 eV for only a limited number of substances. Details of treatment S(α,β) tables is given below. The method by which most Monte Carlo codes sample the target velocity for use in elastic scattering kinematics is outlined in detail by Gelbard[5] and We adopt sampling method given in OpenMC[6] for the sampling of target speed(v_T). The following probability density function is used for sampling v_T for incident neutron speed v_n.

$$q(x)dx = \left[\left(\frac{\sqrt{\pi}y}{\sqrt{\pi}y + 2} \right) \frac{4}{\sqrt{\pi}} x^2 e^{-x^2} + \left(\frac{2}{\sqrt{\pi}y + 2} \right) 2x^3 e^{-x^2} \right] dx \quad (4.51)$$

Where

$$x = \beta dv_T \text{ and } y = \beta dv_n. \quad (4.52)$$

We sample a random number ξ_1 on the interval [0, 1) and if

$$\xi_1 < \frac{2}{\sqrt{\pi}y + 2}$$

then we sample the probability distribution $2x^3e^{-x^2}$ for x using rule C49 in the Monte Carlo Sampler[2] which we can then use to determine the speed of the target nucleus v_T from equation (4.52). Otherwise, we sample the probability distribution $\sqrt{4/\pi}\, x^2e^{-x^2}$ for x using rule C61 in the Monte Carlo Sampler. The cosine can be sampled isotropically as $\mu = 2\xi_2 - 1$ where ξ_2 is a random number on the unit interval. The sampled target speed and cosine are tested to validity range and then accepted. If is not accepted, then we repeat the process and resample a target speed and cosine until a combination is found.

4.9. The S(α,β) Treatment(H,C ~ 4 eV)

For neutrons with thermal energies, generally less than 4 eV, the kinematics of scattering can be affected by chemical binding (H_2O) and crystalline effects of the target molecule (graphite , beryllium). If these effects are not accounted for in a simulation, the reported results may be highly inaccurate. The ACE tables after taking in to account these effects contain the tabular data of

- Thermal inelastic scattering cross section;
- Thermal elastic scattering cross section;
- Correlated energy-angle distributions for thermal inelastic and elastic scattering.

These tables are mainly used for moderating materials such as light or heavy water, graphite, hydrogen in ZrH, beryllium, etc. For the inelastic treatment, the distribution of secondary energies is represented by a set of equally probable final energies (~64) for each member of a grid of initial energies(~180) from an upper limit of typically 4 eV down to 10^{-5} eV, along with a set of angular data (equviprobable μ's 15 in CM) for each initial and final energy. The selection of a final energy E' given an initial energy E can be characterized by sampling from the distribution

4.9.1. Outgoing Energy and Angle for Inelastic Scattering

Each S(α, β, T) table provides a correlated angle-energy secondary distribution for neutron thermal inelastic scattering. There are three representations used in the ACE thermal scattering data: equiprobable discrete outgoing energies, nonuniform yet still discrete outgoing energies, and continuous outgoing energies with corresponding probability and cumulative distribution functions provided in tabular format. These three representations all represent the angular distribution in a common format, using a series of discrete equiprobable outgoing cosines.

Equi-Probable Outgoing Energies

If the thermal data was processed with iwt = 1 in NJOY, then the outgoing energy spectra is represented in the ACE data as a set of discrete and equiprobable outgoing energies. The procedure to determine the outgoing energy and angle is as such. First, the interpolation factor(f) is determined from scanning input energies(bin i,i+1). Then, an outgoing energy bin is sampled from a uniform distribution (j) and then interpolated between values corresponding to neighboring incoming energies:

$$E = E_{i,j} + f(E_{i+1,j} - E_{i,j}) \quad (3.48)$$

where $E_{i,j}$ is the j-th outgoing energy corresponding to the i-th incoming energy. For each combination of incoming and outgoing energies, there is a series equiprobable outgoing cosines. An outgoing cosine bin (k) is sampled uniformly and then the final cosine is interpolated on the incoming energy grid:

$$\mu = \mu_{i,j,k} + f(\mu_{i+1,j,k} - \mu_{i,j,k}) \quad (3.49)$$

where $\mu_{i,j,k}$ is the k^{th} outgoing cosine corresponding to the j^{th} outgoing energy and the i^{th} incoming energy.

Skewed Equi-Probable Outgoing Energies

If the thermal data was processed with iwt = 0 in NJOY, then the outgoing energy spectra is represented in the ACE data according to the following: the first and last outgoing energies have a relative probability of 1, the second and second-to-last energies have a relative probability of 4, and all other energies have a relative probability of 10. The procedure to determine the outgoing energy and angle is similar to the method discussed above, except that the sampled probability distribution is now skewed accordingly.

Continuous Outgoing Energies

The outgoing energy distribution is given in LAW 61, which is like LAW 44 but tabular angular distribution instead of Kalbach-Mann parameters for out going angle sampling. If the thermal data was processed with iwt = 2 in NJOY, then the outgoing energy spectra is represented by a continuous outgoing energy spectra in tabular form with linear-linear interpolation. The sampling of the outgoing energy portion of this format is very similar to Correlated Energy and Angle Distribution, but the sampling of the correlated angle is performed as it was in the other two representations.

4.10. Unresolved Resonance Range Probability Tables (~high Z E < 150 keV)

In the unresolved resonance energy range, resonances may be so closely spaced that it is not possible for experimental measurements to resolve all resonances and this is significant for systems whose spectra peak in or near the unresolved resonance range. The method of treating probability method in Monte Carlo was explained by Levit[7].To properly account for self-shielding in this energy range, ACE table contains probability tables and varies from nuclide to the other. For instance, the unresolved resonance range in ACE table is 2.25 - 25 keV for U-235, 10 - 149.03 keV for U-238, and 2.5 - 30 keV for Pu-239 etc. Each probability table for a nuclide contains the following information at a number of incoming energies within the unresolved resonance range:

1. cumulative probability tables for incident energies (~19)
2. total cross section or total factor
3. elastic cross section or elastic factor
4. fission cross section or fission factor
5. (n,γ) cross section or (n,γ) factor
6. neutron heating number or heating factor

A flag is also present in the probability table that specifies whether an inelastic cross section should be calculated. If so, this is done from a normal reaction cross section (either MT=51 or a special MT). Finally, if the cross sections defined are above are specified to be factors and not true cross sections, they are multiplied by the underlying smooth cross section in the unresolved range to get the actual cross sections. Lastly, the total cross section is calculated as the sum of the elastic, fission, capture, and inelastic cross sections.

Sampling cross sections from probability tables is straightforward. At each of a number of incident energies there is a table of cumulative probabilities (typically 20) and the value of the near-total, elastic, fission, and radiative capture cross sections and heat deposition numbers corresponding to those probabilities. These data supplement the usual continuous data; if probability tables are turned off , then the usual smooth cross section is used. But if the probability tables are turned on (default), if they exist for the nuclide of a collision, and if the energy of the collision is in the unresolved resonance energy range of the probability tables, then the cross sections are sampled from the tables. The near-total is the total of the elastic, fission, and radiative capture cross sections; it is not the total cross section, which may include other absorption or inelastic scatter in addition to the near-total.

The impact of the probability-table approach has been studied and found to have negligible impact for most fast and thermal systems. Small but significant changes in reactivity may be observed for plutonium and U-233 systems, depending upon the detailed shape of the spectrum. However, the probability-table method can produce substantial increases in reactivity for systems that include large amounts of U-238 and have high fluxes within the unresolved resonance region

4.11. Transforming a Particle's Coordinates from CM to Lab system

The exiting particle energy and direction are always given in the target-at-rest (laboratory) coordinate system. For convenience, the angular distributions in ACE files for elastic, discrete inelastic level scattering, and some ENDF-6 inelastic reactions are to be given in the center-of-mass system, and the angular distributions for all other reactions are given in the target-at-rest (Lab) system. The exiting particle energy and direction in the target-at-rest coordinate system are related to the center-of-mass energy and direction as follows:

Once the cosine of the scattering angle μ has been sampled either from a angle distribution or a correlated angle energy distribution, we are still left with the task of transforming the particle's coordinates. If the outgoing energy and scattering cosine were given in the center-of-mass system, then we first need to transform these into the laboratory system.

$$E' = E'_{cm} + \left[\frac{E + 2\mu_{cm}(A+1)\sqrt{EE'_{cm}}}{(A+1)^2}\right] \quad (3.50)\ ; \text{and}$$

$$\mu_{lab} = \mu_{cm}\sqrt{\frac{E'_{cm}}{E'}} + \frac{1}{A+1}\sqrt{\frac{E}{E'}} \quad (3.51)$$

E' = exiting particle energy (laboratory),
E'_{cm} = exiting particle energy (center-of-mass),
E = incident particle energy (laboratory),
μcm = cosine of center-of-mass scattering angle,
μlab = cosine of laboratory scattering angle, and
A = atomic weight ratio (mass of nucleus divided by mass of incident particle).
where Ein = incident neutron energy, μcm = center-of-mass cosine of the angle between incident

and exiting particle directions, and *A* mass of collision nuclide in units of the mass of a neutron (atomic weight ratio).

The scattering cosine still only tells us the cosine of the angle between the original direction of the particle and the new direction of the particle. If we express the pre-collision direction of the particle as $\Omega = (u, v,)$ and the post-collision direction of the particle as $\Omega' = (u', v',')$, it is possible to relate the pre- and post-collision components. We first need to uniformly sample an azimuthal angle φ in $[0, 2\pi)$. After the azimuthal angle has been sampled, the post-collision direction is calculated as

$$u' = \mu u + \frac{\sqrt{1-\mu^2}\,(uwcos\phi - vsin\phi)}{\sqrt{1-w^2}}$$
$$v' = \mu v + \frac{\sqrt{1-\mu^2}\,(vwcos\phi - usin\phi)}{\sqrt{1-w^2}}$$
$$w' = \mu w - \sqrt{1-\mu^2}\sqrt{1-w^2}cos\phi$$

References

1. R. E. MacFarlane and D. W. Muir. *The NJOY Nuclear Data Processing System*. LA-12740-M. Los Alamos National Laboratory, 1994.
2. C. J. Everett and E. D. Cashwell, "A Third Monte Carlo Sampler," Los Alamos National Laboratory Report, LA-9721-MS, (March 1983).
3. J. F. Briesmeister, Ed., "MCNP - A General Monte Carlo N-Particle Transport Code, Version 4C," LA-13709-M (April 2000).
4. P.K. Romano, B. Forget , The OpenMC Monte Carlo particle transport code, Annals of Nuclear Energy 51 (2013) 274–281
5. [Gelbard] Ely M. Gelbard, "Epithermal Scattering in VIM," FRA-TM-123, Argonne National Laboratory (1979).
6. https://github.com/openmc-dev/openmc.git, **OpenMC Documentation, Release 0.9.0**
7. Levitt, L.B., 1972. The probability table method for treating unresolved neutron resonances in Monte Carlo calculations. Nucl. Sci. Eng. 49, 450–457.

5. Constructive Solid Geometry

This chapter deals with the constructive solid geometry. In this method, any complex configuration of materials can be described by its bounding surfaces and it is referred as a cell. A cell is filled with a homogenous material or void or it can be filled with any universe or lattice elements. A universe is formed with two or more cells. Basic introduction to the formation of geometry, the format of cell description, and the format of surface description are presented. Further, the two important calculations involved with geometrical objects are (i) finding a cell/lattice element in which the particle is in given its position, and (ii) calculating the distance to the nearest surface. Besides, to handle reflective surfaces of the geometry, the derivation of reflective angle for various surfaces is presented in sequel.

5.1. Introduction

The basic building block of the geometry is the cell, which is a region of space determined using simple bounding analytical surfaces. Each cell is filled with a homogeneous material composition, void or another universe or a lattice. MCkeff allows building of arbitrary complicated cells by combining bounding surfaces with union (':'), intersection (' ') and complement operators ('#'). A surface is defined as a collection of points that satisfy an equation of the form, $f(x,y,z)=0$, where $f(x,y,z)$ is a given function either a linear or quadratic function. All surfaces are uniquely identified with integers in the code. In a cell description, if a surface number present with a negative sign, it refers to a space (negative half space) of all coordinates for which $f(x,y,z)<0$, and if a surface number present with positive/no sign refers to a space (positive half space) of all coordinates for which $f(x,y,z)>0$. Fig.5.1 depicts the half spaces denoted by -1 an elliptical region inside and +1 denotes the other positive half space outside elliptical region.

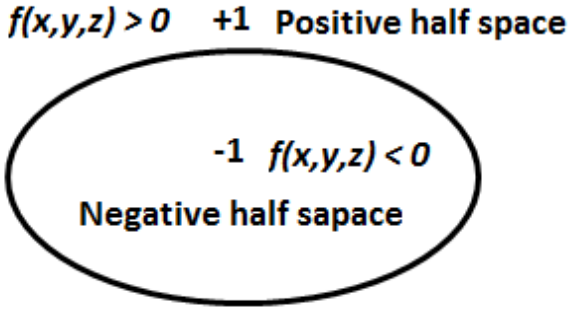


Fig.4.1 Showing negative and positive half spaces

All of the general purpose Monte Carlo codes are capable of handling any complex geometrical objects since they use combination of either analytical 3D geometrical shapes or 2D surfaces to describe the geometry[1-6]. However, Deterministic codes can handle only one type of objects either slabs (x, y, z), or cylinders (ρ, θ, z), or spheres(r, θ, ϕ) owing to differences in their coordinate systems. As a result, the form of radiation transport equation is different in different geometries. However, Monte Carlo codes solve 3D transport equation in rectangular coordinate system globally accounting different objects in their local coordinate systems. Therefore, it is possible in Monte Carlo codes to treat arbitrary complex configuration of materials in cells.

5.2. Format of Surface Definition

Surfaces are the backbone of geometry. MCkeff code allows analytical surfaces defined by equations (linear or quadratic) in order to describe any complex homogenous material region called

cells. The syntax of the surface card is same as that followed in MCNP code[1] and is reproduced below:

<j> [n] <a> < list of surface coefficients>

j = surface number 1<j < 999, with asterisk for a reflecting surface or plus for a white boundary.
n = absent or 0 for no coordinate transformation.
= > 0; specifies number of a TRn card. (not used at present)
= < 0; specifies surface j is periodic with surface n.
a = equation mnemonic from Table 5.1
list = one to ten entries, as required and is shown in Table 5.1.

The surface types, equations, their mnemonics, and the sequence of the data entries required are given in table 5.1. For more explanation, the user is advised to refer Chapter-3 of MCNP manual[1]. The following surfaces are allowed in the MCkeff code at present.

Table 5.1. List of analytical surfaces allowed in MCkeff and the input data entries required

Sno.	Mne monic	Type	Description	Equation	Data entries
1	P	Plane	General	$Ax + By + cZ - D = 0$	A,B,C,D
2	PX		Normal to X-axis	$x - D = 0$	D
3	PY		Normal to Y-axis	$y - D = 0$	D
4	PZ		Normal to Z-axis	$z - D = 0$	D
5	So	Sphere	Centered at origin	$x^2 + y^2 + z^2 - R^2 = 0$	R
6	S		General	$(x - x_0)^2 + (y - y_0)^2 + (z - z_0)^2 - R^2 = 0$	x_0, y_0, z_0, R
7	Sx		Centre on X-axis	$(x - x_0)^2 + y^2 + z^2 - R^2 = 0$	x_0, R
8	Sy		Centre on Y-axis	$x^2 + (y - y_0)^2 + z^2 - R^2 = 0$	y_0, R
9	Sz		Centre on Z-axis	$x^2 + y^2 + (z - z_0)^2 - R^2 = 0$	z_0, R
10	C/X	Cylinder	Parallel to X-axis	$(y - y_0)^2 + (z - z_0)^2 - R^2 = 0$	y_0, z_0, R
11	C/Y		Parallel to Y-axis	$(x - x_0)^2 + (z - z_0)^2 - R^2 = 0$	x_0, z_0, R
12	C/Z		Parallel to Z-axis	$(x - x_0)^2 + (y - y_0)^2 - R^2 = 0$	x_0, y_0, R
13	CX		On X-axis	$y^2 + z^2 - R^2 = 0$	R
14	CY		On Y-axis	$x^2 + z^2 - R^2 = 0$	R
15	CZ		On Z-axis	$x^2 + y^2 - R^2 = 0$	R
16	K/X	Cone	Parallel to X-axis	$\sqrt{(y - y_0)^2 + (z - z_0)^2} + t(x - x_0) = 0$	x_0, y_0, z_0, t^2
17	K/Y		Parallel to Y-axis	$\sqrt{(x - x_0)^2 + (z - z_0)^2} + t(y - y_0) = 0$	x_0, y_0, z_0, t^2
18	K/Z		Parallel to Z-axis	$\sqrt{(x - x_0)^2 + (y - y_0)^2} + t(z - z_0) = 0$	x_0, y_0, z_0, t^2
19	KX		On X-axis	$\sqrt{y^2 + z^2} + t(x - x_0) = 0$	x_0, t^2
20	KY		On Y-axis	$\sqrt{x^2 + z^2} + t(y - y_0) = 0$	y_0, t^2
21	KZ		On Z-axis	$\sqrt{x^2 + y^2} + t(z - z_0) = 0$	

Example 1: j PY 3
This describes a plane normal to the y-axis at y = 3 with positive sense for all points with y > 3.
Example 2: j K/Y 0 0 2 0.25 1
This specifies a cone whose vertex is at (x; y; z) = (0;0; 2) and whose axis is parallel to the y-axis. The tangent t of the opening angle of the cone is 0.5 (note that t^2 is entered) and '1' suggests that only the positive (right hand) sheet of the cone is used. Points outside the cone have a positive sense.

5.3 Format of Cell Definition

Cell is the basic element in a geometry description. Cells are formed with their bounding surfaces along with the binary operators. The specification of the cell must include a list of surfaces along with binary operators (union ':', intersection ' ') to define the region of interest. MCkeff also permits unitary operator called complement operator ('#') meaning not in the region. Each cell is defined using a set of positive and negative surface numbers, corresponding to the surface identifiers defined in the surface cards. The program identifies the cell region by an unique number. Three types of cell filling is possible, namely (i) material (includes void), (ii) universe , or (iii) Lattice. The syntax of cell card is same as that given in MCNP manual[1] and it is

j m d geom params
or
j LIKE n BUT list

j = cell number; 1<j < 999.
M = 0 if the cell is a void.
= material number if the cell is not a void. This indicates that the cell is to contain material m, which is specified on the Mm card. To be defined later.
D = absent if the cell is a void.
= cell material density. A positive entry is interpreted as the atomic density in units of atoms/barn-cm. A negative entry is interpreted as the mass density in units of g/cm^3.
geom = specification of the geometry of the cell. It consists of signed surface numbers and Boolean operators that specify how the regions bounded by the surfaces are to be combined.
params = optional specification of cell parameters by entries in the keyword=value form.
n = name of another cell
list = set of keyword=value specifications that define the attributes that differ between cell n and j.

In the geometry specification, a signed surface number stands for the region on the side of the surface where points have the indicated sense. The plus sign for positive sense is optional. The regions are combined by Boolean operators: intersection (no symbol/ implicit, like multiplication in algebra) ;union, : ; and complement, #. Parentheses can be used to control the order of the operations. of that cell number.

Example: 3 0 -1 2 -4 $ definition of cell 3
Complement operator #3 is equivalent to #(-1 2 -4)

5.4. Universe and Lattice Definitions

The MCkeff allows use of repeated structures viz. universes (U) and Lattices (LAT) or arrays in the input specification similar to that of MCNP. This makes it possible to describe only once the cells and surfaces of any structure (unit) that appears more than once in a geometry. This unit then can be replicated at other locations using keywords such as U (universe), FILL, TRCL, LAT (lattice) and with "LIKE m BUT" construct on a cell card. This feature will be of use specifying fuel pin and repeating it within a fuel assembly or a fuel assembly within a core. A universe is defined as a collection cells that occupy entire space or a lattice cell. Universe size will be truncated by the boundaries of the cell that contains it. The two types of lattice shapes allowed in the code namely rectangular or hexagonal prisms. At present rotation and translation are restricted to cells which are filled by the universe. It is not necessary to use or assign universes in a geometry if there are no repeated structures. Any cell in the geometry that is not assigned to a specified universe is

automatically part of the *base universe* whose coordinates are just the normal coordinates in Euclidean space.

5.5. Finding a Distance to the Nearest Surface

In Monte Carlo simulation, one of the most basic algorithms is determining the distance to the surface. if we compute the distance to all surfaces bounding a cell in the direction of particle travel, we can determine the nearest one. Suppose we have a particle at (x_p, y_p, z_p) traveling in the direction(u, v, w), the distance 'd' to the surface is computed by substituting variables of surface equation $f(x, y, z) = 0$ with the following

$$x = x_p + du,\ y = y_p + dv,\ z = z_p + dw \qquad (5.1).$$

When repeated structures are used in the geometry which are specified in a local coordinate system (levels), it is necessary to calculate the distance to the surfaces bounding the cell in each level. This should be done starting the highest (most global) level going down to the lowest (most local) level. In case of lattice filled cell, no physical boundaries exist, it is necessary to check the lattice index at first followed by the distance to the surfaces in local coordinates representing lattice boundaries.

Derivation of distance to the representative surface in each group of list given in the table 5.1 follows in sequel. To find the distance d to a surface from a particle starting position (x_p, y_p, z_p) traveling in the direction (u, v, w), we need to substitute variables of the surface equation from those given equation (5.1), which results in the general following form:

$$f(x_p + du, y_p + dv, z_p + dw) = 0. \qquad (5.2)$$

Solution 'd' to equation (5.2) can be different for different surfaces. If solutions to equation(5.2) does not exist or the solutions are complex, then the particle's direction of travel will not intersect the surface. If the solution to the equation (5.2) is negative, this means that the surface is "behind" the particle and it is ignored. if the solution is positive, it means the surface is "in front" of the particle and the distance and the surface number are stored if it is a minimum. The minimum condition is checked to find out whether it is closer than the previously-computed distances to the other surfaces. Care must be taken in deciding the minimum distance since due to floating point operation which may result in nearly same values. To avoid this ambiguity, we need to check

$$\frac{|d - d_{min}|}{d_{min}} < \epsilon \qquad (5.3)$$

where d is the distance to a surface just calculated, d_{min} is the minimum distance found so far and $\epsilon = 10^{-14}$. If equation (5.3) is satisfied, only the previous surface is taken as the nearest surface.

5.5.1. Generic Plane

First group in table 5.1 contain linear equation of planes. The equation for a generic plane is $Ax + By + Cz - D = 0$. From the solution of this equation, solutions to other planes can be easily deduced. we solve this equation for a distance 'd' by replacing its variables as defined in equation (5.1) and is given by

$$A(x_p + du) + B(y_p + dv) + C(z_p + dw) = D \quad (5.4)\ .$$

In equation (5.4) after rearranging the terms and the solution d can be obtained as

$$\mathrm{d} = \frac{D - Ax_p - By_p - Cz_p}{Au + Bv + Cw} \qquad (5.5)$$

before evaluating equation (5), we need to check whether the denominator is zero. If so, this means that the particle's direction of flight is parallel to the plane and it will therefore never hit the plane. If 'd' is positive, the surface number and distance is stored for processing further. Similarly, the distance to other planes normal to coordinate axes can be deduced by assigning appropriate values to A,B,C. For instance distance for a plane normal to X-axis *(x- x_0=0)* can be obtained setting A=1,B=C=0 in equation (5.5) as *d= (D-x_p)/u*.

5.5.2. Cylinder Parallel to an Axis

Let us consider a cylinder parallel to, for example, the x-axis with the center at $(y_0, z_0)$is

$$(y - y_0)^2 + (z - z_0)^2 = R^2 \quad (5.6)$$

Replacing the variables y and z in the above from equation (5.1), we get

$$(y_p + dv - y_0)^2 + (z_p + dw - z_0)^2 = R^2$$

Let us define $\bar{y} = y_p - y_0$,*and* $\bar{z} = z_p - z_0$, then the equation is

$$(\bar{y} + dv)^2 + (\bar{z} + dw)^2 = R^2 \qquad (5.7)$$

Expanding equation (7) and rearranging terms, we obtain

$$(v^2 + w^2)d^2 + 2(\bar{y}v + \bar{z}w)\, d + (\bar{y}^2 + \bar{z}^2 - R^2) = 0 \quad (5.8)$$

This is a quadratic equation in d. To simplify notation, let us define $a = v^2 + w^2, b = (\bar{y}v + \bar{z}w), and\ c = (\bar{y}^2 + \bar{z}^2 - R^2)$ Thus, the distance is just the solution to $ad^2 + 2bd + c = 0$, and is given by

$$d = \frac{-b \pm \sqrt{b^2 - ac}}{a} \qquad (5.9)$$

A few conditions must be checked for. If $a = 0$, this means the particle is parallel to the cylinder and will thus never intersect it. Also, if $b^2 - ac < 0$, this means that both solutions to the quadratic are complex. In physical terms, this means that the ray along which the particle is traveling does not make any intersections with the cylinder. If we do have intersections and $c < 0$, this means that the particle is inside the cylinder. Thus, one solution should be positive and one should be negative. Clearly, the positive distance will occur when the sign on the square root of the discriminant is positive since $a > 0$. If we have intersections and $c > 0$ this means that the particle is outside the cylinder. Thus, the solutions to the quadratic are either both positive or both negative. If they are both positive, the smaller (closer) one will be the solution with a negative sign on the square root of the discriminant. The same equations and logic can be used for cylinders that are parallel to the y- or z-axis with appropriate substitution of variables.

5.5.3. Sphere

The equation for a sphere whose origin at (x_0, y_0, z_0) and radius R is

$$(x - x_0)^2 + (y - y_0)^2 + (z - z_0)^2 = R^2 \quad (5.10)$$

By substituting values of x,y and z from equation (5.1) in to equation (5.10), we get

$$(x_p + du - x_0)^2 + (y_p + dv - y_0)^2 + (z_p + dw - z_0)^2 = R^2 \quad (5.11)$$

Let us define $\bar{x} = x_p - x_0$, $\bar{y} = y_p - y_0$, and $\bar{z} = z_p - z_0$. We then have

$$(\bar{x} + du)^2 + (\bar{y} + dv)^2 + (\bar{z} + dw)^2 = R^2 \quad (5.12)$$

Expanding equation (12) and rearranging terms, we obtain

$$(u^2 + v^2 + w^2)d^2 + 2(\bar{x}u + \bar{y}v + \bar{z}w)\, d + (\bar{x}^2 + \bar{y}^2 + \bar{z}^2 - R^2) = 0$$

This is a quadratic equation in d. To simplify notation, let us define $b = \bar{x}u + \bar{y}v + \bar{z}w$, and $.c = \bar{x}^2 + \bar{y}^2 + \bar{z}^2 - \bar{R}^2$, and noting $(u^2 + v^2 + w^2) = 1$ Thus, the distance is just the solution to $d^2 + 2bd + c = 0$:

$$d = -b \pm \sqrt{b^2 - c} \quad (5.13)$$

If the discriminant $b^2 - c < 0$, this means that both solutions to the quadratic are complex. In physical terms, this means that the ray along which the particle is traveling does not make any intersections with the sphere. If $c < 0$, this means that the particle is inside the sphere and makes intersection on either side. Thus, one solution should be positive and one should be negative. The positive distance will occur when the sign on the square root of the discriminant is positive. If we have intersections but $c > 0$ this means that the particle is outside the sphere. The solutions to the quadratic will then be either both positive or both negative. If they are both positive, the smaller (closer) one will be the solution with a negative sign on the square root of the discriminant.

5.5.4. Cone Parallel to an Axis

Let us consider a cone parallel to, for example, the x-axis with the center at $(x_0, y_0, z_0)$is

$$(y - y_0)^2 + (z - z_0)^2 - t^2((x - x_0)^2 = 0 \quad (5.14)$$

Where t= tan(θ), opening half-angle of the cone. Replacing the variables x , y and z in the above from equation (5.1), we get

$$(y_p + dv - y_0)^2 + (z_p + dw - z_0)^2 - t^2((x_p + du - x_0)^2 = 0$$

Let us define $\bar{y} = y_p -, \bar{z} = z_p - z_0, and\ \bar{x} = x_p - x_0$, then the equation is

$$(\bar{y} + dv)^2 + (\bar{z} + dw)^2 - t^2((\bar{x} + du)^2 = 0 \quad (5.15)$$

Expanding equation (7) and rearranging terms, we obtain

$$(v^2 + w^2 - t^2u^2)d^2 + 2(\bar{y}v + \bar{z}w - t\bar{x}u)\, d + (\bar{y}^2 + \bar{z}^2 - t^2\bar{x}^2) = 0 \quad (5.16)$$

This is a quadratic equation for d. To simplify notation, let us define $a = v^2 + w^2 - t^2u^2, b = (\bar{y}v + \bar{z}w - t\bar{x}u), and\ c = (\bar{y}^2 + \bar{z}^2 - t^2\bar{x}^2)$ Thus, the distance is just the solution to $ad^2 + 2bd + c = 0,$ and is given by

$$d = \frac{-b \pm \sqrt{b^2 - ac}}{a} \quad (5.17)$$

A condition a=0 must be checked for. The comments made with respect to quadratic equations are valid here also.

5.6. Finding a Cell of a Given Point $(x_p, y_{p,} z_p)$

Another basic algorithm used in Monte Carlo codes is to determining the cell that contains a given point in the global coordinate system, i.e. if the particle's position is $(x_p, y_{p,} z_p)$, what cell is it currently in. As mentioned before cell may be filled with either the material (base cell, highest level and global coordinates) or with another universe (lower level with local coordinates) or a lattice element (lower level with local coordinates). Since there is a possibility of describing system in multiple levels of coordinates, we must perform a recursive search for the cell. First, we start in the highest (most global) universe, which we call the base universe, and loop over each cell within that universe. The specified point is inside the cell if it satisfies sense of surfaces describing the cell. The search is terminated when we have identified the cell filling the normal material containing the point. If the cell is filled with another universe, we then search all cells within that universe to see if any of them contain the specified point. If the cell is filled with a lattice, the position within the lattice is determined, and then whatever universe fills that lattice position is recursively searched. The search ends once a cell containing a normal material is found that contains the specified point.

5.7. Handling Surface Crossings

A particle will cross a surface if the distance to the nearest surface is greater than the distance sampled to the next collision. To decide on the fate of the particle, we need to check for boundary condition of the surface. Four types of boundary conditions handled in the MCkeff code namely (i) transmitive (ii) vacuum (iii) reflective and (iv) periodic. If the surface is transmitive, the neighboring cell is found out and the particle cell number is changed. If a vacuum boundary condition applied, the particle is killed and the program picks up a new particle from the source bank to continue simulation. If the surface is reflective, the particle's direction is changed according to the procedure given in section under Reflective Boundary conditions. For a periodic boundary condition (only planes normal to coordinate axes), the particle position is changed to the opposite surface (periodic surface) of the same cell and simulation continues.

5.8. Finding a Lattice Tile

Lattice tile is a volumetric region of a cell. To find out lattice element containing the given point (particle position) , its index must be determined at first and then the universe filling this lattice element must be assigned. There are two types lattice filling is allowed in the code viz. rectangular and hexahedral. The search algorithm of finding index of lattice element depends on the type lattice filling.

5.8.1. Rectangular Lattice Indexing

Indices are assigned to tiles in a rectilinear lattice based on the tile's position along the x, y, and z axes. A tile is a volume element and a part of the big cell. Figure 5.2 maps the indices for a 2D lattice. The indices, (1, 1), map to the lower-left tile. (1, 2) and (2, 2) map to the lower-right and upper-right tiles, respectively.

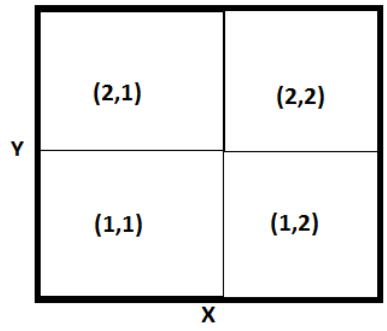


Fig.5.2. Rectangular lattice in a cell(2x2 lattice)
Tile nos. are indicated (Ix, Iy)

In general, a lattice tile is specified by the three indices, (ix, iy, iz). If a particle's current coordinates are (x, y, z) then the indices can be determined from these formulas:

$$i_x = \left\lfloor \frac{x - x_0}{p_x} \right\rfloor$$
$$i_y = \left\lfloor \frac{y - y_0}{p_y} \right\rfloor$$
$$i_z = \left\lfloor \frac{z - z_0}{p_z} \right\rfloor$$

where (x_0, y_0, z_0) are the coordinates to the lower-left-bottom corner of the lattice cell, and p_x, p_y, p_z are the pitches along the x, y, and z axes, respectively.

5.8.2. Hexagonal Lattice Indexing

A skewed coordinate system is used for indexing hexagonal lattice tiles. Rather than a y-axis, another axis is used that is rotated 30 degrees counter-clockwise from the y-axis. This axis is referred to as the α-axis. Figure 5.3 shows how 2D hexagonal tiles are mapped with the (x, α) basis. In this system, (0, 0) maps to the center tile(blue), (0, 2) to the top tile(green), and (2, -1) to the middle tile on the right side(yellow).

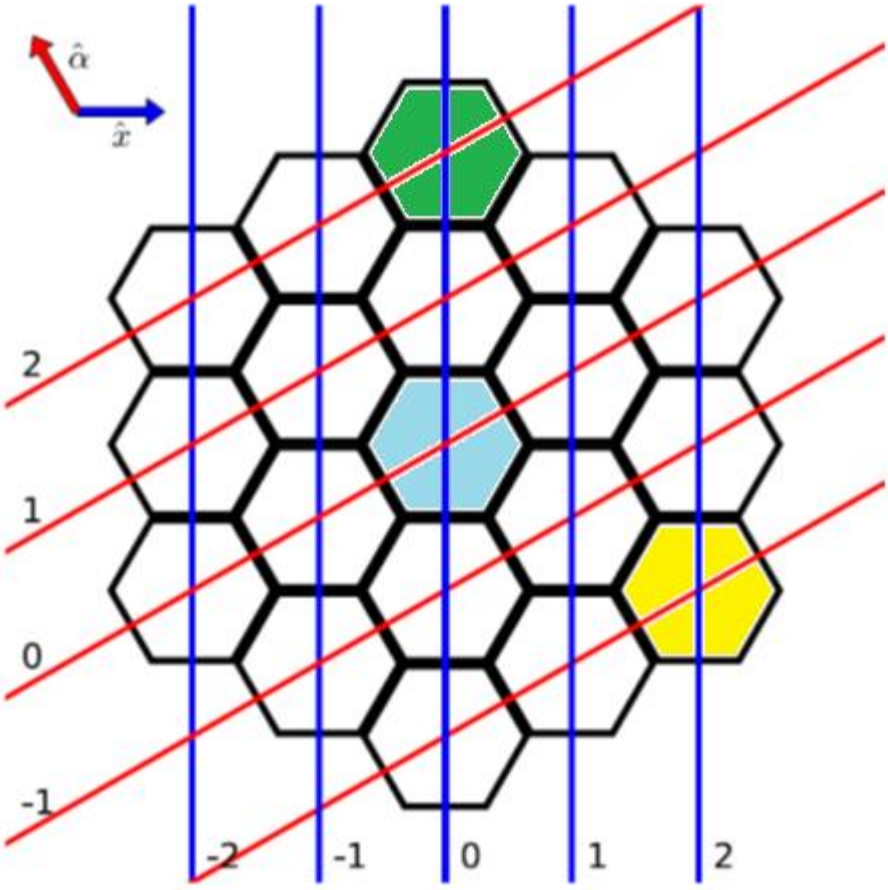


Fig. 5.3. Hexagonal lattice tile indices.

Unfortunately, the indices cannot be determined with one simple formula as before. Indexing requires a two-step process, a coarse step which determines a set of four tiles that contains the particle and a fine step that determines which of those four tiles actually contains the particle.
In the first step, indices are found using these formulas:

$$\alpha = -\frac{x}{\sqrt{3}} + y$$

$$i_x^* = \left\lfloor \frac{x}{p_0\sqrt{3}/2} \right\rfloor$$

$$i_\alpha^* = \left\lfloor \frac{\alpha}{p_0} \right\rfloor$$

where $p0$ is the lattice pitch (in the x-y plane). The true index of the particle could be $((i_x^*, i_\alpha^*), (i_x^* + 1, i_\alpha^*), (i_x^*, i_\alpha^* + 1), (i_x^* + 1, i_\alpha^* + 1)$.

The second step selects the correct tile from that neighborhood of 4. OpenMC[2] does this by calculating the distance between the particle and the centers of each of the 4 tiles, and then picking the closest tile. This works because regular hexagonal tiles form a Voronoi tessellation which means that all of the points within a tile are closest to the center of that same tile. MCkeff adopts the same algorithms as is followed in reference 2. Indexing along the z-axis uses the same method from rectilinear lattices, i.e.

$$i_z = \left[\frac{z - z_0}{p_2}\right]$$

5.9 Reflective Boundary Conditions

If the velocity of a particle is $\vec{V}$ it crosses a surface of the form $f(x, y, z) = 0$ with a reflective boundary condition, it can be shown based on geometric arguments that the velocity vector will then become

$$\boldsymbol{V' = V - 2(V.\hat{n})\hat{n}} \quad \boldsymbol{(5.18)}$$

where $\hat{\boldsymbol{n}}$ is a unit vector normal to the surface at the point of the surface crossing. The rationale for this can be understood by noting that $(\boldsymbol{V}.\hat{\boldsymbol{n}})\hat{\boldsymbol{n}}$ is the projection of the velocity vector onto the normal vector. By subtracting two times this projection, the velocity is reflected with respect to the surface normal. Since the magnitude of the velocity of the particle will not change as it undergoes reflection, we can work with the direction of the particle instead, simplifying equation (14) to

$$\boldsymbol{\Omega' = \Omega - 2(\Omega.\hat{n})\hat{n}} \qquad \boldsymbol{(5.19)}$$

where $\boldsymbol{V = ||V||\Omega}$. The direction of the surface normal will be the gradient of the surface at the point of crossing, i.e. $\hat{\boldsymbol{n}} = \boldsymbol{\nabla f(x, y, z)}$. Substituting this into equation (15), we get

$$\boldsymbol{\Omega' = \Omega - \frac{2(\Omega.\nabla f)\nabla f}{||\nabla f||^2}} \quad \boldsymbol{(5.20)}$$

If we write the initial and final directions in terms of their vector components, $\boldsymbol{\Omega = (u, v, w)}$ and, $\boldsymbol{\Omega'^{=(}u', v', w')}$ is allows us to represent equation (5.20) as a series of equations:

$$\boldsymbol{u' = u - \frac{2(\Omega.\nabla f)}{||\nabla f||^2}\frac{\partial f}{\partial x}}$$

$$\boldsymbol{v' = v - \frac{2(\Omega.\nabla f)}{||\nabla f||^2}\frac{\partial f}{\partial y}}$$

$$\boldsymbol{w' = w - \frac{2(\Omega.\nabla f)}{||\nabla f||^2}\frac{\partial f}{\partial z}} \quad \boldsymbol{(5.21)}$$

One can then use equation (15) to develop equations for transforming a particle's direction given the equation of the surface.

5.10 Plane Perpendicular to an Axis

For a plane that is perpendicular to an axis, the rule for reflection is almost so simple that no derivation is needed at all. Nevertheless, we will proceed with the derivation to confirm that the rules of geometry agree with our intuition. The gradient of the surface $f(x, y, z) = x - x_0) = 0$ is

simply $\nabla f = (1, 0, 0)$. Note that this vector is already normalized, i.e. $||\nabla f|| = 1$. The second two equations (17) tell us that v and w do not change and the first tell us that

$$\boldsymbol{u' = u - 2u = -u} \quad \textbf{(5.22)}$$

We see that reflection for a plane perpendicular to an axis only entails negating the directional cosine for that axis.

5.11 Generic Plane

A generic plane has the form $f(x, y, z) = Ax + By + Cz - D = 0$. Thus, the gradient to the surface is simply $\nabla f = (A, B, C)$whose norm squared is $A^2 + B^2 + C^2$. This implies that

$$\frac{\boldsymbol{2(\Omega . \nabla f)}}{\boldsymbol{||\nabla f||^2}} = \frac{\boldsymbol{2(Au + Bv + Cw)}}{\boldsymbol{(A^2 + B^2 + C^2\ )}} \quad \textbf{(5.23)}$$

Substituting equation (20) into equation (17) gives us the form of the solution. For example, the x-component of the reflected direction will be

$$\boldsymbol{u' = u - \frac{2(Au + Bv + Cw)}{(A^2 + B^2 + C^2\ )}} \quad \textbf{(5.24)}$$

5.12 Cylinder Parallel to an Axis

A cylinder parallel to, for example, the x-axis has the form $f(x, y, z) = (y - y_0)^2 + (z - z_0)^2 - R^2 = 0$. Thus, the gradient to the surface is

$$\boldsymbol{\nabla f = 2\begin{pmatrix} 0 \\ y - y_0 \\ z - z_0 \end{pmatrix} = 2\begin{pmatrix} 0 \\ \bar{y} \\ \bar{z} \end{pmatrix}} \quad \textbf{(5.25)}$$

where we have introduced the constants $\bar{y}$and $\bar{z}$. Taking the square of the norm of the gradient, we find that

$$\boldsymbol{||\nabla f||^2 = 4\bar{y}^2 + 4\bar{z}^2 = 4R^2} \quad \textbf{(5.26)}$$

This implies that

$$\frac{\boldsymbol{2(\Omega . \nabla f)}}{\boldsymbol{||\nabla f||^2}} = \frac{\boldsymbol{\bar{y}v + \bar{z}w}}{\boldsymbol{R^2}} \quad \textbf{(5.27)}$$

Substituting equations (5.26) and (5.27) into equations (5.21) gives us the form of the solution. In this case, the x-component will not change. The y- and z-components of the reflected direction will be

$$\boldsymbol{v' = v - \frac{2(\bar{y}v + \bar{z}w)\bar{y}}{R^2}} \quad \textbf{(5.28)}$$

$$w' = w - \frac{2(\bar{y}v + \bar{z}w)\bar{z}}{R^2} \quad (5.29)$$

5.13 Sphere

The surface equation for a sphere has the form $f(x,y,z) = (x-x_0)^2 + (y-y_0)^2 + (z-z_0)^2 - R^2 = 0$. Thus, the gradient to the surface is

$$\nabla f = 2\begin{pmatrix} x-x_0 \\ y-y_0 \\ z-z_0 \end{pmatrix} = 2\begin{pmatrix} \bar{x} \\ \bar{y} \\ \bar{z} \end{pmatrix} \quad (5.30)$$

where we have introduced the constants $\bar{x}, \bar{y}, \bar{z}$. Taking the square of the norm of the gradient, we find that

$$||\nabla f||^2 = 4\bar{x}^2 + 4\bar{y}^2 + 4\bar{z}^2 = 4R^2 \quad (5.31)$$

This implies that

$$\frac{2(\Omega.\nabla f)}{||\nabla f||^2} = \frac{\bar{x}u + \bar{y}v + \bar{z}w}{R^2} \quad (5.31)$$

Substituting equations (5.31) and (5.32) into equation (5.21) gives us the form of the solution:

$$u' = u - \frac{2(\overline{\bar{x}u + \bar{y}}v + \bar{z}w)\bar{x}}{R^2}$$

$$v' = v - \frac{2(\overline{\bar{x}u + \bar{y}}v + \bar{z}w)\bar{y}}{R^2}$$

$$w' = w - \frac{2(\overline{\bar{x}u + \bar{y}}v + \bar{z}w)\bar{z}}{R^2} \quad (5.32)$$

5.14 Cone Parallel to an Axis

A cone parallel to, for example, the z-axis has the form$f(x,y,z) = (x-x_0)^2 + (y-y_0)^2 + t^2(z-z_0)^2 = 0$. Thus, the gradient to the surface is

$$\nabla f = 2\begin{pmatrix} x-x_0 \\ y-y_0 \\ -t^2(z-z_0) \end{pmatrix} = 2\begin{pmatrix} \bar{x} \\ \bar{y} \\ -t^2\bar{z} \end{pmatrix} \quad (5.33)$$

where we have introduced the constants $\bar{x}$, $\bar{y}$, and $\bar{z}$. Taking the square of the norm of the gradient, we find that

$$||\nabla f||^2 = 4\bar{x}^2 + 4\bar{y}^2 + 4t^4\bar{z}^2$$

$$= 4t^2\bar{z}^2 \quad + 4t^4\bar{z}^2$$

$$= 4t^2 \quad (1 + t^2)\,\bar{z}^2 \qquad (5.34)$$

This implies that

$$\frac{2(\Omega.\nabla f)}{\left|\left|\nabla f\right|\right|^2} = \frac{\bar{x}u + \bar{y}v - t^2\bar{z}w}{t^2\,(1 + t^2)\,\bar{z}^2} \qquad (5.35)$$

Substituting equations (5.34) and (5.35) into equation (5.21) gives us the form of the solution:

$$u' = u - \frac{2(\bar{x}u + \bar{y}v - t^2\bar{z}w)\bar{x}}{t^2\,(1 + t^2)\bar{z}^2}$$

$$v' = v - \frac{2(\bar{x}u + \bar{y}v - t^2\bar{z}w)\bar{y}}{t^2\,(1 + t^2)\bar{z}^2}$$

$$w' = w - \frac{2(\bar{x}u + \bar{y}v - R^2\bar{z}w)\bar{z}}{t^2\,(1 + t^2)\bar{z}^2} \qquad (5.36)$$

5.15 Building Neighbor Lists

This is required in order to Monte Carlo codes faster while locating the particle cell or when the particle crosses surface and enters a new cell.. After the geometry has been loaded and stored in memory from an input file, MCkeff builds a list for each surface containing any cells that are bounded by that surface in order to speed up processing of surface crossings. The algorithm to build these lists is as follows. First, we loop over all cells in the geometry and count up how many times each surface appears in a specification as bounding a negative half-space and bounding a positive half-space. Two arrays are then allocated for each surface, one that lists each cell that contains the negative half-space of the surface and one that lists each cell that contains the positive half-space of the surface. Another loop is performed over all cells and the neighbor lists are populated for each surface.

References

1. J. F. Briesmeister, Ed., “MCNP - A General Monte Carlo N-Particle Transport Code,
2. Version 4C,” LA-13709-M (April 2000).
3. Dermott E. Cullen , “TART 2002: A Coupled Neutron-Photon, 3-D, Combinatorial Geometry, Time Dependent Monte Carlo Transport Code”, UCRL-ID-126455, Rev. 4, (June 2003), Lawrence Livermore National Laboratory
4. Jaakko Leppänen**, “** Serpent – a Continuous-energy Monte Carlo Reactor Physics Burnup Calculation Code “,User’s Manual, March 6, 2013..Paul K. Romano , Benoit Forget, “The OpenMC Monte Carlo particle transport code”, Annals of Nuclear Energy 51 (2013) 274–281

5. Sutton, T.M. etal., “The MC21 Monte Carlo transport code”. Joint International Topical Meeting on Mathematics and Computation and Supercomputing in Nuclear Applications. Monterey, California., 2007.
6. Criticality benchmark of McCARD Monte Carlo code for light-water-reactor fuel in transportation and storage packages, Nuclear Engineering and Technology 50 (2018)

6. MCkeff Code Validation and Result Analysis

This chapter contains verification and validation procedure adopted for MCk_{eff} code. To start with, the need for criticality code development and the salient features of the code are presented. Firstly, the geometrical features available in the code are tested by choosing appropriate criticality benchmark problems and the physics models are tested by the selection of suitable problems. Finally, applicability of the code for estimating k_{eff} for sub-critical assemblies and to run full core bench mark problem are presented. Further improvements that are possible to make the code versatile, and concluding remarks are given towards the end of this chapter.

6.1. Introduction

With the closure of many experimental facilities, the nuclear criticality safety analyst increasingly is required to rely on computer calculations to identify safe limits for the handling and storage of fissile materials[1]. However, in many cases, the analyst has little experience with the specific codes available at his/her facility. Remember that a single calculation of keff and its associated confidence interval with any code is meaningless without an understanding of the context of the problem, the quality of the solution, and a reasonable idea of what the result should be.

In this chapter an attempt has been to validate and verify the geometry and physics models implemented in the indigenously developed Monte Carlo code, Mckeff, by comparing the results of several complex criticality benchmark problem results with the other well established codes such as MCNP[2] and OpenMC[3]. The MCNP5 version code was chosen for comparison since it has been extensively validated and has thousands of world-wide users, and is relatively stable. The other code, OpenMC is chosen because it is capable of using the same ACE format cross sections as the present code and MCNP5. Therefore, any differences in the results among these codes will be limited to those arising from the geometry and physics algorithms implemented in them. The input file format for MCkeff code is same and it is subset of those valid for MCNP5. This is done to enable the users to run both codes without altering input file. Since the input file format for OpenMC is different, the equivalent inputs are constructed to compute the results. To test the geometrical and physics features incorporated in the MCkeff code, problems are chosen from 'Criticality Monte Carlo calculations: A primer' , since this report was prepared to explain many features of MCNP code geometry and physics models implemented to novice users.

In the first section of this chapter, complex criticality problems are chosen to test the geometrical models incorporated in the code are presented from simple to complex repeated structures. In the subsequent section, criticality problems to test the physics models incorporated such as thermal scattering, S(a,b) and unresolved resonances are described. To test the suitability of code to estimate keff of sub-critical assemblies are given. Further, its utility to run complex full core problems is demonstrated. In the end, problems associated with Monte Carlo code, further improvements to be made in the preset code and conclusions are provided towards the end.

6.2. Salient features of Mckeff Code

The Monte Carlo code 'Mckeff' for estimation of neutron multiplication factor (k_{eff}) of a fissile system has been developed both for windows and linux based operating systems with gnu FORTRAN compiler[4]. This also has been compiled using 'IFORT' compiler[5] under windows 10 operating system. The code has been developed in a modular form using latest FORTRAN 2008 features[6]. The salient features of the code are:

1. The code accepts free format input data in 3 blocks delimited by a blank line for each block
2. Enables building of arbitrary complex configuration of materials in cells. Cells, in turn, are formed by the use of first and second degree signed surfaces numbers denoting half-spaces(positive/negative) combined with binary operators of intersection and union.

3. Code employs continuous energy point wise cross sections for simulations avoiding multi group effects. All major physical interactions between neutron and nuclide are simulated precisely.
4. Thermal motion of nuclides accounted at low neutron energies and crystalline and molecular binding effects are taken care by using S(α,β) tables for some moderating materials and Unresolved resonance tables used if present in the data tables for range of neutron energies between 1 to 100 keV.
5. Extensive input verification and validation done to provide warning and fatal error messages
6. Interactive 2D geometry plotter with zoom in and zoom out option allows close inspection and checking of geometry errors.
7. Plot option exist to view keff versus cycle with error bars,
8. DEBUG facility to trace the random walk of the particle.
9. Volume and mass estimation in cells optionally can be invoked, and
10. Summary of cross sections can be written in a text file 'cxs.txt' optionally through command line argument

The Mck$_{eff}$ code or for that matter any other code is capable only of analyzing the problem as it is specified; it will not necessarily identify inaccurate modeling of the geometry, nor will it know when the wrong material has been specified. Therefore, it is the responsibility of the user to communicate the problem in mind in a language (input file) that is understood by the code.

6.2.1. Output File and Comments

The MCkeff code echoes on the screen whatever is written in the output file. Like MCNP code, the code can be run from 'DOS' prompt '>' by typing 'MCkeff'. In the code, command line interpreter subroutine exist which parses various options given on command line. The code takes 'inp' as default file name (if not specified) and checks for keyword 'out' for output filename and assigns sequentially as 'out-n+1' if already pervious files exist. Fig.1 gives the screen capture image that is displayed by running the code successfully. On the screen some positions with arrow →#n marked are explained in sequel.

→#1. The following line is a command line and the text typed is "MCkeff". Since no options are given, the program assumes input filename as "INP" and no output filename exist, the program searches for the 'out' keyword and finds out1 and out2 files in the current directory and assigns 'out3" for the current output filename.

→#2. If the program is compiled enabling OpenMP commands, it assigns maximum number of threads i.e. 4 here. Each thread executes the part of the code.

→#3. The input file 'INP' echoed with line numbers. There are 9 lines in this input.

→#4. After parsing the input lines, the code may display warning or fatal error messages if present. Next, the code looks for the directory path specified to read cross section data.

→#5. At this position, the program display the 2 cells information those are present in input.

→#6. Surfaces information is displayed. In this problem one surface is given.

→#7. Material information is displayed. In this problem one material is given with 3 nuclides.

→#8. Nuclide cross sections read displayed one after another.

→#9. From the initial source read, the program computes/duplicates the number of source points equal to the number of particles per cycle is required and first cycle simulation continues.

→#10. After running required number of particles per batch/cycle, the program display the keff computed by track length estimator. Since initial guess of fission source is given as the center of the sphere, hence keff is more than one in the first cycle. Subsequently, the fission sites stored in the previous cycle are taken as source points, the keff is less than the previous one. In this input, inactive cycles given as 5 and therefore, the keff's computed are based on each cycle. From the 7^{th} cycle onwards, it will estimates the keff as average over the effective cycles and also gives standard deviation associated with the estimate.

→#11. From the beginning of active cycles, the keff estimated by 3 ways viz. collision, absorption and track length. The average over all the effective cycles is displayed along with the standard deviation. To overcome correlation effects, it is recommended to use combined keff as the answer for the problem.

→#12. Running time statistics are given at the end.

The lines that follow will give the cpu time taken for the problem initialization (includes input processing, cross section data reading), which is 2.688 seconds. Monte Carlo simulation time for 10,000 particles per batch and per 10 batches is 1.391 seconds.

6. MCkeff Code Validation and Result Analysis

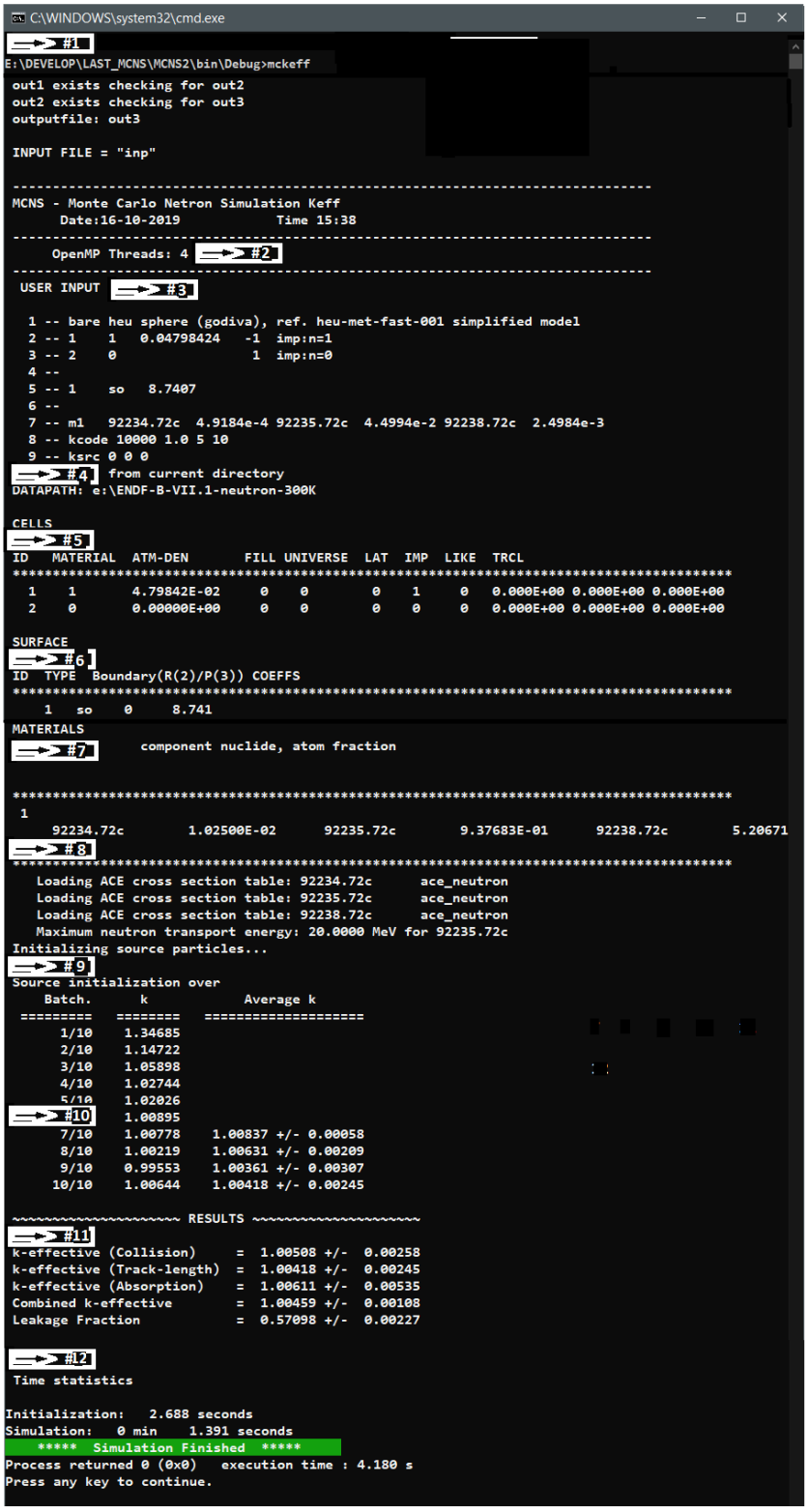


Fig.1 Messages displayed on the screen while executing the MCkeff code.

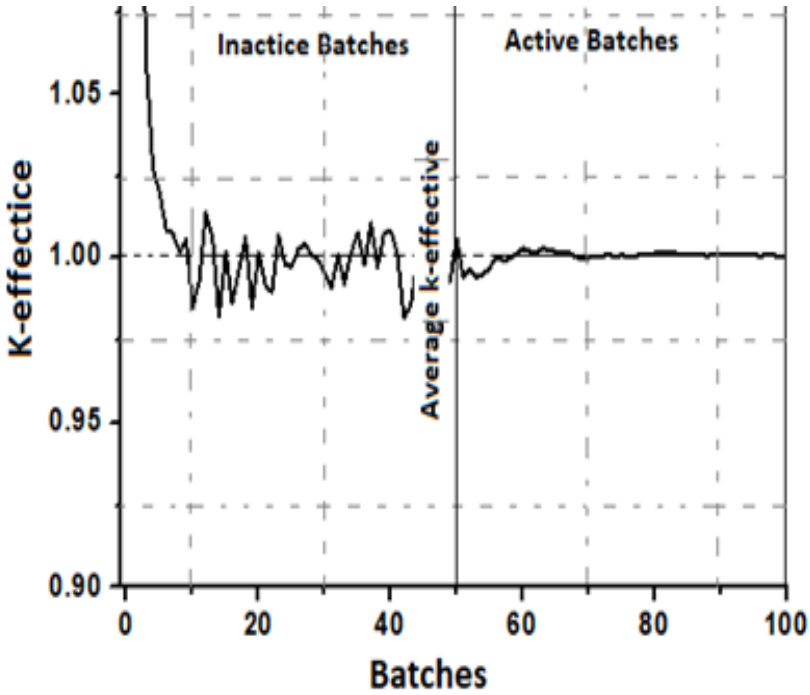


Fig.2. Plot of keff(tracklength) versus cycle number

The above input file when run with 5000 particles per cycle with 50 inactive cycles and total of 100 cycles. The outfile contains all the 3 keff values for each cycle. The track length keff value versus cycle number is plotted in Fig.2. It can be observed from the figure that keff values have a large undulations over the average value till 50 cycles and somewhat smooth beyond. This happens because that initially, the initial source distribution gets corrected based on the materials and geometry of the system. After 50 cycles (assumed fission source distribution has converged), the average value (over the active cycles) of keff is plotted and it is smooth as seen from Fig.2.

6.3. Mckeff code Verification and Validation

Before performing calculations using Monte Carlo codes, verification and validation (V&V) is very important in terms of reliability of the calculational results. Therefore, V&V of MCkeff code has been performed for a variety of geometrical systems and with different physics models incorporated in the code. This practice has been followed for all the codes developed in the past by all the developers. For instance MCNP has been benchmarked against the KENO standard test set as well as experimental results, where available. Similarly, latest Monte Carlo criticality code McCARD code was verified by comparing its results with those of SCALE code for single pin-cell and single assembly benchmark problems. Following the foot print of earlier developers, Mckeff code results are compared with those from MCNP5 and OpenMC. MCNP code is chosen for comparison since it has been extensively validated. The other code, OpenMC is chosen because it is capable of using the same ACE format cross sections as the present code and MCNP5. Therefore, any differences in results between these codes will be limited to those arising from the geometry and physics algorithms implemented in the codes.

Firstly criticality problems to check the correctness of geometrical models incorporated are presented from simple to complex repeated structures. secondly, the criticality problems to test the physics models incorporated such as thermal scattering, S(α,β) and unresolved resonances are described. To meet these two objectives, the range of criticality problems depicted in the manual on ‘ criticality calculation with Monte Carlo: A Primer’ is highly suited, since these problems are meant for explaining the features of MCNP code, covering both geometrical features and physics models incorporated in this code. However, we have chosen few representative problems with increasing complexity of geometrical situations. The problem description, the corresponding Input file, 2D-geometry plot side-by-side and the results are discussed in sequel. The results are recomputed using

MCNP code using the same input file as that used by Mckeff code. The equivalent input files required to run OpenMC code are prepared and the results are generated. It should be noted here that the MCNP uses implicit capture as biasing technique, where as other two code OpenMC and MCkeff do not use any biasing technique.

6.3.1. Test of geometrical models implemented in Mckeff code

All input and output files of MCkeff and MCNP codes are included in the project distribution DVD. This DVD also contains full documentation of the project.

6.3.1.1. EXAMPLE: BARE PU SPHERE

We start with a simple benchmark problem of a sphere. This problem is a bare sphere of plutonium metal of diameter 12.77 cm with a coating of nickel shell of thickness 0.0127 cm (also known as Jezebel reactor). The input file is given in the text box below and 2D view of geometry is shown in Fig.6.1. Since the Nickel coating is very thin, its view is shown in adjoining figure. Adequate explanation can be found in the reference[1] about the sequence of input line data and keywords and its required data.

```
Example 6.1: Bare Pu sphere w/ Ni shell Jezebel simple model.

C Cell cards
1 1 4.0290e-2  -1     imp:n=1    $ Pu sphere inside
2 2 9.1322e-2   1 -2  imp:n=1    $ Thin Nickel shell
3 0 2                 imp:n=0    $  Outside sphere

C Surface cards
1 so 6.38493
2 so 6.39763

C Data cards
kcode 5000 1.0 50 250
ksrc 0 0 0
m1 94239.72c 3.7047e-2 94240.72c 1.751e-3   94241.72c 1.17e-4
     31069.72c 1.375e-3
m2 28058.72c 0.6808 28060.72c 0.2622   28061.72c 0.0114 28062.72c 0.0363
     28064.72c 0.0093
```

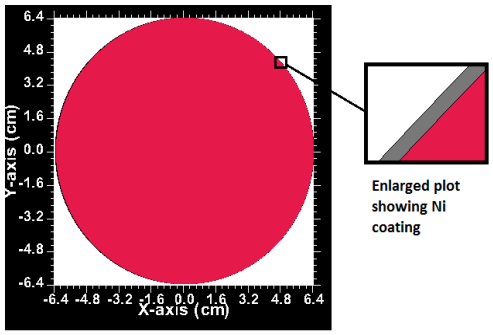


Fig.6.1 Jezebel Pu Sphere with Ni coating

This problem was run with 5000 particles per cycle over 250 cycles skipping 50 cycles before accumulating the results. The MCNP and the present code combined keff values are 1.00130±0.00063, 0.99981±0.00060 respectively. The agreement between them is very good and the values are with in ±1σ confidence interval.

6.3.1.2. EXAMPLE: REFLECTED PU CYLINDER

This problem is chosen to test the 2 dimensional cylindrical geometry capability of Mckeff code. This criticality bench mark problem consists of plutonium metal cylinder of diameter 9.87cm and of height 6.909 cm reflected on all sides with 5cm thick shell of natural uranium. The input file used for MCNP and MCkeff codes is same and it is given below in the text boxes. Horizontal and vertical sectional view of the geometry is shown in Fig.6.2

This problem was run with 5000 particles per cycle over 250 cycles skipping 50 cycles before accumulating the results. The MCNP and the present code combined keff values are 1.02520±0.00068, 1.02663±0.00077 respectively. The agreement between them is very good and the values are with in ±1σ confidence interval.

```
Example 6.2. Pu cylinder, all sides nat.U reflector
C Cell Cards
1 1 -15.8      -1 2 -3          imp:n=1    $ pu cylinder
2 2 -18.8      -4 -6 5 #1       imp:n=1    $ u-nat reflector
3 0            4:-5:6           imp:n=0

C Surface Cards
1 cx 4.935
2 px 0.0
3 px 6.909
4 cx 9.935
5 px -5.0
6 px 11.909

C Data Cards
m1 94239.72c 1.0
m2 92238.72c 0.992745 92235.72c 0.007200
C Criticality Control Cards
kcode 5000 1.0 50 250
ksrc 3.5 0 0
```

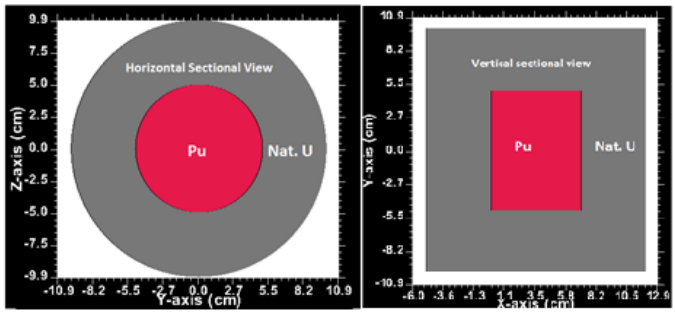


Fig.6.2. Pu 2D Cylinder reflected with 5cm natural U

6.3.1.3. EXAMPLE: RECTANGULAR LATTICE OF 3x2 PU CYLINDERS

This example is chosen to demonstrate the capability of handling rectangular lattice by MCkeff code. The problem consists of partially filled Pu cylindrical containers. The input file is same as that given for 6.3.1.4. The differences are explained here. The cell card 5 shown as

5 0 -8 9 -10 11 lat=1 fill=1 u=2 imp:n=1 $ lattice

Defines cell of 35.58x35.58 infinite in Z-direction filled with universe 1 i.e. Pu solution container. The number of lattice elements are undefined (infinite in all directions). To limit these elements to a finite number, it is kept in a window, which is defined in cell 6 as

6 0 91 -121 111 -141 3 -4 fill=2 imp:n=1 $ window

This cell dimensions are 106.74x71.16 which limits the lattice elements 3 in x-direction and 2 in y-direction. The plots of this input file is shown in Fig.6.3.

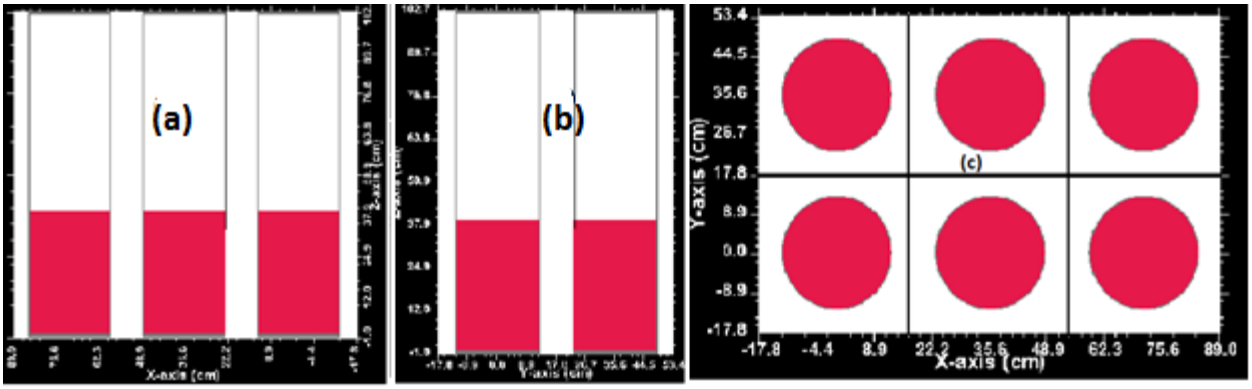


Fig.6.3. Plots of Pu solution of partially filled cylinders of 3x2 lattice (a)XZ view (b)YZ view (c) XY view

This problem was run with 5000 particles per cycle over 250 cycles skipping 50 cycles before accumulating the results. The source points are given one in each at the center of the cylinder. The MCNP and the present code combined keff values are 0.99126±0.00115, 0.99462±0.00103, respectively. The agreement between them is very good and the values are with in ±1σ confidence interval.

6.3.1.4. EXAMPLE: CHANGE IN MATERERIALS OF LATTICE ELEMENTS

This problem is also rectangular lattice of 3x2 same as that of example 6.3.1.3. However, filling universes are different at two locations. The input file is given in the text boxes below.
Similar to that discussed in example 6.3.1.3, here the lattice cell 5 is

```
Example 6.4, Hexahedral Lattices(3x2). Lattice with 1 empty
element.
C Cell Cards
1 1 9.9270e-2 -1 5 -6 u=1 imp:n=1 $Pu Soln.
2 0 -1 6 -7 u=1 imp:n=1 $ void above Pu Soln.
3 2 8.6360e-2 -2 #1 #2 u=1 imp:n=1 $ SS
4 0 2 u=1 imp:n=1 $ void
5 0 -8 9 -10 11 lat=1 fill=0:2 0:1 0:0
     2 1 3 1 3 1 u=2 imp:n=1 $ lattice
6 0 9 -12 11 -14 3 -4 fill=2 imp:n=1 $ window
7 0 -9:12:-11:14:-3:4 imp:n=0 $ outside
11 like 1 but mat=3 rho=-1.60 u=3 imp:n=1
12 like 2 but u=3 imp:n=1
13 like 3 but u=3 imp:n=1
14 like 4 but u=3 imp:n=1

C Solution Cylinder
1 cz 12.49
2 cz 12.79
5 pz 0.0
6 pz 39.24
7 pz 101.7
C Beginning of lattice surfaces
8 px 17.79
9 px -17.79
10 py 17.79
```

```
C Beginning of Window Surfaces
3 pz -1.0
4 pz 102.7
12 px 88.95
14 py 53.369

C Data Cards
kcode 5000 1.0 150 550
c one source point in each volume of Pu Solution
ksrc 0 0 19.62 35.58 0 19.62 71.16 0 19.62
     0 35.58 19.62 35.58 35.58 19.62 71.16 35.58 19.62
C Material cards
m1 1001.72c 6.0070-2 8016.72c 3.6540-2 7014.72c 2.3699-3
     94239.72c 2.7682-4 94240.72c 1.2214-5 94241.72c 8.3390-7
     94242.72c 4.5800-8
c mt1 lwtr.60t
m2 24050.72c 7.195-4 24052.72c 1.38589-2 24053.72c 1.5713-3
     24054.72c 3.903-4
     26056.72c 3.704-3 26056.72c 5.80869-2 26057.72c 1.342-3
     26058.72c 1.773-4
     28058.72c 4.432-3 28060.72c 1.7069-3 28061.72c 7.42-5
     28062.72c 2.363-4 28064.72c 6.05-5
m3 6000.72c 1
c mt3 grph.60t
```

5 0 -8 9 -10 11 lat=1 fill=0:2 0:1 0:0 2 1 3 1 3 1 u=2 $ lattice.

Here the fill pattern is defined as ‘2 1 3 1 3 1’ unlike the previous one where all fill elements are 1. These numbers corresponds to the labels of universes given in the input of cell cards. The universe 1 is partially filled Pu solution cylinder (red), universe 2 is the lattice element itself ‘Void’ (white)

and the universe 3 is graphite cylinder(yellow) which are shown below in the Fig.6.4. The widow cell is 6 which limits the number of lattice elements to be filled and which is cell 6 and it is given below.

6 0 9 -12 11 -14 3 -4 fill=2 imp:n=1 $ window

The plots of this example is shown in Fig.6.4.

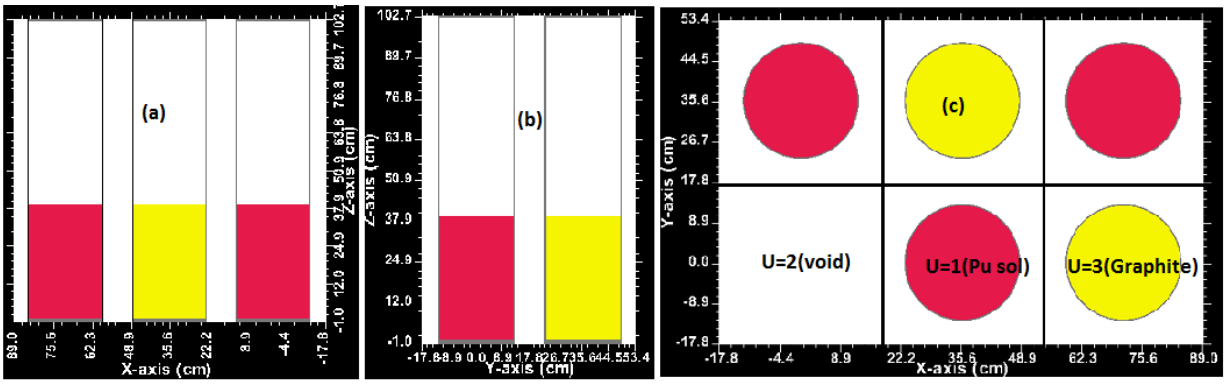


Fig.6.4. Pu sol. cylinders 2x3 lattice with change of materials in 2 elements (a) XZ view (b) YZ view (c)XY view

This problem was run with 5000 particles per cycle over 550 cycles skipping 150 cycles before accumulating the results. The MCNP and the present code combined keff values are 0.91436±0.00068, 0.91206±0.00072, respectively. The agreement between them is very good and the values are with in ±1σ confidence interval.

6.3.1.5. EXAMPLE: 3D LATTICES (3x2x2) OF PU CYLINDERS

This problem is chosen to demonstrate the capability of handling 3D lattices by Mckeff code. This input is similar to that given in example 6.3.1.4. except that it has 2 levels in the Z-direction.For this example cells 1 and 2 define the cylindrical region of Pu solution and the void above respectively. Cell 3 is the stainless steel container and cell 4 is everything outside. All these form a universe 1 and it is filled in a lattice cell 5. And cell 6 is a window cell and filled with 3x2x2 lattice elements. The input file is shown below and the plots of XZ,YZ and XY views are given in the Fig.6.5.

The 3D lattice cell "**5"** is infinite in all directions filled with universe 2, and is defined as below,

5 0 -8 9 -10 11 -16 3 lat=1 u=2 fill=1 imp:n=1

This cell dimensions are "106.74x71.16x226.69" which limits the lattice elements 3 in x-direction and 2 in y-direction and 2 in z-direction.

6 0 91 -12 111 -14 181 -161 fill=2 imp:n=1

The plots of this input file is shown in Fig.6.5.

```
Example 7-3, 3-D (3x2x2) Lattice
C Cell Cards
1 1 9.9270e-2     -1 5 -6 u=1 imp:n=1
2 0               -1 6 -7 u=1 imp:n=1
3 2 8.6360e-2     -2 -4 #1 #2 u=1 imp:n=1
4 0 2:4 u=1 imp:n=1
5 0 -8 9 -10 11 -16 3 lat=1 u=2 fill=1 imp:n=1
6 0 91 -12 111 -14 181 -161 fill=2 imp:n=1
7 0 -91:12:-111:14:-181:161 imp:n=0

C Solution Cylinder Surface Cards
1 cz 12.49 $ Inner cylinder
2 cz 12.79 $ Outer SS cylinder
4 pz 102.7 $ Top of SS tank
5 pz 0.0 $ Bottmon of Solution
6 pz 39.24 $ Top of Solution
7 pz 101.7 $ Top of void above soln.
C Beginning of Lattice Surfaces
8 px 17.79
9 px -17.79
91 px -17.78
10 py 17.79
11 py -17.79
111 py -17.78
C Window Surfaces
3 pz -1.0
12 px 88.95
14 py 53.37
```

```
C Surfaces 16 and 18 bound the lattice in the z direction
16 pz 112.7 $ Top of lattice cell
161 pz 112.69 $ Top of lattice cell
18 pz -114.7 $ Bottom of lattice cell
181 pz -114.69 $ Bottom of lattice cell

C Data Cards
C Material Cards
m1 1001.72c 6.0070-2 8016.72c 3.6540-2
     7014.72c 2.3699-3
     94239.72c 2.7682-4 94240.72c 1.2214-5
     94241.72c 8.3390-7 94242.72c 4.5800-8
mt1 hh2o.71t
m2 24050.72c 7.195-4 24052.72c 1.38589-2
     24053.72c 1.5713-3 24054.72c 3.903-4
     26056.72c 3.704-3 26056.72c 5.80869-2
     26057.72c 1.342-3 26058.72c 1.773-4
     28058.72c 4.432-3 28060.72c 1.7069-3
     28061.72c 7.42-5 28062.72c 2.363-4
     28064.72c 6.05-5
C Criticality Control Cards
kcode 5000 1.0 50 250
ksrc 0 0 19.62 35.58 0 19.62 71.16 0 19.62
     0 35.58 19.92 35.58 35.58 19.62 71.16 35.58 19.62
C These source points are place in the added cylinders
     0 0 -94.08 35.58 0 -94.08 71.16 0 -94.08
     0 35.58 -94.08 35.58 35.58 -94.08 71.16 35.58 -94.08
```

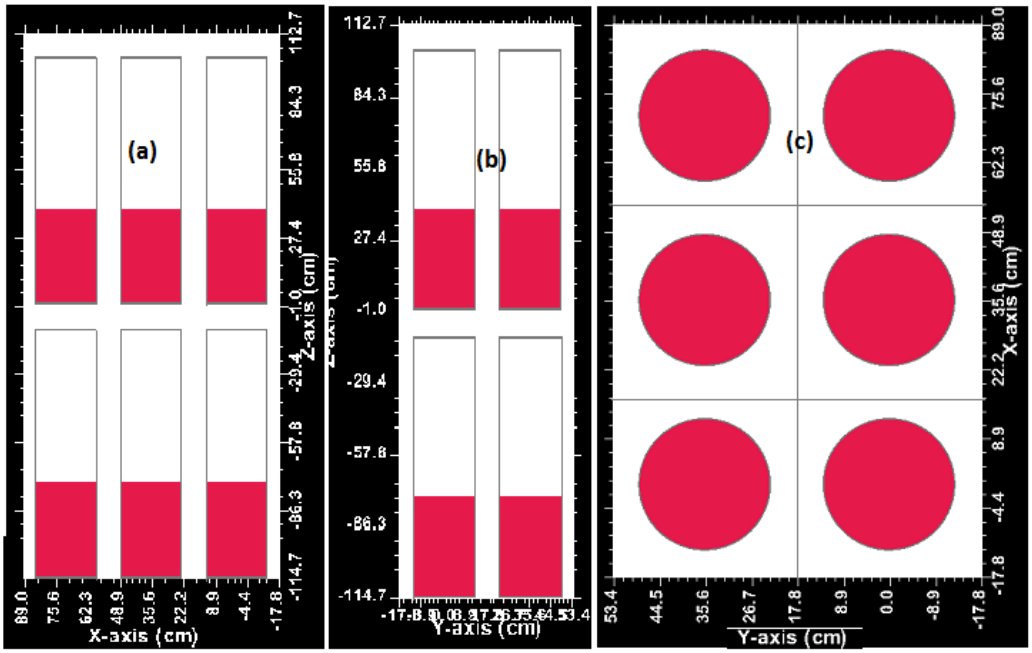


Fig.6.5. Plots of 3D lattice (3x2x2) (a) XY view (b) YZ view (c) YX view

This problem was run with 5000 particles per cycle over 250 cycles skipping 50 cycles before accumulating the results. The MCNP and the present code combined keff values are 0.99661±0.00101, 1.00137± 0.00111 respectively. The agreement between them is very good and the values are with in ±1σ confidence interval.

6.3.1.6. EXAMPLE: HEXAGONAL LATTICE OF PU CYLINDERS

This is a hexagonal array of seven open U(93.2% enrichment)O_2F_2 solution cylinders (LA-10860 page 125). The uranium-fluoride solution is contained in seven aluminum cylinders with a 7.60 cm surface separation between cylinders. There is 20 cm of water reflection below and radially about the cylinders. There is no water above the aluminum containers. Opposite sides of the six-sided lattice cell must be equal in length and parallel. The dimension of concern that determines the six surfaces of the lattice cell is the pitch of the solution containers. The pitch for this example is calculated by adding the outer diameter of the aluminum container and the surface separation.

The data for this problem are:

```
Example 6-6: Hexagonal Lattice of U Cylinders
C Cell Cards
1 1 9.8983e-2 -1 5 -4 u=1 imp:n=1
2 0 -1 4 u=1 imp:n=1
3 2 -2.7 -2 (1:-5) u=1 imp:n=1
4 3 -1.0 2 u=1 imp:n=1
5 0 -8 11 -7 10 -12 9 lat=2 fill=1 u=2 imp:n=1
6 0 -13 6 -3 fill=2 imp:n=1
7 3 -1.0 (13:-6) -3 15 -14 imp:n=1
8 0 14:3:-15 imp:n=0

C Solution Cylinder Surface Cards
1 cz 7.60 $ outer radius of the solution
2 cz 7.75 $ outer radius of container
4 pz 23.4 $ top of solution
5 pz 0.0 $ bottom of solution
C Surfaces 7-12 are the array lattice cell
7 p 1 1.73205 0 23.1
8 px 11.55
9 p -1 1.73205 0 -23.1
10 p 1 1.73205 0 -23.1
```

```
11 px -11.55
12 p -1 1.73205 0 23.1
C Window Surfaces
3 pz 40.0 $ Top of aluminum cylinder
6 pz -1.0 $ bottom of aluminum container
13 cz 32.0 $ cylinder for array window
C Reflector Surfaces
14 cz 52.0 $ outer radius of reflector
15 pz -21.0 $ botmmon edge of reflector

C Data Cards
C Material Cards
m1 1001.72c 6.1063-2 8016.72c 3.3487-2 9019.72c 2.9554-3
     92235.72c 1.3784-3 92238.72c 9.9300-5
m2 13027.72c 1.0
m3 1001.72c 2 8016.72c 1
C Criticality Control Cards
kcode 5000 1.0 50 250
ksrc 0 0 11.7 -23.1 0 11.7 23.1 0 11.7
     -11.55 20.0 11.7 -11.5 -20.0 11.7
     11.55 20.0 11.7 11.5 -20.0 11.7
```

The geometry plot of this example is shown in Fig.6.6.

This problem was run with 5000 particles per cycle over 250 cycles skipping 50 cycles before accumulating the results. The MCNP and the present code combined keff values are 0.0114±0.0001,0.0118±0.0002, respectively. The agreement between them is very good and the values are with in ±1σ confidence interval.

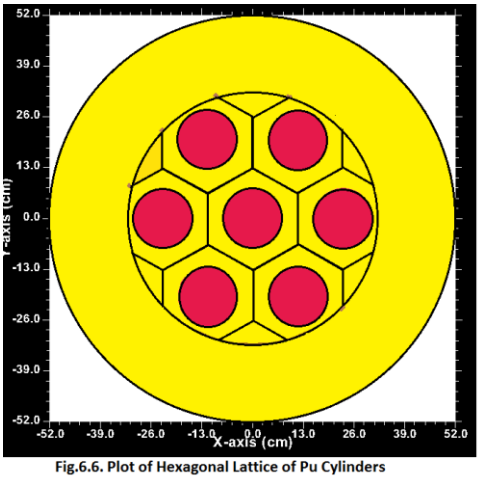


Fig.6.6. Plot of Hexagonal Lattice of Pu Cylinders

6.3.2. Test of physics models implemented in Mckeff code

This section presents the physics models implemented in the Mckeff code. There are two essential aspects to be taken care of in Monte Carlo simulation. They are thermal motion of target nuclides and unresolved resonance effects. Without addressing these problems, for some situations, the Monte Carlo simulation results might be misleading. The following text addresses these issues by selecting the appropriate benchmark criticality problems.

6.3.2.1. S(α,β) Thermal Neutron Scattering Laws for Moderators

This criticality benchmark problem chosen to demonstrate two aspects of MCkeff namely, thermal scattering laws and reflective boundary conditions. In the input textbox under surface cards, surface number from 2 to 5, which stands for reflective boundary condition. For problems that have a thermal spectrum such as a commercial light-water reactor, it is essential to accurately treat the scattering of neutrons from bound scatterers such as hydrogen in water. The S(α,β) thermal scattering treatment is a complete representation of thermal neutron scattering by molecules and crystalline solids. Two processes are allowed: (1) inelastic scattering with cross section σ_{in} and a coupled energy-angle representation derived from an ENDF S(α,β) scattering law, and (2) elastic scattering with no change in the outgoing neutron energy for solids with cross section σ_{el} and an angular treatment derived from lattice parameters. Cullen et al., 2004 had proposed a simple pin cell model which probably may highlight the differences in answers due to variety of bound scattering treatment. This problem consists of an infinite lattice of fuel pins with a 2 in. pitch and varying fuel pin radii(1/2, ¼. 1/8 in). The fuel consists of only two nuclides, U-235 and U-238, and furthermore there is no fuel cladding or gap. The water is not borated, i.e. it consists only of hydrogen and oxygen. By having a very simple model, any differences in answers can be almost solely attributed to the bound-scattering treatment.

All of the 6 sets of pin cell problems with and without S(a,b)) have been run with three different codes namely Mckeff, MCNP5, OpenMC using same cross section libraries and similar physics treatments and criticality control parameters. The text box below contains the input file of MCNP and MCkeff codes.

```
Cullen S(a,b) Benchmark Case 1: 1/2" Pin
1      1   -18.8    -1              imp:n=1
2      2    -1.0     1 2 -3 4 -5    imp:n=1
3      0             -2:3:-4:5      imp:n=0

1     cz    1.27
*2    px   -2.54
*3    px    2.54
*4    py   -2.54
*5    py    2.54

m1    92238.72c   0.9902   92235.72c
0.0098
m2    1001.72c     2.0       8016.72c
1.0
mt2   hh2o.71t
kcode 10000 1.00 50 4050
ksrc  0.0 0.0 0.0
```

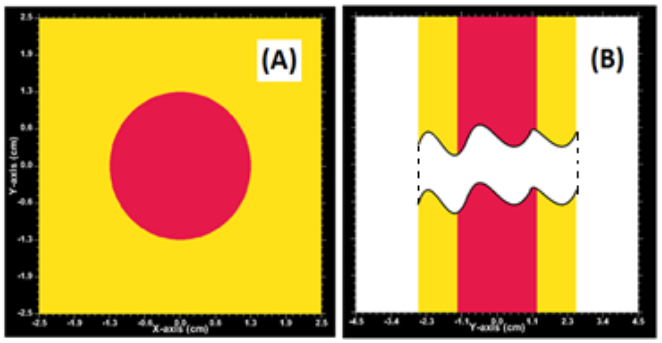


Fig 6.7. ½ inch pin cell 2D plot (Benchmark - indc-usa-107), (A) XY plot, (B) YZ plot (z-infinite)

Table 6.1. shows k-effective and its standard deviation for the six cases described above.

Case	MCk_{eff}	OpenMC-0.8.0	MCNP
1/2″ pin, no *S*(*α*, *β*)	1.01710 +/- 0.00004	1.01703 +/- 0.00004	1.01638 +/- 0.00004
1/2″ pin, *S*(*α*, *β*)	0.96822 +/- 0.00004	0.96815 +/- 0.00004	0.96820 +/- 0.00004
1/4″ pin, no *S*(*α*, *β*)	1.01519 +/- 0.00005	1.01533 +/- 0.00005	1.01329 +/- 0.00005
1/4″ pin, *S*(*α*, *β*)	0.92228 +/- 0.00005	0.92227 +/- 0.00005	0.92214 +/- 0.00006
1/8″ pin, no *S*(*α*, *β*)	1.01780 +/- 0.00005	1.01769 +/- 0.00005	1.01329 +/- 0.00007
1/8″ pin, *S*(*α*, *β*)	0.90942 +/- 0.00005	0.90943 +/- 0.00005	0.90937 +/- 0.00007

Table 6.1. shows k-effective and its standard deviation for the six cases described above. It should be noted that the uncertainties reported here were calculated assuming no correlation between successive batches and therefore may be under predicted. From the above table it is clear that the results from the MCk$_{eff}$ is in good agreement with a difference less than 170 pcm with respect to OpenMC and MCNP. The values can be made closer by running the OpenMC ans MCkeff with more number of cycles, since these two does not use any biasing technique.

6.3.2.2. Unresolved resonance treatment

Besides thermal scattering, the other energy range that requires special treatment is the unresolved resonance range. For many nuclides, resonances in the 1–100 keV energy range are so narrow and closely spaced that is not possible to experimentally resolve the details of all resonances. For problems that are sensitive to the unresolved resonance range, not accounting these effects can result in serious errors in reported answers. We have chosen to compare results on a model of the Big Ten critical assembly (IEU-MET-FAST-007 from the International Handbook of Evaluated Criticality Safety Benchmark Experiments (NEA Nuclear Science Committee, 2009) as a means of validating the implementation of the probability table method in MCkeff. This assembly is a large, mixed uranium metal cylindrical core of 10% enrichment surrounded by a U-238 reflector. The geometry plots are given in Fig.6.8.

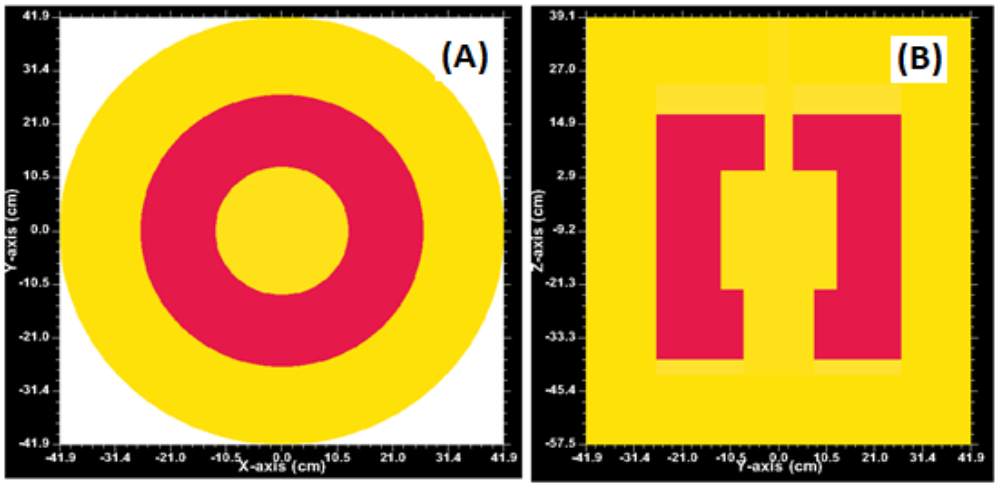


Fig 6.8. Big Ten IEU-MET-FAST-007 Improved Benchmark Case 4

The input file for this problem is given in the textbox below.

```
Big Ten IEU-MET-FAST-007 Improved Benchmark Case 4
1    2    0.047609     -3   8 -10                $ bottom 10% cylinder
2    2    0.047609     -4  10 -11                $ lower 10% cylinder
3    2    0.047609     -2  11 -13                $ upper 10% cylinder
4    2    0.047609     -1  13 -14                $ top 10% cylinder
5    3    0.048196      3  -5   8  -9            $ bottom natural U plate
6    1    0.048159      2  -5   9 -12 #1 #2 #3  $ homogenized HEU + natural U
7    3    0.048196      2  -5  12 -13            $ top natural U plate
8    4    0.047779      5  -6   7 -14            $ DU outer annulus
9    4    0.047779     -5   7  -8                $ DU bottom annulus
10   4    0.047779      1  -5  13 -14            $ DU top annulus
11   0                   (6:-7:14)

1     cz    2.25014        $ outer radius of top 10% cylinder
2     cz    3.10996        $ outer radius of upper 10% cylinder
3     cz    7.62000        $ outer radius of bottom 10% cylinder
4     cz   12.54604        $ outer radius of lower 10% cylinder
5     cz   26.67000        $ inner radius of DU reflector annulus
6     cz   41.91000        $ outer radius of DU reflector annulus
7     pz  -57.46750        $ bottom of DU reflector
8     pz  -41.73361        $ top of DU reflector bottom
9     pz  -38.24644        $ bottom of homogenized HEU and natural U
10    pz  -22.39010        $ bottom of bottom 10% cyliner
11    pz    4.35102        $ bottom of lower 10% cylinder
12    pz   17.16665        $ top of homogenized HEU and natural U
13    pz   23.81250        $ bottom of DU reflector top
14    pz   39.05250        $ top of DU reflector

imp:n 1 9r 0                  $     Set importance for all cells to one
m1    92234.72c 5.4058e-5   $     Homogenized HEU, Natural U, and Voids
      92235.72c 4.9831e-3 92236.72c 1.3733e-5 92238.72c 4.3108e-2
m2    92234.72c 2.4761e-5   $     Intermediate Enriched Uranium (10 wt%)
      92235.72c 4.8461e-3 92236.72c 4.3348e-5 92238.72c 4.2695e-2
m3    92234.72c 2.6518e-6   $     Natural Uranium
      92235.72c 3.4701e-4 92238.72c 4.7846e-2
m4    92234.72c 2.8672e-7   $     Depleted Uranium
      92235.72c 1.0058e-4  92236.72c 1.1468e-6 92238.72c 4.7677e-2
kcode 10000 1.0 100 3000
```

The Big Ten benchmark was run in MCNP5 and OpenMC using the same ENDF/B-VII.I cross section libraries. Again, each MCNP run had 50 inactive 1000 active batches, each consisting of 10,000 particles, whereas the OpenMC run had 4000 active batches. Table 6.2. shows k-effective and its standard deviation for all runs. The results clearly show that the unresolved resonance probability table treatment in MCkeff has been implemented correctly, with results agreeing with other two code values with less than 100 pcm in reactivity.

Table 6.2. Effective multiplication factor for Big Ten benchmark from MCNP expanded criticality validation suite. (Benchmark – IEU-MET-FAST-007)

Ptable	MCkeff	OpenMC	MCNP
on	1.00473 +/- 0.00011	1.00460 +/- 0.00011	1.00472 +/- 0.00017
off	1.00052 +/- 0.00011	1.00086 +/- 0.00011	1.00060 +/- 0.00017

6.4. Full-core problems

It is desirable to also compare results on a benchmark with complicated geometry and materials. One such benchmark problem is the Monte Carlo Performance Benchmark originally proposed by Hoogenboom et al. (2011). The specific aim of this benchmark is to monitor the increase in performance of Monte Carlo calculations of full-core reactor problems. The model consists of a typical PWR core layout with 241 fuel assemblies, each with a 17 by 17 lattice of fuel pins including 24 control rod guide tubes and an instrumentation tube. The fuel is composed of 34 different nuclides: a mix of actinides, minor actinides, and key fission products. A model of the Monte Carlo Performance Benchmark input file was taken from reference 4. Fig. 6.9 shows the layout of the ¼ th core (A), fuel assembly within a core (B) including guide tubes and the fuel pin(C). To get an estimate of the effective multiplication factor, the MCNP model was run with
100,000 particles per cycle for 150 inactive and 1000 active batches. Mckeff and OpenMC were run with 100,000 particles per cycle, 150 inactive batches, and 4000 active batches, since these two codes do not use any biasing techniques, using the same ENDF/B-VII.1 libraries. Table 3 shows the effective multiplication factors and their standard deviations and a agreement was found among these codes.

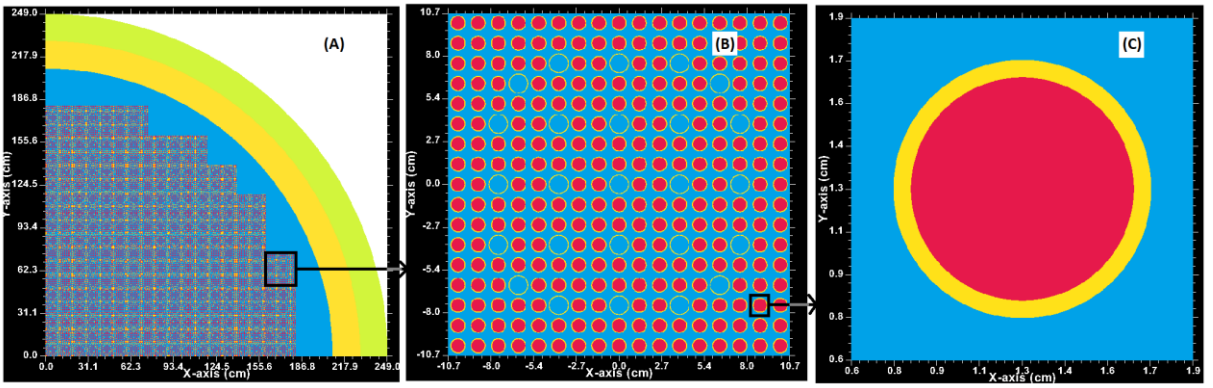


Fig. 6.9 shows the layout of the ¼ th core (A), fuel assembly within a core (B) including guide tubes and the fuel pin(C).

Table 6.3. keff of full core problem

Performance benchmark	keff
MCkeff	1.00088 +/- 0.00004
OpenMC	1.00105 +/- 0.00004
MCNP	1.00103 +/- 0.00007

From the above table it is clear that the keff value of MCkeff is in good agreement with other codes with a minimal difference of less than 100 pcm in reactivity.

6.5. Sub-critical assembly problem

Fuel storage facilities and fuel reprocessing vessels are always maintained the keff value well below the 1. Most often, it is required to predict keff values precisely for these configurations also. Here, a typical bird cage input is shown in the text box below. The 2D plot of bird cage problem is shown in Fig.6.10.

INPUT FILE

```
C   Single isolated bird cage model
C   cell cards
1    5 -2.0 1 -25 -5
2    1 -1.205E-3 25 -2  -5
3    3  -8.00     (-1  3  -5):( 2  -4   -5 ):(5  -6   3  -4)
4    1  -1.205E-3  (-3  7  -6):( 4  -8   -6 ):(6  -11  7  -8)
5    4  -1.406     (-7  9  -11):(8  -10  -11):(11 -12  9 -10)
6    1  -1.205E-3  (-9  13 -12):(10 -14  -12):(12 -17 13 -14)
7    4  -1.406     (-13 15 -17):(14 -16  -17):(17 -18 15 -16)
8    1  -1.205E-3  (-15 19 -23):(16 -20  -23):(18 -23 19 -20)
9    3  -8.00      (-19 21 -23):(20 -22  -23):(23 -24 21 -22)
10   2  -0.998207  (-100 24):(-100 22):(-100 -21)
11   0 100

c    surface cards
1    pz     -7.000
2    pz      7.000
3    pz     -7.075
4    pz      7.075
5    cz      6.000
6    cz      6.075
7    pz     -7.865
8    pz      7.865
9    pz     -7.875
10   pz      7.875
11   cz      6.465
12   cz      6.475
13   pz     -8.665
14   pz      8.665
15   pz     -8.675
16   pz      8.675
17   cz      6.865
18   cz      6.875
19   pz     -9.000
20   pz      9.000
21   pz     -9.150
22   pz      9.150
23   cz      7.000
24   cz      7.150
25   pz      1.842
100  SO      100.0

c   data cards
imp:n 1 9r 0
m1   6000.72c  -0.000124   7014.72c  -0.755268   8016.72c  -
0.231781
     18040.72c -0.012827        $ Air COmposition 1.205E-3 g/cc
m2   1001.72c  -0.111894   8016.72c  -0.888106 $ Water
m3   6000.72c  -0.000150   14028.72c -0.005000   15031.72c -
0.000230
     16032.72c -0.000150   24052.72c -0.190000   25055.72c -
0.010000
     26056.72c -0.694480   28058.72c -0.100000   $ SS 304L
Composition
m4   1001.72c  -0.048382   6000.72c  -0.384361   17035.72c -
0.567257 $ PVC
m5   8016.72c  -0.118080   94239.72c -0.652621   94240.72c -
0.185203
     94241.72c -0.035277   94242.72c -0.007055 $ PuO2
Composition 2.0 g/cc
kcode  5000 1.0 50 100
ksrc        0.0 0.0 -2.579
```

This problem had run with a total of 100 batches each with 5000 particles skipping first 50 batches.

.

Table 6.4. keff comparison of sub-critical assembly

	keff
MCNP	0.32155 +/- 0.00123
MCkeff	0.32242 +/- 0.00144

From the above table it is clear that the keff value of MCkeff is in good agreement with MCNP with a minimal difference of less than 100 pcm in reactivity.

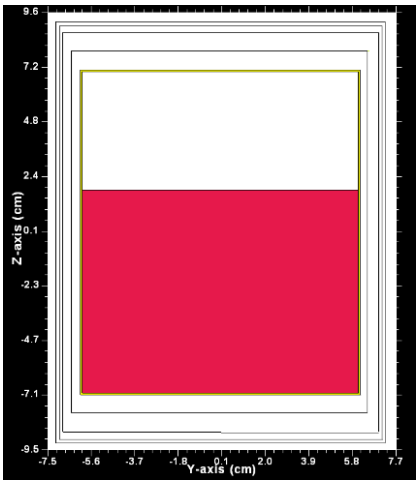


Fig. 6.10. Plot of typical single isolated bird cage to store PuO2 powder.

6.5.1. More Comparison Tests

Tables 6.5 to 6.8 shows the comparison of keff results obtained with MCkeff and other standard codes. These problems were run with different KCODE control parameters such as number of

neutrons per cycle, number of inactive cycles and the total number of cycles. In all cases combined keff values are presented along with the standard deviation. This lends further confidence on the implantation of various geometry and collision physics modules in the MCkeff code. Table 6.5 shows comparison results of Various geometrical configurations such as cylinders, lattices with various combinations**.** The input files are provided in the CD under subfolder .inputs/primer and the outputs are given in the same CD under the subfolder .outp/primer.

Table 6.5. Combined K_{eff} comparison of MCNP criticality primer [1] inputs with Plutonium fuel for various simple geometrical configurations.

S.No	File	Title	MCNS*	MCNP
1	Inp01	Example 1-2a. Jezebel simple model. Bare Pu sphere w/ Ni shell	0.99981 +/- 0.00060	1.00130 +/- 0.00063
2	Inp02	Example 2-3. Bare Pu Cylinder	1.01577 +/- 0.00069	1.01439 +/- 0.00062
3	Inp03	Example 2-4. Pu cylinder, radial U(nat) reflector	0.88640 +/- 0.00062	0.88611 +/- 0.00057
4	Inp04	Example 2-5. Pu cylinder, radial U(nat) reflector	1.02663 +/- 0.00077	1.02520 +/- 0.00068
5	Inp05	Example 4-3, Repeated Structures: Two Cylinders	1.05971 +/- 0.00107	1.05937 +/- 0.00091
6	Inp06	Example 5-3, Square Lattice of 3x2 Pu Cylinders	0.99462 +/- 0.00103	0.99126 +/- 0.00115
7	Inp07	Example 5-4, Hexahedral Lattices. Change in Material in 2 elements.	0.94478 +/- 0.00100	0.94742 +/- 0.00097
8	Inp08	Example 5-5, Hexahedral Lattices. Lattice with 1 empty element.	0.91206 +/- 0.00072	0.91436 +/- 0.00068
9	Inp09	Example 5-6, Hexahedral Lattices. Different Fill Cell Size	0.93945 +/- 0.00095	0.93861 +/- 0.00097
10	Inp10	Example 7-3, 3-D (3x2x2) Lattice	1.00137 +/- 0.00111	0.99601 +/- 0.00101
11	Inp11	Example 7-9, 3-D Lattice with one water element.	0.99316 +/- 0.00098	0.99588 +/- 0.00095

*All Values obtained from MCkeff are found to be deviating less than 1% with respect to MCNP.

The table 6.6. presents the comparison results with the other two Monte Carlo codes. These problems are chosen to test MCkeff code capability in handling different fuel compositions**.** The input files are provided in the CD under subfolder .inputs/fuels and the outputs are given in the same CD under the subfolder .outp/fuels.

Table 6.6. Initial Combined K_{eff} comparison of few criticality benchmark [4] problems using MCkeff with MCNP and OpenMC.

S.No	File	Title	MCkeff*	OpenMC	MCNP
1	Inp01	Reflected U-Hydride critical assembly, HEU-COMP-INTER-003 case 6	0.99700 +/- 0.00127	0.99613 +/- 0.00017	0.99736 +/- 0.00109
2	Inp02	Bare HEU sphere (Godiva), ref. HEU-MET-FAST-001 detailed model	1.00032 +/- 0.00054	0.99988 +/- 0.00012	0.99940 +/- 0.00039
3	Inp03	Topsy Oralloy sphere 2-in Tuballoy reflector, HEU-MET-FAST-003	0.99527 +/- 0.00113	0.99492 +/- 0.00013	0.99362 +/- 0.00087
4	Inp04	Idealized Oralloy sphere on Lucite ring in H2O, HEU-MET-FAST-004 3D model	1.02328 +/- 0.00127	1.00318 +/- 0.00054	1.02284 +/- 0.00099
5	Inp05	ZEUS-1 10 Uniform Units, HEU-MET-INTER-006 case 1	0.99323 +/- 0.00053	0.99282 +/- 0.00055	0.99300 +/- 0.00049
6	Inp06	Un-reflected sphere of enriched Uranium nitrate, HEU-SOL-THERM-013 Case 1	0.97928 +/- 0.00051	0.99894 +/- 0.00054	0.98075 +/- 0.00034
7	Inp07	Jemima #1, Idealized Model, IEU-MET-FAST-001 Case 1	1.00136 +/- 0.00074	1.00030 +/- 0.00013	1.00041 +/- 0.00055
8	Inp08	Un-reflected UO2F2+H2O Cylindrical Assembly, SHEBA-II, ref. LEU-SOL-THERM-001	0.99488 +/- 0.00083	1.01164 +/- 0.00018	0.99445 +/- 0.00071
9	Inp09	Pu sphere surrounded by HEU, Planet assembly, MIX-MET-FAST-001	0.99991 +/- 0.00062	0.99939 +/- 0.00012	0.99992 +/- 0.00056
10	Inp10	Bare Pu-239 Jezebel, ref. PU-MET-FAST-001	1.00030 +/- 0.00055	0.99921 +/- 0.00041	1.00115 +/- 0.00057
11	Inp11	Un-reflected sphere of Plutonium Nitrate Solution, PU-SOL-THERM -009 Case 3A	1.00886 +/- 0.00073	1.01917 +/- 0.00045	1.00841 +/- 0.00037
12	Inp12	Bare U-233 Jezebel, ref. U233-MET-FAST-001	1.00095 +/- 0.00067	1.00030 +/- 0.00012	1.00006 +/- 0.00064
13	Inp13	Uranyl-Flouride Solution with Beryllium reflector, U233-SOL-INTER-001 Case 1	0.99019 +/- 0.00125	0.98509 +/- 0.00020	0.98959 +/- 0.00141
14	Inp14	Un-reflected sphere of U233 nitrate, U233-SOL-THERM-001 Experiment 1	0.97839 +/- 0.00094	1.00104 +/- 0.00017	0.97959 +/- 0.00060

*Values obtained from MCNSkeff are found to deviating less than 1% with respect to MCNP and OpenMC.

Table 6.7. are chosen to test the MCkeff in solving the sub-critical assembly problems. They are related to typical fuel reprocessing vessels and fuel storage facilities. The input files are provided in the CD under subfolder .inputs/storage and the outputs are given in the same CD under the subfolder .outp/storage.

Table 6.7. Comparison of MCkeff results with that of MCNP

SN	File	Title	MCkeff*	MCNP
1	Inp01	Keff of Mixing vessel	0.62147 +/- 0.00165	0.62085 +/- 0.00125
2	Inp02	Single isolated bird cage model	0.32242 +/- 0.00144	0.32155 +/- 0.00123
3	Inp03	PWRFW SO.in	0.80845 +/- 0.00068	0.80954 +/- 0.00053
4	Inp04	Single isolated bird cage model S2.in	0.00458 +/- 0.00004	0.00453 +/- 0.00003

*Values obtained from MCkeff are found to deviating less than 1% with respect to MCNP.

Table 6.8. shows the comparison of keff values between MCkeff and MCNP values. These set of inputs demonstrates all the incorporated geometry configurations. Inputs contains cell translations, 2D and 3D lattices with and without fill patterns and partially filled 2D and 3D lattices. Problems with reflective and periodic boundaries are also given. The inputs for these problems have been written from scratch. The input files are provided in the CD under subfolder .inputs/geometry and the outputs are given in the same CD under the subfolder .outp/geometry.

Table 6.8. Comparison of MCkeff results with that of MCNP for testing lattice structures

S.N	File	Title	MCkeff*	MCNP
1	Inp01	Fill and universe attributes test	1.02912 +/- 0.00213	1.02861 +/- 0.00258
2	Inp02	Test for complement operator	0.99981 +/- 0.00393	0.99948 +/- 0.00260
3	Inp03	Cell translation test (trcl)	0.99048 +/- 0.00047	0.98983 +/- 0.00042
4	Inp04	Lattice cell with fill pattern specified (2D)	1.01361 +/- 0.00578	1.00865 +/- 0.00309
5	Inp05	Lattice cell with filling universe number specified(2D)	0.99156 +/- 0.00187	0.99080 +/- 0.00138
6	Inp06	Nested lattice(2D)	0.18373 +/- 0.00018	0.18385 +/- 0.00018
7	Inp07	Cylinder with partially filled lattice(2D)	0.64734 +/- 0.00153	0.64653 +/- 0.00420
8	Inp08	Box with partially filled lattice(2d)	0.74149 +/- 0.00323	0.73861 +/- 0.00160
9	Inp09	3D lattice with fill pattern specified	0.99251 +/- 0.00335	0.99048 +/- 0.00208
10	Inp10	3D lattice with filling universe number	0.99251 +/- 0.00335	0.99048 +/- 0.00208
11	Inp11	Partially filled 3D lattice	0.85690 +/- 0.00391	0.85955 +/- 0.00138
12	Inp12	Nested 3D lattice	0.15471 +/- 0.00013	0.15472 +/- 0.00014
13	Inp13	Translated cell with filled universe	1.00027 +/- 0.00273	1.00003 +/- 0.00222
14	Inp14	k-inf Experiment, PU-COMP-INTER-001 reflective boundary	1.01025 +/- 0.00065	1.01144 +/- 0.00069
15		k-inf Experiment, PU-COMP-INTER-001 periodic boundary	1.01133 +/- 0.00066	1.01117 +/- 0.00064

*Values obtained from MCNSkeff are found to deviating less than 1% with respect to MCNP.

NOTE:

1. All the problems have been run similar number of batches as that of MCNP as a part of initial results comparison, where as MCNS has to run 4 times more batches than that of MCNP.
2. All the inputs for the above problems can be found in the document MCNS_inputs.docx

6.6. Results on Parallelism

MCk*eff* can be run in parallel mode using the available number of cores on a Computer. MCk*eff* uses maximum available resources on a computer by default, or uses a specific number of threads as given by the user with command line option. MCk*eff* written with compiler directive based parallelising library, OpenMP which has support to the latest fortran compilers.

Testing the proper implementation of parallelizing capabilities in MCk*eff* is very essential prior to other tests. To compare the serial and parallel computing timing results, MCk*eff* had been tested on 2 different Intel CPUs (Intel® Xeon® CPU E3-1225 v3 @ 3.2 GHz, 4 core [Runs on Ubuntu 16.04 LTS] and Intel® Core™ i5-6200U CPU @ 2.3GHz, Dual core [Runs on Ubuntu 16.04 LTS]) using varying number of threads with the GNU provided Fortran compilers.

For the testing purpose, the benchmark input with id HEU-MET-FAST-001 has been chosen with 50000 particles per batch, 50 inactive and 500 total batches. This problem made to run on 2 CPUs in serial and parallel mode. The timing statistics are presented in the below table.

Table 5. Serial and Parallel mode timing statistics

Processor	Mode	Time (Seconds)
Intel® Xeon® CPU E3-1225 v3 @ 3.2 GHz	Serial	255.67
	Parallel	73.68
Intel® Core™ i5-6200U CPU @ 2.3GHz	Serial	278.22
	Parallel	130.77

Timing statistics varies from one to another CPU depending on the number of cores it has and the clock speed. From the table it is clear that the increase in speed in parallel mode is upto 2-3 times more than that of serial mode. This clearly shows that the OpenMP implementation in MCk*eff* is excellent.

6.7. Conclusions

A new Monte Carlo neutron transport code called MCkeff has been developed for estimation of neutron multiplication factor (keff) in fissile systems. The code employs constructive solid geometry method for describing the materials in cells which are bounded by a first or second degree surfaces. The code uses continuous energy point-wise cross sections written in ACE format same as that used by MCNP codes. This enables the program to track individual particles accounting detailed collision physics with very few approximations. At thermal energies, exact treatment of kinematics followed with use of S(α,β) tables and sampling of thermal motion of target nuclides and using probability tables data in the range of 1 to 100 keV in the unresolved energy range for nuclides where this data exists. All the modules of the code is written from the scratch in the latest FORTRAN language (2008 features) and is compiled using both ‘gnu’ and ‘IFORT’ Fortran compiler. The results of the code has been compared with the results of standard legacy codes such as MCNP and OpenMC codes for a wide variety of criticality benchmark problems. These problems are chosen to test the complex geometry models as well as physics models incorporated in the code.

From the comparison provided above, it can be observed that results show remarkable agreement and demonstrate that the physics implementation can be considered validated for the problem domains covered. In particular, the results on the pin-cell problem (Cullen et al., 2004) and the Big Ten benchmark (NEA Nuclear Science Committee, 2009) demonstrate that the implementation of S(a, b) thermal scattering models and the unresolved resonance probability table

method do not exhibit any obvious deficiencies. Results on the Monte Carlo Performance Benchmark (Hoogenboom et al., 2011) were presented to demonstrate the ability to model large models with considerable geometric and material complexity.

6.8. Future works

A new Monte Carlo particle transport code called MCkeff has been developed to compute keff of nuclear fuel cycle facilities. However, due to lack of time some desirable features are not taken up. The following are the list of works can be taken up in future.

1. The MCkeff code has been vectorized (incorporating OpenMP compiler directives to make use of multiple cores/multiple threading) and its efficiency tested. However, it is not parallelized to make use of parallel computers with distributed memory. Since the Monte Carlo codes are inherently suitable to run on parallel computers, efforts can be made to modify the code incorporating Message Passing Interface (MPI) routines at appropriate places in the code.
2. At Present the code is tailor made to compute integral parameter i.e., keff. To make it more versatile, many tally estimators can be added such as flux across surface or the flux inside a cell, current across the surface, energy deposition in a cell, next event estimator etc.
3. Complex geometry is built with the fixed number of analytical surfaces in the code. However, it is easy and many users are familiar to construct geometry with the help of few 3D macro-bodies such as boxes, cylinders, spheres etc. Therefore, it is desirable to add this facility to make the MCkeff code more versatile and user friendly.
4. MCkeff code takes input as either spatially distributed discrete source points or some uniform distribution over a region. Efforts can be made to define arbitrary region of space with energy, angle biasing parameters enabling the code efficient in sampling initial distributions.
5. Proven Statistical tests need to be performed on Output parameters to provide validity of Monte Carlo results. Adding these guide the user in assessing the correctness of the results.
6. MCkeff code can be made as a general purpose Monte Carlo code by incorporating collision physics models of gamma rays and other charged particles. For charged particle transport condensed random walk has to be followed, needs extensive modifications of the present code.
7. Monte Carlo code must compute temperature dependent cross section on-the-fly rather than dependent on data processing codes. This feature can be incorporated easily.
8. It is desirable to estimate fission product concentrations at each time step for running reactor dynamic problems. Very few such codes are in vogue such as MONTE BURN. This is desirable for precise estimation of reactivity parameters.
9. There is a need to couple thermal hydraulics code to take temperature feedback effects, and
10. It is essential to develop theoretical models for error propagation in Monte Carlo codes with the time step.

Acknowledgments

This project work was performed under the DAE-BRNS project sanction No. 36(4) /14/40/2015. The principal investigator, the co-investigator and research fellow are thankful to the Director, Manipal Centre for Natural Sciences, Manipal Academy of Higher Education, for constant encouragement and for helpful discussions on the project. The authors of this report express sincere thanks to the project collaborators and their colleagues from Bhabha Atomic Research Centre, Mumbai for testing the code and providing many valuable comments in improving the code.

References

1. Roger Brewer, Criticality Calculations with MCNP5: A Primer, Los Alamos National Laboratory, LA-UR-09-00380, January 2009.
2. Oak Ridge National Laboratory, "SCALE: a Comprehensive Modeling and Simulation Suite for Nuclear Safety Analysis and Design", Oak Ridge National Laboratory, 2011.
3. John C. Wagner, James E. Sisolak, Gregg W.McKinney, MCNP:Criiticdity Safety Benchmark Problems, LA-12415, DEC-1993.
4. Paul K. Romano , Benoit Forget, "The OpenMC Monte Carlo particle transport code", Annals of Nuclear Energy 51 (2013) 274–281
5. Sutton, T.M. etal., "The MC21 Monte Carlo transport code". Joint International Topical Meeting on Mathematics and Computation and Supercomputing in Nuclear Applications. Monterey, California., 2007.
6. D. Haon, et al., in: Validation of the Apollo-Moret Neutronic Codes on Critical Experimental Configurations Simulating the Shipping Casks for Light Water Fuels, PATRAM'80, 1980.
7. Criticality benchmark of McCARD Monte Carlo code for light-water-reactor fuel in transportation and storage packages, Nuclear Engineering and Technology 50 (2018)
8. Jaakko Leppänen, "Serpent – a Continuous-energy Monte Carlo Reactor Physics Burnup Calculation Code", March,2013
9. R. E. MacFarlane and D. W. Muir. *The NJOY Nuclear Data Processing System*. LA-12740-M. Los Alamos National Laboratory, 1994.
10. J. Ueki and F. B. Brown. Stationarity and Source Convergence in Monte Carlo Criticality Calculation. LA-UR-02-6228. Los Alamos National Laboratory, 2002.
11. F. B. Brown. On the Use of Shannon Entropy of the Fission Distribution for Assessing Convergence of Monte Carlo Criticality Calculations. In Proc. PHYSOR-2006, American Nuclear Society's Topical Meeting on Reactor Physics Organized and hosted by the Canadian Nuclear Society. Vancouver, BC, Canada, Sept. 10–14 2006.
12. The NEA Expert Group on Source Convergence in Criticality-Safety Analysis. URL www.nea.fr/html/science/wpncs/convergence/, Reviewed March, 2007.
13. gnu fortran compiler (The GNU Operating System and the free Software Movement *https://www.gnu.org*)
14. Intel Fortran compiler, (Intel Parallel Studio Cluster Edition *https://software.intel.com/en-us/parallel-studio-xe)*
15. fortran 2008 features, *http://fortranwiki.org/fortran/show/Fortran+2008+status*
16. Hoogenboom, J.E., Martin, W.R., Petrovic, B., 2011. The Monte Carlo performance benchmark test – aims, specifications and first results. In: International Conference on Mathematics and Computational Methods Applied to Nuclear Science and Engineering. Rio de Janeiro, Brazil.
17. Miguel Hermanns, "Parallel Programming in Fortran 95 using O99penMP", Universidad Polit´ecnica de Madrid, Spain, April 2002
18. The Message Passing Interface (MPI) Standard. URL=www-unix.mcs.anl.gov/mpi/, reviewed March 2006.

7. Collection of Publications

1. Monte Carlo simulation: Optimization of Computer memory and time[#]

Sachin Shet and K.V.Subbaiah
Manipal Centre for Natural Sciences, Manipal University
Manipal-576104, Karnataka

Abstract

Monte Carlo technique is widely employed for solving practical radiation transport problems to arrive at a precise solution in a complex geometrical arrangement of materials. This technique is based on the simulation of a "history of a particle or radiation", which comprises of tracking all the possible interaction events of particle randomly occurring with matter as it traverses through it. Although the outcome of each history is prone to statistical uncertainty, the repetition of such a history over a large number would result in the prediction of mean behavior, which is an expected answer for the problem. The simulation of history of a particle calls for the availability of copious quantity of random numbers in the range (0 to 1) and the probability density functions (PDF) or Cumulative distribution functions (CDF), which are deduced from the interaction processes of radiation with matter. The first task can be easily overcome, since there are number of random generators available in the open literature meant for different operating systems satisfying the basic characteristics of random numbers. However, CDFs required are not unique and depend on the type of radiation and the composition of the medium and hence not available in the directly useable form. These have to be generated from the nuclear data table such as ENDF (Evaluated Nuclear Data File), which are available free of cost internationally. The main difficulty with the use of nuclear data table is the number of data points which vary widely from one ~~isotope~~ nuclide to the other. For instance in the case of neutron transport, hydrogen as few thousands data points in the energy grid ranging from 0 to 20 MeV, whereas Uranium as few millions points to cover the same energy range. Therefore storage of data ~~management~~ and ~~its~~ how it is effectively used in Monte Carlo simulations place a crucial role.

In the present work, the usage of computer memory in handling voluminous unstructured data is optimized by the use of dynamic allocation and de-allocation instead of priori fixed allocation, and it is accomplished with the use of memory address pointers. The sorting and editing (deletion and insertion of members) of huge data table scan be done at a faster speed with the help of doubly linked list, which does not search for elements in sequential order. The details of this work will be presented in sequel.

I. Introduction

The prediction of radiation quantity as it traverses through material media is essential while employing the nuclear radiations for the betterment of human kind. From the radiation quantity, it is possible to deduce many of the parameters of practical interest such energy absorbed, heat generated, damage potential and so on. There are two distinct methods of solution (predicting the quantity of radiation) in vogue namely (i) Deterministic method and (ii) Monte Carlo method. The former method is ideally suited for simple arrangement of materials in geometrical figures. And, the latter method is employed, although it is known to consume a lot of computation time, in situations where the materials are arranged in complex configurations. Many of the systems encountered in day to day applications are composite and complex thereby a lot of interest in use

of Monte Carlo method. As a result, rapid development is taking place in inventing new algorithms

in Monte Carlo method as well as efficient programming techniques to reduce the computational time. Random numbers and probability tables are the two essential parameters of all

Monte Carlo Methods. The basic principle under lying in this technique is the simulation of a "history of a particle or radiation", which comprises of tracking all the physical interaction events occurring randomly as it traverses through the matter. Although the outcome of each history is prone to statistical uncertainty, the repetition of such a history over a large number would result in the prediction of mean behavior, which is an expected answer to the problem. The solution accuracy (precision) mainly depends on the sample size. However, it is often experienced for problems of large dimensions having many independent variables, the desired accuracy of the solution cannot be accomplished by just increasing the sample size. To circumvent this difficulty, one has to resort to advanced Monte Carlo methods, which employ variance reduction techniques. Suitability of a particular technique largely depends on the problem and judicious selection of it is an art which demands considerable experience of the user.

In the present work, two basic aspects of Monte Carlo technique pertaining to neutron transport are investigated. They are identification of fast and efficient pseudo random number generators and optimal usage of computer memory in handling voluminous amount of unstructured data. Sorting of both of these issues will make the Monte Carlo method as an attractive alternative for solving radiation transport problems.

II. Monte Carlo Method

Monte Carlo methods are a class of computational algorithms that rely on repeated random sampling of physical and mathematical systems to compute their results. Major components of Monte Carlo methods are as follows:

- *Random number generator* (RNG)
- *Sampling of Probability distribution functions*
- *Scoring* /tally specification
- *Error estimation*
- *Variance reduction techniques*
- *Parallelization and vectorization*

First two components are discussed in detail under the results and discussion section.

III. Results and Discussions

As mentioned earlier, the investigations are limited to two aspects of Monte Carlo method, in particular reference to solving neutron transport problems with optimization of CPU time and computer random access memory utilization. It is known billions of random numbers may be used to solve any practical problem and the selection of efficient and fast algorithms of RNGs will reduce the CPU time consumed. Towards, this end search has been carried out and several RNG routines are investigated and the analysis presented below.

III.a) Random number generators-CPU time optimization

Random number generators are the back bone of all Monte Carlo techniques. In fact, there are no true random number generators. However, even if there exist any, it is not useful for programming purposes on two counts: (i) these will not be of use for developers for program debugging purposes, and (ii) the end results will be fluctuating which does not throw any light on the correctness of the Monte Carlo program. Therefore, all developers of Monte Carlo program rely on pseudo random number generators in the interval (0 to 1), which are fast and provide reproducible sequence satisfying minimal statistical checks such as uniformity, minimum correlation, long period etc. Hereafter, the word pseudo will be dropped and referred ad random numbers.

Table-1. Random number generating functions and time required for 100 million

In the present work 9 random number generating functions available in the published literature, which reportedly generate random numbers uniformly in the interval (0 to 1) and must

have long cycle length(Number after which repetition of same sequence starts). All of them need starting integer(s) called seed(s) and the output will be a single or double precision real number in the interval (0 to 1). As we deal with millions of random numbers, the time required to generate this random numbers by the computer play an important role in the Monte Carlo simulation. For the 9 random number generating functions reported to have satisfied minimal statistical tests, we calculate the average and the time required to generate 100 million numbers is presented in the table-1.

Description	Seeds	Cycle Length	Average	Time (sec)
1.Linear Congruential 32 bit generator.[1]	**No of Seeds: 3** **1< S1, S2, S3 < 30000**	**$6.9x10^{12}$**	**0.4998**	**9.6250**
2.Linear Congruential 32 bit generator.[1]	**No of Seeds: 2** **1< S1,S2 <2147483562**	**$2.3x10^{18}$**	**0.5000**	**7.0156**
3.Linear Congruential 64 bit generator.[1]	**No of Seeds: 3** **1 < S1, S2, S3 < 30000**	**$6.9x10^{12}$**	**0.4998**	**11.3906**
4.Linear Congruential 64 bit generator.[1]	**No of Seeds:2** **1< S1,S2 <2147483562**	**$2.3x10^{18}$**	**0.5000**	**8.0625**
5.Multiplicative pseudo random number generator.[2] [3] [4] [5] 32 bit generator	**No of Seeds: 1** **1< S1 <2147483562**	**NA**	**0.4997**	**5.2655**
6.Multiplicative pseudo random number generator.[2] [3] [4] [5] 64 bit generator	**No of Seeds: 1** **1< S1 <2147483562**	**NA**	**0.4997**	**8.5156**
7.Multiplicative pseudo random number generator.[2] [3] [4] [5] 64 bit generator	**No of Seeds:1** **1< S1 <2147483562**	**NA**	**0.4997**	**8.5156**
8.Multiplicative pseudo random number generator.[6] It is a 32 bit generator, and it uses intrinsic MIN MAX functions.	**No of Seeds:1** **-1 < S1< -2147483562**	**$>2x10^{18}$**	**0.4999**	**28.4062**
9.Multiplicative pseudo random number generator. [6] uses absolute value function and is a 32 bit generator.	**No of Seeds:1** **-1 < S1 < -2147483562**	**$>2x10^{18}$**	**0.5003**	**18.6094**

From the above tables it is seen that the average and time required to generate random numbers varies from one function to another function. Further, it can be observed that the 1st and the 3rd linear congruential generators which uses 3 seeds took more time compared to the 2nd and 4th linear congruential generators which uses only 2 seeds. By using more seeds, steps required to generate a random number will be more hence it take more time to generate a set of random numbers. The 9th Multiplicative pseudo random number generator have large period ($>2x10^{18}$) compared to other function as it is using absolute value with the other normal operations. It can be inferred from the above table, that the 5th takes minimum amount of time for generating random numbers can be used in Monte Carlo Programs.

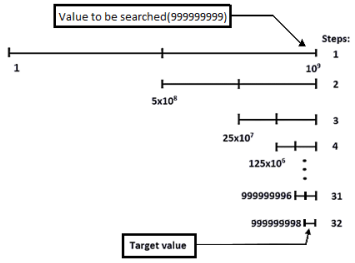
III.b) CPU time and Unstructured Data storage optimization

Another basic requirement in Monte Carlo simulation is Probability Density functions (PDF) and Cumulative Distribution functions (CDF). For neutron transport, these are deduced based on the physical interactions that take place between neutron and the nuclide. The basic interaction data is presented in the form of tables spanning the neutron energy interval in the range of 0 to 20 MeV. Since the magnitude of interaction probability is not smooth varying function of neutron energy and

hence needs to be tabulated at close points of energy. Further, the amount of data tabulated for a nuclide varies from one nuclide to another. For instance, hydrogen has few thousand data points whereas uranium has few millions.

To store and access such unstructured data which varies widely from one nuclide to another consume a large amount of computer memory in conventional programming method. However, the use of "ALLOCATABLE" statement existing in modern FORTRAN programming, allows the user to dynamically allocate the required memory space and thus preventing the reservation a large chunk of unused memory of a computer. Further, the statement of "DEALLOCATE" enables one to reuse the same memory again and again for different purposes.

Another operation which often required is searching for the particular element of data in the huge tabular data. As a customary practice, the search begins with start index till the end of data. This method of searching calls for a lot of CPU time if the data element to be searched lies at the other end of data table. However, if one uses binary search algorithm the number of operations required can be reduced drastically. For instance in a table of natural numbers, to find the interval where "N" lies takes "N" operational steps in sequential method whereas in the binary search method the number is only $\log_2 N$. By adopting binary search method in Monte Carlo simulation, the CPU required can be drastically reduced. Suppose one wants to find a number 999,999,999 in 10^9 natural numbers, in a sequential operation the CPU time required will be 1000 second whereas it will be 32 micro-seconds in a binary search. The same is illustrated in the Fig.1.



To find 999999999 in a sorted list of numbers from 1 to 10^9:

Sequential search: If we try to search by sequential method, it will be taking 999999999 steps to reach the target value 999999999.

Fig.7.1. Illustration of "Binary Search"

Binary search:

In binary search, the interval is halved and tested which half the desired value exist. If the value lies in the 2nd half, the lower value is set to the mid value else the upper value is set to the mid value. Thus every time the search interval becomes half and this is shown in the figure 7.1. The value to be searched in this case in 32 steps. Therefore, binary search method requires a less CPU time.

IV. Conclusions

Monte Carlo is power technique to solve radiation transport problems in arbitrary complex geometrical arrangement of materials. This basic principle adopted in this technique is the simulation of a "history of a particle or radiation", which comprises of tracking all the physical events of particle randomly occurring with matter as it traverses through it. Although the outcome of each history is prone to statistical uncertainty, the repetition of such a history over a large number would result in the prediction of mean behavior, which is an expected answer for the problem. However, the main drawback of this method is the memory and time required to solve a given problem prohibiting its usage. The computer memory and the CPU time are the two important parameters to be addressed to make the Monte Carlo technique attractive. The CPU time can be drastically reduced by the proper selection random number generator and based on this work suitable random number generator has been identified. Use of dynamic allocation and de-allocation, it is possible to reduce the memory usage and by putting the dame memory locations for different purposes. Further, it is demonstrated use of proper search algorithm reduces CPU time drastically.

References

1. Brian Wichman, David Hill, “Algorithm AS183: An Efficient and Portable pseudo Random Number Generator”, Applied Statistics, Volume 31.
2. Paul Bratley, Bennett Fox, Linus Schrage, “A Guide to Simulation”,Second Edition, Springer, 1987, ISBN: 0387964673,LC: QA76.9.C65.B73.
3. Bennett Fox, ”Algorithm 647: Implementation and Relative Efficiency of Quasirandom Sequence Generators”, ACM Transactions on Mathematical Software, Volume 12, Number 4, December 1986, pages 362-376.
4. Pierre L'Ecuyer, ”Random Number Generation”, in Handbook of Simulation, edited by Jerry Banks, Wiley, 1998, ISBN: 0471134031, LC: T57.62.H37.
5. Peter Lewis, Allen Goodman, James Miller, “A Pseudo-Random Number Generator for the System/360”, IBM Systems Journal, Volume 8, 1969, pages 136-143.
6. William H. Press, Saul A. Teukolsky, William T. Vetterling, Brian P. Flannery, “Numerical Recipes in Fortran-The art of scientific computing”
7. Carter, L.L. ,Cashwell, E.D, “Particle-transport simulation with the Monte Carlo method”

2. Comparison of measured and calculated dose rates of Am-Be source with Monte Carlo simulation[#]

Sachin Shet and K.V.Subbaiah
Manipal Centre for Natural Sciences, MAHE, Manipal-576104, Karnataka, India.

Abstract

Monte Carlo simulations (MCS) results for dose rates at an experimental location are compared with measured values of Am-Be source housed in a concrete bunker. The dose rates comprises of both neutron and gamma radiations emitted from the source. The gamma dose rates for 4.44 MeV (emitted from the source) and 2.2 MeV (capture gamma rays from the concrete) are deduced from the count spectrum obtained with 2"×2" NaI(Tl) crystal (available in the literature[1]). The neutron and total gamma dose ratescalculated using MCS and compared with the measured neutron and total gamma dose rates are in mostly good agreement.

I. Introduction

Am-Be sources are widely employed to carry out both neutron and gamma radiation experiments such as activation foils etc. To carry out the experiments safely, the source has to be housed in a suitable container. The radiation fields around the source are required to carry out experiments. In the present work, MCS have been carried out for dose rates due to neutron and gamma rays for a particular location, where experimental results are available enabling comparison between them.

II. Material & Methods

he present experimental facility has a 16 Ci Am-Be source housed inside a concrete bunker, which emits 4×10^7 neutrons/s[1] and 3×10^7 photons/s[2]. The gamma dose rates due to 4.4 MeV and 2.2 MeV are deduced from the count spectrum obtained with 2"×2" NaI(Tl) crystal[1]. The neutron and gamma dose rates calculated using MCS are compared with the measured dose rates.

III. Results & Discussion

MCS results compared with measured dose rates for Am-Be source of strength 16 Ci housed in a concrete bunker which emits 4×10^7 neutrons/s[1]. Am-Be source emits neutrons besides it emits 4.44 MeV gamma ray with sufficient intensity produced from the ^{13}C excited state, whose strength is 3×10^7 photons/s (0.75 photons per neutron[2]). The gamma ray of 59.5 keV emitted by Am source does not contribute to the dose rate at experimental location since it gets absorbed by the SS structure present around the source. MCS have been carried out accounting the bunker structure in detail. The computations include the dose rates due to both neutron and gamma (source and capture gammas) and are presented below.

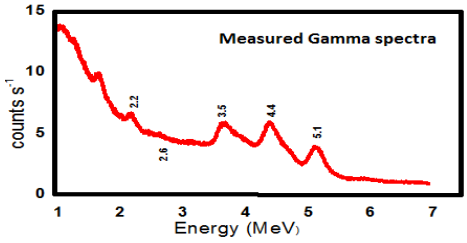


Figure 1 - Measured gamma spectra (reproduced from reference 1)

A) Comparison of Neutron dose rate

The neutron dose rate measured at 85 cm away from the source mid plane with Gamma neutron survey meter is compared with the calculated dose rate. MCNP simulated dose rate is 2036.8 µSv/hr

whereas the measured dose rate at the same location is 1985.0 µSv/hr. The difference in values is less than 3% and are in excellent agreement with each other.

B) Deduction of dose rates for 4.44 MeV & 2.2 MeV gamma from NaI (Tl) count spectrum

Gamma dose rate mainly comprises of 2 components viz. 4.44 MeV source gamma (emitted by the source) and 2.2 MeV capture gamma. The flux at these values are deduced from the count spectrum (Fig.1) measured at 85 cm from the source. The resulting dose rate are compared in table 1 and are in good agreement.

Table 1– Comparison of dose rates

Parameter	4.44 MeV	2.2 MeV*
FWHM(keV)	223	128
Channels	52	67
Counts(s^{-1}) 4.44 MeV	215.45	70.47
3.42 MeV#	34.7	
Efficiency of Detector	3.4%	6.0%
Photons(p/cm^2-s)	293.58	46.6
Dose rate (µSv/hr)	15.85 ±0.8	1.99 ±0.03
MCNP cal. Dose Rate	16.11 ±0.1	1.59 ±0.01

*Capture gamma by $^1H(n,\gamma)^2H$ interaction.
Double escape peak of 4.44 MeV.

C) Comparison of total dose rate by gammas

The total gamma dose rate measured with GM survey meter compared with MCS result.

Measured total dose rate: 9.0 µSv/hr
Simulated total dose rate: 17.84 µSv/hr

D) **Conclusion**

The neutron dose rate calculated is in good agreement with the measured dose rate. However, the under estimate of measured total gamma dose rate might have resulted due to less sensitivity of GM counter for 4.44MeV gammas.

References

1. P. Priyada & P.K. Sarkar, "Use of prompt gamma emissions from polyethylene to estimate neutron ambient dose equivalent", Nuclear Instruments and Methods in Physics Research A 785 (2015) page 135-142.
2. Isao Murataa et al, "Neutron and gamma-ray source-term characterization of AmBe sources in Osaka University", Progress in Nuclear Science and Technology Volume 4 (2014) page 345-348.

3. Advances in Monte Carlo methods in Radiation Transport

Sachin Shet and K.V.Subbaiah
Manipal Centre for Natural Sciences, MAHE, Manipal-576104, Karnataka, India.

Introduction

Monte Carlo (MC) technique is widely employed for solving practical radiation transport problems. This technique is based on the simulation of a "history of a particle" which accounts for the absorption as well as energy transfer mechanisms. Such simulations are useful in predicting the radiation field strengths around the radiation sources thereby enabling control of exposures happening in the nuclear fuel cycle facilities. MC simulation requires copious quantity of random numbers in the range (0 to 1) and the probability density functions (PDF) which need to be deduced from Evaluated Nuclear Data File (ENDF). The main difficulty in the case of neutron transport with point cross sections is the number of data points are voluminous and vary widely from one nuclide to the other. In the present work, the usage of computer memory in handling voluminous data is optimized by the use of dynamic allocation and de-allocation. MC codes known to consume a lot of CPU time, which can be reduced considerably by adopting unionized energy grid method and efficient search algorithm.

Materials and Methods

Major components of Monte Carlo methods are:

– Random number generator (RNG)
– Sampling theory
– Scoring
– Error estimation
– Variance reduction techniques

First two components where in the advances have been made are reported in sequel.

Results and Discussions

The following are the advances[1] attempted in the present work in reducing the CPU time and memory storage for handling data.

Random number generators-CPU time optimization

In the literature[2] a large number of RNGs are available which satisfies minimal statistical tests. For selecting a fast RNG which takes minimum CPU time, several RNGs are tested for the time required to generate 100 million random numbers, and the representative results presented in table.1. Currently Monte Carlo simulations are done on parallel machines, which uses the following RNG.

$$\xi_{i+k} = g^k \xi_i + c\,\frac{g^k - 1}{g - 1} \bmod M$$

Where ξ_{i+k} is i+k th random number,
i and k are integers, g is the multiplier, c is the additive term and M is the largest integer.

Table.1.Random number generating functions with time for
100 million random numbers

S.N	Random number generator (s is the seed of RNG)	Time taken(s)
1	r4_random(s1,s2,s3)	9.6250
2	R4_uni(s1,s2)	7.0156
3	R8_random(s1,s2,s3)	11.3906
4	R8_uni(s1,s2)	8.0625
5	R4_uniform_01(s)	5.2655
6	R4_uniform_02(s)	8.5156
7	R4_uniform_03(s)	18.0124

Dynamic allocation of memory

To store and access unstructured data which varies widely from one nuclide to another, it consume a large amount of computer memory in conventional programming method. However, the use of "**ALLOCATABLE**" statement, allows the user to dynamically allocate the required memory space. Further, the statement of "**DEALLOCATE**" enables one to reuse the same memory again and again for different purposes.

Unionized energy grid method

In sampling of nuclide and type of reaction taking place it is required to search the large data table of nuclides. This can be avoided by using the unionized energy grid. The CPU time can be improved by the use of unionized energy grid method and by following the improved search algorithms. Thus saving CPU time.

Search algorithms

At each interaction point it is required to compute the total and individual reaction cross-section of all nuclides present, by searching huge data table. For instance in a table of length 1 billion to search at fag end on a computer operating at 1 million operations per second, it requires 1000 seconds. While the binary search requires only 32 μs. Thus the use of Binary search algorithms reduces the CPU time drastically. The advances reported here if adopted in future MC codes will make them attractive over other codes.

References

1.Paul K Romano and Benoit Forget, (2013),"The OpenMC Monte Carlo Transport Code", Ann.Nuc.Eneergy, Vol.51, P-274-281.
2.Press W.H. et al., (1992), "Numerical Recipes in Fortran, The art of scientific computing". second Ed. P-266-320.

4. Development of Monte Carlo Code for the Estimation of Neutron Multiplication Factor in Fissile Materials

Sachin Shet and K.V.Subbaiah
Manipal Centre for Natural Sciences, MAHE, Manipal-576104, Karnataka, India.

I. Introduction

Fissile isotopes of Uranium and Plutonium are used in nuclear industries for power generation. Improper handling or storage of fissile materials may lead to criticality accidents [1] resulting in release of a large quantity of energy there by causing damage to the systems and the surroundings. In general such systems are designed with a lot of safety margin causing the economic burden on the projects. To circumvent the cost of the project, it is necessary to estimate precisely the neutron multiplication factor (k_{eff}) of the system. Although many methods of calculating k_{eff} are in vogue. Monte Carlo (MC) methods (MCNP[2],OpenMC[3]) are preferred due to the capability of these methods in handling complex arrangement of geometrical shapes and in accounting detailed physics. The basic of MC method as applicable to neutron transport is sampling of events randomly that may happen with nuclei while neutron traversing through the material media. The computer codes based on MC procedure that exist today needs more of Central Processing Unit (CPU) time and computer memory prohibiting the usage in general. However, rapid developments both in software and hardware of computers, in addition to development of new algorithms in MC procedures attracting the users for solving variety problems. To reduce CPU time (process which consumes more CPU time is identified as search algorithm in large arrays), new algorithms are being used such as binary search method and logarithmic energy grid method besides making use of parallel computers with multiple cores. Further to limit the memory usage dynamic allocation is adopted avoiding the reservation of large fixed memory. In the present work the MC code namely Monte Carlo for Neutron Simulation of k_{eff} (MCNS_k_{eff}) has been developed in house incorporating the above advanced features for the estimation of neutron multiplication factor in fissile materials configured in complex geometrical shapes.

II. Methods

Neutron multiplication factor of a system is defined as the ratio of number of neutrons produced in a generation to that of number started in the system. To estimate neutrons produced, it is necessary to account all predominant modes of interaction of neutrons with the nuclei present in the system. Generally neutron interactions with nuclei are elastic scattering, inelastic scattering, radiative capture, nuclear fission and particle emission reactions such as (n,2n), (n,3n), (n,d) and (n,p) etc. The interaction probabilities (cross sections) are depend on the neutron energy and these exhibit sharp resonances demanding the tabulation at very close energy intervals. Such tabulations known as continuous-energy cross section data or point wise data available as ACE files (A Compact ENDF). Such data generated using NJOY [4] code for several commonly used nuclides available and is used for testing the code developed. In addition to tabulated cross sections for each of the reaction, secondary angle and energy distributions is also provided.

Geometric cells bounded by surfaces containing materials are specified by combining the surfaces with the use of binary operators such as union, intersection and complement. It is possible to construct any complex configurations of materials with a limited number of surfaces. In the present code we use 21 basic surfaces of first and second degree which are enough to describe any conceivable geometry.

k_{eff} of a system is calculated iteratively starting with a finite number of neutrons in cells containing fissile materials (source points) and tracking them from its birth to its death through the system while scoring the events of interest namely fission. Although any event may not have any

physical meaning, but average over a collection of large number of events represent the behavior of the system, which is the required answer for the problem. In MC simulation birth of a neutron is taken to be the source point or fission site, and death of a neutron taken to be either absorption or escape from the system. At the end of tracking of predetermined number of neutrons, k_{eff} is estimated and fission sites are stored for starting subsequent iterations. The iterations are stopped when the value of k_{eff} of a system converges. To ensure that the neutron population does not grow exponentially, number of fission sites are controlled if found in excess and sampled over and again if the number of fission sites are less than the predetermined number.

III. Details of development

The Mckeff is cross-platform code developed for both Linux and Windows operating systems (OS) written in Fortran language with Fortran 2008 features. The notable features are:

- Input file – divided as 3 blocks of data accepts in a free format
- Debug feature – input is processed and if found any warning and fatal error messages are displayed
- Parallel mode run – By default, Mckeff uses all the available number of threads in a core. User can run w*ith the specified number of threads with the use of command line arguments
- Continue Run –Large CPU problems can be broken and can be made restart runs. Code status binary file is maintained.
- S(α,β) treatment - For neutrons energies less than 4 eV, the free gas treatment of kinematics is not valid. Scattering kinematics is different for crystalline solids and bound molecules and these are taken care of by using S(α,β) cross section tables
- Unresolved resonance region probability tables – To account for presence of large number of resonances in the energy region (keV), probability table method is employed.
- Repeated structure handling – Input can be specified in terms of universes and lattices, which can be repeated at ease.
- Geometry viewer –Interactive of 2D geometry plots can be made to identify geometry errors.

A typical 2D plot of rectangular lattice is shown in Fig.1

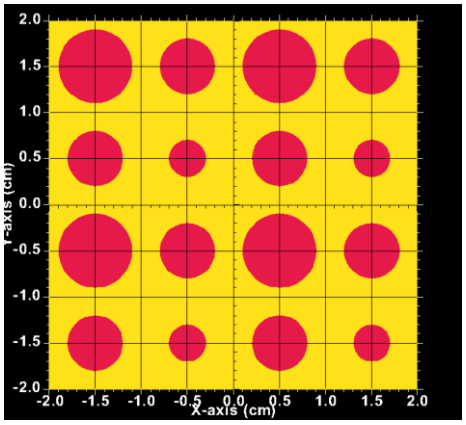


Fig 1. 2D plot of rectangular lattice with three different universes (red-fuel, yellow-water)

IV. Results

Mckeff has been indigenously developed in Fortran language [2008] for operating systems both for Linux and Windows. To validate all the features of the code, variety of criticality benchmark problems are chosen representing solid, liquid fuels and their arrangement in terms of lattices. k_{eff} can be estimated by three different scoring functions namely absorption, collision and tracklength.

$$\text{Collision estimator} \quad k_{eff}^{c} = \frac{1}{N}\sum_{i} W_i \left[\frac{\Sigma_k f_k \upsilon_k \sigma_{f_k}}{\Sigma_k f_k \sigma_{T_k}}\right]$$

$$\text{Absorption estimator} \quad k_{eff}^{A} = \frac{1}{N}\sum_{i} W_i \upsilon_k \frac{\sigma_{f_k}}{\sigma_{c_k} + \sigma_{f_k}}$$

$$\text{Tracklength estimator} \quad k_{eff}^{TL} = \frac{1}{N}\sum_{i} W_i \sigma d \sum_{k} f_k \upsilon_k \sigma_{f_k}$$

The above equations contain standard symbols and details are given Reference 2. By combining these 3, the average k_{eff} of the system is estimated which will have less standard deviation. All the

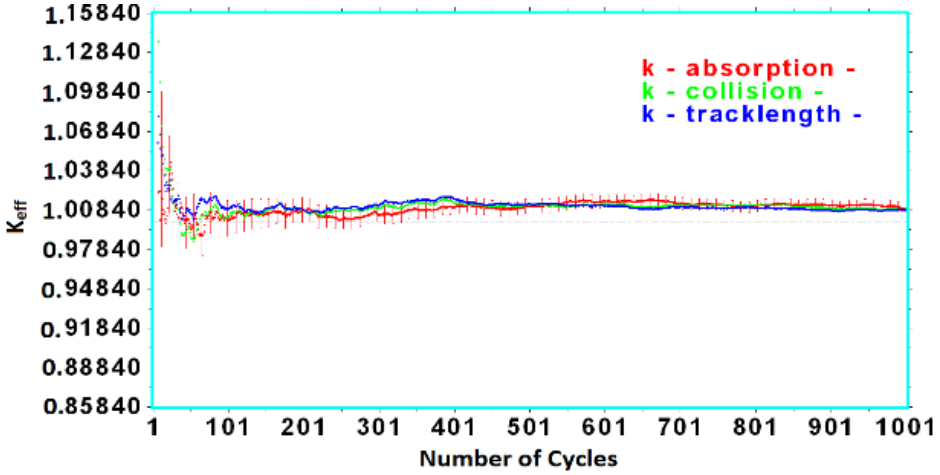


Fig 2 k_{eff} vs Cycle for Godiva reactor assembly

3 k_{eff} for Godiva reactor obtained with present code versus cycle number is shown in Fig.1 along with one standard deviation [σ]. The σ is very large in the beginning of cycles indicating the system has not reached its equilibrium state. The σ softens off after about 100 cycles indicating the convergence of k_{eff} value beyond this point.

Table 1 compares the typical results of average k_{eff} obtained from the present code with that of MCNP using the same cross section data and identical physics parameters, which has been validated by many other workers. A good agreement between them is observed.

Problem		MCk_{eff}	MCNP
1. Godiva Reactor[solid-sphere]		1.0070 ± 0.0006	0.9995 ± 0.0005
2. UO_2F_2 Solution [liquid Cylinder]	S(α,β)	0.9967 ± 0.0008	0.9984± 0.0007
	No S(α,β)	0.9982 ± 0.0008	1.0013 ± 0.0007
3. Lattice[Rectangular]		0.05956 ± 0.0001	0.05983 ± 0.0001
4. Lattice [Hexagonal]		0.0121 ± 0.0002	0.0114 ± 0.0001

Table-1: Comparison of k_{eff}

V. Acknowledgement

Authors express sincere thanks to Dr. Kapil Deo Singh and Dr. S. Anand for providing training on parallel computer. Authors also express thanks to Director, MCNS, MAHE for constant encouragement throughout the work.

VI. References

1. Knief, Ronald Allen. "Nuclear criticality safety." Am. Nucl. Soc., Illinois (1985).
2. X-5 Monte Carlo Team, MCNP–A General Monte Carlo N-Particle Transport Code, Version 5 - Volume III: Developer's Guide, *Los Alamos National Laboratory report LA-CP-03-0284*
3. Paul K. Romano, Nicholas E. Horelik, Bryan R. Herman, Adam G. Nelson, Benoit Forget, and Kord Smith, "OpenMC: A State-of-the-Art Monte Carlo Code for Research and Development," Ann. Nucl. Energy, 2015
4. R.E. McFarlane &D.W. Muir: The NJOY Nuclear Data Processing System, Version 91. LA-12740-M (October 1994)
5. DISLIN – Fortran graphics library - Max Planck Institute for Solar System Research

5. Advances of Monte Carlo Methods in Radiation Physics

K.V.Subbaiah
Manipal Centre for Natural Sciences, Manipal Academy of Higher Education
Manipal, Karnataka-576104, India.

ABSTRACT

Monte Carlo methods (MCM) are widely employed for solving problems in several areas of science, engineering, mathematics and a host of other fields. These methods are well suited for problems whose outcome is statistical in nature. However, these methods are also used even if the problems are deterministic in nature, by recasting them in to the problems of probabilistic in nature. Major drawback of the method is that it needs a large amount of computer time and memory and the other drawback is that the result is not a finite number but a spread over the mean value. These drawbacks are circumvented to some extent, owing to the advent of fast computers on the one hand and the continuous efforts of scientists on the other hand. As a result several noteworthy advances in Monte Carlo methods are made and thus making the method as a preferred one over the several other approaches. This talk briefly touches up on the advances in MCM made for solving problems pertaining to physics in radiation research.

Monte Carlo method arrives at the answer to the problem by sampling randomly all of the physical variables over a large number of trails. In radiation transport, the variables are different interaction processes between the radiation and the matter, and the MCM keeps track of all the events as the radiation traverses through the matter. Such a book keeping process in Monte Carlo nomenclature is called as a "history of a particle". Repetition of histories over a large number enables one to arrive at a desired answer. To accomplish this task, simulation requires copious quantity of random numbers in the range (0 to 1) and the probability density functions (PDF) which need to be deduced from Evaluated Nuclear Data File (ENDF) containing the interaction processes and their magnitudes. Major functional- modules of Monte Carlo methods are:

– Random number generator (RNG)
– Sampling theory
– Scoring
– Error estimation
– Variance reduction techniques

The advances made so far in handling the above modules are

- Fast search algorithms to reduce computational time
- Efficient data management in reducing the computer memory need
- Recent methods of sampling physical variables
- Improvements in scoring methods, and
- Applying of automatic variance reduction technique to enhance computational speed

In the present talk, the above points are discussed in detail.

6. Art of programming for parallel computers

Sachin Shet and K.V.Subbaiah.
Manipal Centre for Natural Sciences, Centre of excellence,
Manipal Academy of Higher Education (MAHE), Manipal – 576104.

Keywords: Parallel Computing, Hybrid programming, OpenMP, Message Passing Interface

Abstract

Parallel computing is the simultaneous use of multiple computing resources to solve a computational problem, which helps in various fields where high-performance computation is necessary such as weather forecasting, modeling of the economy of a nation and in simulations of physical systems in scientific computing etc. Manipal Centre for Natural Sciences (MCNS), Manipal Academy of Higher Education (MAHE), Manipal equipped with a computing cluster namely Shakti HPCC (High-Performance Cluster Computer) consisting of 8 nodes of Intel Xeon E5 family. In this work, it is demonstrated that it is possible to solve problems with parallel computing using Shakti HPCC very efficiently within a very short period compared to the usual computers.

Shakti HPCC runs on RedHat Linux Enterprise Server operating system over which Intel computing tools are provided for the efficient use of the resources. It is very essential to verify the correctness of the installed libraries such as Intel MKL (Math Kernel Library), tools (compilers, debugger, and several other Intel tools) and connectivity prior to parallel computing. Fortran multithreaded, message passing, and hybrid programs are made to run on Shakti, to check the implementation of parallel computing tools such as OpenMP and Intel MPI and the connectivity between the nodes, which shows the correctness of parallel computing environment setup. As Shakti HPCC falls under the hybrid (Shared-Distributed) memory architecture, hybrid parallel programming models are also used for verification. Parallel multithreaded program to compute π has been tested which demonstrate the difference in total execution time of serial and parallel mode runs on a single node, and the results are presented.

7. MCk$_{eff}$ A Monte Carlo Neutron Transport Code for estimation of neutron multiplication factor of Fissile Systems

K.V.Subbaiah
Manipal Centre for Natural Sciences, Centre of Excellence
Manipal Academy of Higher Education, Madhav Nagar
Manipal-576104, KARNATAKA,INDIA

ABSTRACT

A new Monte Carlo code called MCNSkeff for solving neutron transport equation is currently under development at the Manipal Centre for Natural Sciences, MAHE, Manipal. MCNSkeff is an acronym for **M**onte **C**arlo **N**eutron transport **S**imulation code , for estimation of neutron multiplication factor (**keff**) of fissile systems. Simulations are carried out at random from the birth of a neutron till its death using continuous-energy cross sections and recording all the events(history) while the neutron traversing in a system containing fissile materials. While any particular history may not have any physical significance, however an average of histories over a large number provides the physical characteristics of the medium. The events simulated are elastic scattering, fission (or first-chance fission, second-chance fission, etc.), inelastic scattering, (n, xn), (n, c), and various other absorption reactions given in any particular evaluation. For those reactions with one or more neutrons in the exit channel, secondary angle and energy distributions are sampled as per the laws prescribed in the ENDF 6 file. For thermal neutron energies below 4eV, collision kinematics are different due to motion of target nucleus. This is treated accurately in the code either sampling target velocity assuming Maxwell distribution of target nuclei or using S(α,β) cross sections and coupled outgoing angle-energy tables. Besides, at around intermediate neutron energies, unresolved resonance energy region, (few keV to tens of keV) self-shielding effects on cross sections are significant and properly accounted in the code using probability tables of cross sections. At the end of simulation, the code estimates from the recorded histories the neutron multiplication factor (k) of a system by three different ways, namely collision estimator (k^{col}), absorption estimator (k^{abs}) and track length estimator (k^{tra}) along with their standard deviations. Further, the combined k_{eff} is estimated from these three to give a reduced uncertainty band. The salient features of the code are:

- Written from scratch in latest FORTRAN 2008 for Windows and Linux Operating systems
- Input processing is done extensively to detect all possible errors to give warning and fatal error messages
- It allows building of complex configuration of materials in cells bounded by linear and quadratic surfaces
- It has provision for specifying universes and lattices enabling built of complex structures with ease
- Interactive debug features of 2D geometry plotting, tracing of particle and cross section summary
- Attempts are being made to run on Parallel computers.

MCNSkeff code is validated by comparing results of several benchmark criticality problems with other criticality legacy codes[1,2] results and found to be in a remarkable agreement among themselves. In the talk, details of Monte Carlo procedure, sampling theory, estimation of neutron multiplication factors are discussed. Further, merits and demerits of the code and validation of results with other legacy codes will be discussed.

References

7. Collection of Publications

1. F.B. Brown, B.C. Kiedrowski, J.S. Bull, "Verification of MCNP6.1 and MCNP6.1.1 for Criticality Safety Applications," Los Alamos National Laboratory report, LA-UR-14-22480 (2014).
2. Paul K. Romano, Nicholas E. Horelik, Bryan R. Herman, Adam G. Nelson, Benoit Forget, and Kord Smith, “OpenMC: A State-of-the-Art Monte Carlo Code for Research and Development,” Ann. Nucl. Energy, 82, 90–97 (2015).

8) Simulation of Dose Rates for MCNS - Table Top Accelerator at Manipal using FLUKA

Sachin Shet and K.V.Subbaiah
Manipal Centre for Natural Sciences, Centre of Excellence,
Manipal Academy of Higher Education, Manipal – 576104, Karnataka, India.

Abstract

A 50-kV indigenously designed tabletop accelerator (TTA) has recently been fabricated and installed at Manipal Centre for Natural Sciences (MCNS), Manipal Academy of Higher Education (MAHE) with the technical support of the design team from Inter-University Accelerator Centre, Delhi [1,2]. The MCNS-TTA is equipped with a cold plasma-based Penning Ionization Gauge (PIG) ion source. The full ion source is assembled in nylon housing and connected to the +50 kV power supply. Inside the nylon housing, there is annular cylindrical Cathode (37mm diameter made up of Mild Steel) and Anode (22mm diameter made up of Stainless Steel). High voltage was applied to the anode keeping the cathode at the ground. A precaution has been taken in such a way that the required gas will flow from the Cathode body via a small opening without the vacuum deteriorating. Electrons dislodged from H_2 gas gets accelerated towards anode striking steel and produce bremsstrahlung radiation. Gamma-ray dose measurements around the TTA facility was carried out during installation and it was found that the existing shield arrangement was inadequate. Monte Carlo code FLUKA [3] calculations have been carried out to estimate the optimum shielding. In order to simulate the experimental scenario, a simplified geometry of ion source assembly is considered. In this simulation, a beam of electrons with an energy of 50 keV is incident on a 6 mm thick mild steel plate. The plate is at a distance of 2 cm from the source position and detectors are

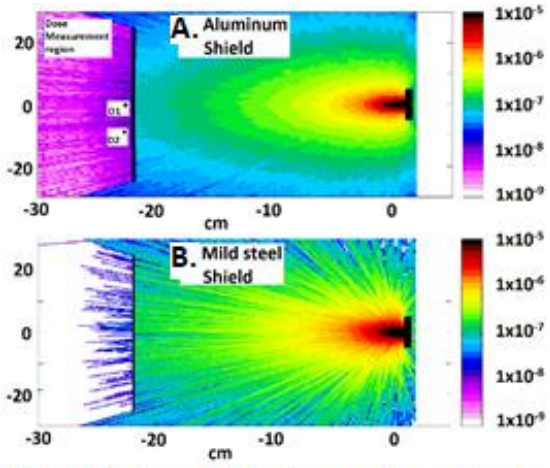


Fig 1. Photon distribution produced due to 50 keV electrons impinging on MS target

located at 24 cm away from the target. The EMFCUT card available in FLUKA is used to set the transport and production threshold for electrons and photons. The electrons and photons transport are scored using the USRBIN. The photon energy distribution is scored at a distance of 24 cm from the source position using the track length estimator USRTRACK. A total of 10^8 histories divided into 10 batches are used for the simulation. The simulations were carried out first with 1.5 mm thick Aluminium sheet (existing one) and it was observed that the dose rate at one point was higher than the permitted level. The photon intensity distribution versus distance from the target is shown in Fig.1A. Next, the calculations are repeated with 1.5 mm mild steel sheet at a distance of 22 cm same as the previous case. The resulting photon distribution with MS sheet is shown in Fig. 1B. The

photon distributions are converted to dose rates with ICRP-74 and the dose rates measured with the ROTEM gamma survey meter are compared and found to be in good agreement with their estimates. Gamma dose rates found to be well below permissible levels on with the MS shield. The TTA has since been upgraded with the 2 mm MS cage cover for the ion source for additional safety. This has ensured that any emitted radiation is completely shielded. Details of the findings using the FLUKA code will be presented.

Acknowledgements

The MCNS-TTA is a fully supported project of MCNS, MAHE. The authors thank P. Priyada, formerly Post-Doctoral Fellow, MCNS and Amit Roy, Adjunct Faculty, MCNS and former Director, IUAC, New Delhi for useful discussions. Sachin Shet is Senior Research Fellow at MCNS, MAHE under DAE-BRNS project No. 36(4)/14/40/2015.

References

1. P. K. Rath et al, "*Fabrication and. Installation of a Table Top Accelerator (TTA) at Manipal*." DAE-BRNS Symposium on Nuclear Physics, Volume 63 (2018), 1060.
2. P.K. Rath et al, "*Fabrication of Table Top Accelerator (TTA) at MCNS*" MRC-2019-TS-10, 2019
3. T.T. Böhlen et al, "*The FLUKA Code: Developments and Challenges for High Energy and Medical Applications"* **Nuclear Data Sheets 120, 211-214 (2014).**