Application of ENDF Nuclear Data for Testing a Monte-Carlo Neutron and Photon Transport Code

P. Siangsanan, W.Dharmavanij and S. Chongkum
Physics Division, Office of Atomic Eneegy for Peace (OAEP),
Ministry of Science Technology and Environment, Thailand

A Monte-Carlo photon and neutron transport code was developed at OAEP. The code was written in C and C++ languages in an object-oriented programming style. Constructive solid geometry (CSG), rather than combinatorial, was used such that making its input file more readable and recognizable. As the first stage of code validation, data from some ENDF files, in the MCNP's specific format, were used and compared with experimental data. The neutron (from a 300 mCi Am/Be source) attenuation by water was chosen to compare the results. The agreement of the quantity $1/\Sigma$ among the calculation from SIPHON and MCNP, and the experiment—which are 10.39 cm, 9.71 cm and 10.25 cm respectively—was satisfactorily well within the experimental uncertainties. These results also agree with the 10.8 cm result of N.M., Mirza, et al.

1. Intrduction

We have developed our own Monte-Carlo particle transport code. The particles of our interest are neutron and photon, because they are available in our laboratory. We have used the reputed MCNP code, including its ACE nuclear data libraries, as our benchmark. Obviously, the MCNP is very long and complicated code because it was written in FORTRAN and has an old-style programming. It could have taken us much effort to study MCNP's source code directly. Instead, we have chosen to study the calculation method from other sources (including MCNP manual, of course) and developed our own code using MCNP as a guide. We chose C/C++ language because it has a neat structure, especially the OOP style of C++. Not only the enhanced speed of code development, its source code reuseability is also important. We can add new features to the code with much less effort–comparing to FORTRAN. Some people may worry about speed penalty to choosing C/C++ over FORTRAN, but we do not, because there are many good numerical libraries written in C/C++ that is comparable to that in FORTRAN. In our case we have chosen BLITZ++ (from http://oonumerics.org/blitz) because the vector operations are extensively used in our code.

2. The Code

The name of the code is Simple Implementation of PHOton and Neutron transport code (SIPHON). It is not final yet, but it has some major useful features. We have used the concept of Constructive Solid Geometry (CSG)[8] which define the geometry as a solid object. There are four basic shapes in SIPHON: cuboid, cylinder, sphere and cone. Objects of other shape could be obtained through Boolean operation between them. Some features of SIPHON are:

- User interface via text input file (an example input file in this benchmark is shown below)
- Repeated structure
- Fixed particle sources
- Neutron and photon transport
- Thermal neutron scattering treatment
- Photon energy deposition in a cell
- Track length estimate of particle flux in a cell

The flowchart of the code is shown in fig.1.

Table 1: An example of input file

```
object{ #declare srcwall = cylinder{<0,0,-.93>, <0,0,.93>, .87 } mat<0> }
#declare srcvial1 = cylinder{ <0,0,-1.13>, <0,0,1.97>,1.1 }
#declare srcvial2 = cylinder{ <0,0,-.93>, <0,0,1.77>,.9 }
object{ srcvial1 ~ srcvial2 mat<7> }
#declare srctube = object{ #declare srcvoid1 = cylinder{
<0,0,-15>, <0,0,32>, 2.3 } ~ #declare srcvoid2 = cylinder{
<0,0,-14.8>, <0,0,32>, 2.1 } mat<4> }
object{ srcvoid2 ~ srcvial1 mat<6> }
object{ srcvial2 ~ srcwall mat<6> }
#declare detvshroud = cylinder{ <16.2,0,-13.335>, <16.2,0,13.335>, 1.31 }
#declare upperCd1 = object{ #declare uCd1 = cylinder{
<16.2,0,-13.335>, <16.2,0,-1>, 1.49 } ~ cylinder{
<16.2,0,-13.335>, <16.2,0,-1>, 1.31 } mat<2> }
#declare lowerCd1 = object{ #declare lCd1 = cylinder{
<16.2,0,1>, <16.2,0,13.335>, 1.49 } ~ cylinder{
<16.2,0,1>, <16.2,0,13.335>, 1.31 } mat<2> }
#declare knifetool1 = cylinder{ <16.25,2,-14>, <16.25,2,14>, 2.1 }
#declare knifetool2 = cylinder{ <16.25,-2,-14>, <16.25,-2,14>, 2.1 }
#declare knifetool3 = cylinder{ <14.25,0,-14>, <14.25,0,14>, 2 }
#declare patchCd1 = object{ cylinder{ <16.2,0,-13.335>, <16.2,0,-1>, 1.58 } ~
uCd1 ~ knifetool1 ~ knifetool2 ~ knifetool3 mat<2> }
#declare patchCd2 = object{ cylinder{ <16.2,0,1>, <16.2,0,13.335>, 1.58 } ~
lCd1 ~ knifetool1 ~ knifetool2 ~ knifetool3 mat<2> }
#declare dettube = object{ #declare detvoid1 = cylinder{
<16.2,0,-15>, <16.2,0,32>, 1.9 } ~ #declare detvoid2 = cylinder{
<16.2,0,-14.8>, <16.2,0,32>, 1.7 } mat<4> }
object{ detvoid2 ~ upperCd1 ~ lowerCd1 ~ patchCd1
~ patchCd2 ~ detvshroud mat<6> }
#declare detwall = object{ detvshroud ~
#declare detel = cylinder{
<16.2,0,-9.8425>, <16.2,0,10.795>, 1.259 } mat<8> }
#declare detector = object{ detel mat<5> }
#declare splittank1 = cuboid{ <-55,-55,-55>, <5.25,55,55> }
#declare splittank2 = cuboid{ < 5.25,-55,-55>, <8.25,55,55> }
#declare splittank3 = cuboid{ < 8.25,-55,-55>, <11.25,55,55> }
#declare splittank4 = cuboid{ < 11.25,-55,-55>, <55,55,55> }
#declare tank = cylinder{ <0,0,-16>, <0, 0,16>, 35 }
#declare watertank1 = object{ (tank & splittank1) ~ srcvoid1 mat<1> }
#declare watertank2 = object{ tank & splittank2 mat<1> }
#declare watertank3 = object{ tank & splittank3 mat<1> }
#declare watertank4 = object{ (tank & splittank4) ~ detvoid1 mat<1> }
object{ #declare otank = cylinder{ tank mat<8> }
object{ #declare env = sphere{ <0,0,0>,50 } ~ otank ~ srcvoid1
~ detvoid1 mat<6> }
object{ sphere{ <0,0,0>,55 } ~ env mat<0> }
//+++-Last(data) Section-++++++
mode = neutron
source={volume<1>, vect=<1,0,0>, dir=<0>, erg=<d 1> }
//Am/Be neutron energy distribution.
cdf 1{i=[ 0.0900 .1852 .3704 .5555 .7407 .9259 1.1111 1.2963
1.4815 1.6667 1.8518 2.0370 2.2222 2.4074 2.5926 2.7778
2.9630 3.1481 3.3333 3.5185 3.7037 3.8889 4.0741 4.2592
4.4444 4.6296 4.8148 5.0000 5.1852 5.3704 5.5555 5.7407
5.9259 6.1111 6.2963 6.4815 6.6667 6.8518 7.0370 7.2222
7.4074 7.5926 7.7778 7.9630 8.1481 8.3333 8.5185 8.7037
8.8889 9.0741 9.2592 9.4444 9.6296 9.8148 10.0000 10.1852
10.3700 10.5555 ],
p=[0.00000 .15314 .30671 .45185
```

```
-cont-
.57871 .69014 .78871 .87585 .96013 1.05370 1.15656 1.25942
1.36228 1.46656 1.58227 1.72798 1.90226 2.08226 2.24512 2.39583
2.53869 2.65726 2.78654 2.92654 3.07225 3.21939 3.37082 3.52368
3.65939 3.77867 3.88867 3.98467 4.06638 4.15066 4.24423 4.33637
4.41808 4.49522 4.57236 4.65879 4.74522 4.83450 4.92021 4.99449
5.05449 5.09663 5.12591 5.14591 5.16305 5.18019 5.20090 5.22733
5.25590 5.28233 5.30376 5.32162 5.33590 5.34661]}
mat 1[-1 1001.60 2 8016.60 1 lwtr.01t] //H2O
mat 2[-8.65 48000.35 1] //Cd
mat 3[1.236e-5 5010.60 1] //B-10
mat 4[-7.87 26000.50 1] // Fe
mat 5[1.2875e-5 5010.60 .96 5011.60 .04 9019.60 3] //BF-3
mat 6[-.0013 7014.60 4.348 8016.60 1.087] // Air
mat 7[-.97 6000.60 1 1001.60 2] // Polyethylene
mat 8[-7.92 26000.50 -.695 24000.50 -.19
28000.50 -.095 25055.60 -.02] // ss304
tally{ neutron flux within cell<detector> }
tally{ neutron flux within cell <detector> multiplier<1,3,107> }
run 500000 particles
imp[1 1 1 1 1 8 8 8 8 8 8 8 8 1 2 4 8 1 1 0]
```

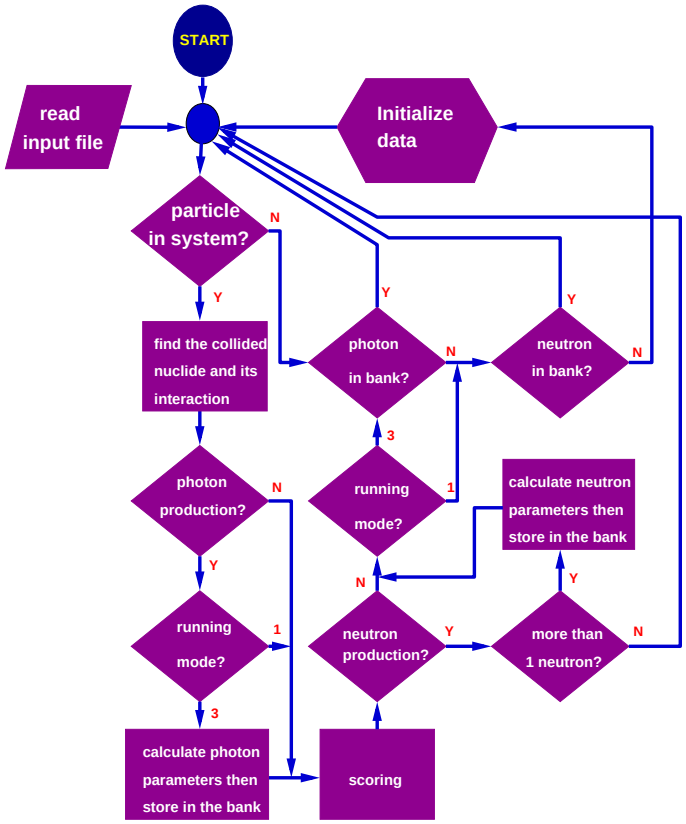

Figure 1: The flowchart of SIPHON

3. Experiment

The neutron attenuation in water was chosen to be the first validation scheme of the code. The experimental setup composed of a 300 mCi Am/Be neutron source installed in the middle of a 70-cm-diameter water tank. A cadmium-covered BF_3 detector was installed along side

of the source to measure the neutron activity. The distance of the detector to the source is allowed to vary from 2 cm to 26 cm with 2-cm-step increment. The thickness of cadmium is 1.8 mm and a 2-cm-wide window was provided at the middle of the detector. The window of the detector was aligned to the center of the source. The setup is shown in fig.2 and the detailed description of the detector is shown in fig.3

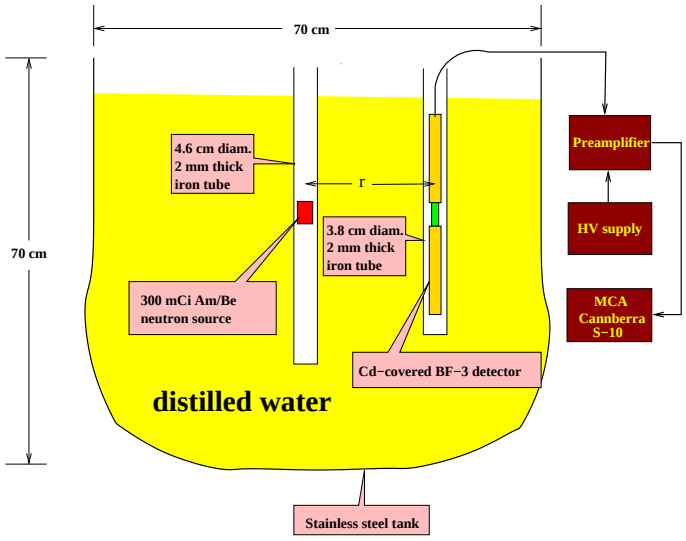

Figure 2: The experimental setup

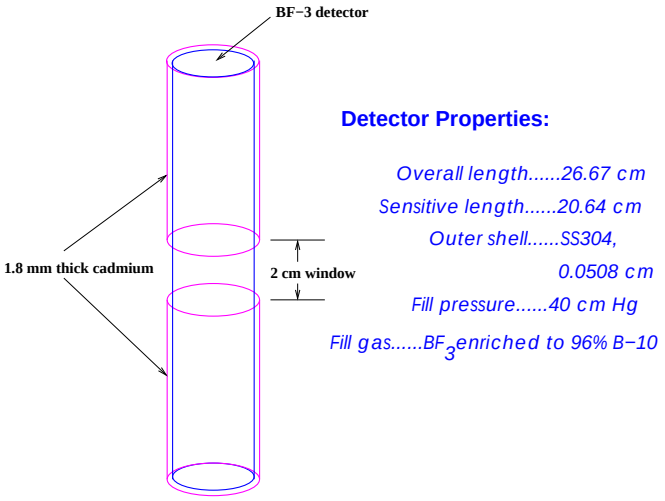

Figure 3: The detector description

4. Calculation

The model of the calculation at r = 16.2 cm is shown as the input file in table 1. In this calculation we have treat the implicit neutron capture, thermal neutron scattering and geometry splitting. At other r's the models are similar.

5. Results

The results from the experiment and the calculation are shown in table 2. The plot of $C^B \cdot r^2$ against r, where C^B is the activation of boron within the detector and can be

calculated from $C^B = \int_0^\infty \Sigma_{(n,\alpha)}(E)\phi(E)dE$, is also shown in fig.4. The source emission rate used was 96% of 6.6×10^5 n/s.

r (cm)	counting rate (n/s)				
	$C^B = \int_0^\infty \Sigma_{(n,\alpha)}(E)\phi(E)dE$				
	SIPHON		MCNP		Experiment
	C^B(rel.error)	fom	C^B(rel.error)	fom	mean(rel.error)
6.2	247.76(0.0015)	134.77	250.72(0.024)	13	304.47(0.0033)
8.2	209.75(0.013)	119.63	216.31(0.015)	10	249.36(0.0036)
10.2	155.97(0.013)	88.71	149.89(0.018)	7.3	182.84(0.0043)
12.2	114.11(0.011)	64.44	108.84(0.021)	5.2	129.04(0.0051)
14.2	81.13(0.013)	46.81	73.66(0.017)	3.6	87.87(0.0061)
16.2	55.37(0.0078)	59.96	52.27(0.015)	22	58.6(0.0075)
18.2	39.07(0.014)	41.72	34.39(0.016)	5.2	41.51(0.009)
20.2	26.83(0.016)	34.52	23.28(0.012)	13	28.00(0.01)
22.2	18.86(0.015)	24.61	16.00(0.013)	8.6	18.86(0.013)
24.2	12.89(0.024)	18.65	10.59(0.016)	6.6	12.88(0.016)
26.2	9.02(0.02)	12.38	7.67(0.02)	4.5	9.8(0.018)
28.2	6.18(0.02)	9.14	5.06(0.02)	3.4	6.35(0.023)
30.2	4.16(0.026)	4.17	3.36(0.029)	2.2	4.47(0.027)

Table 2: The counting rate from measurements and calculations

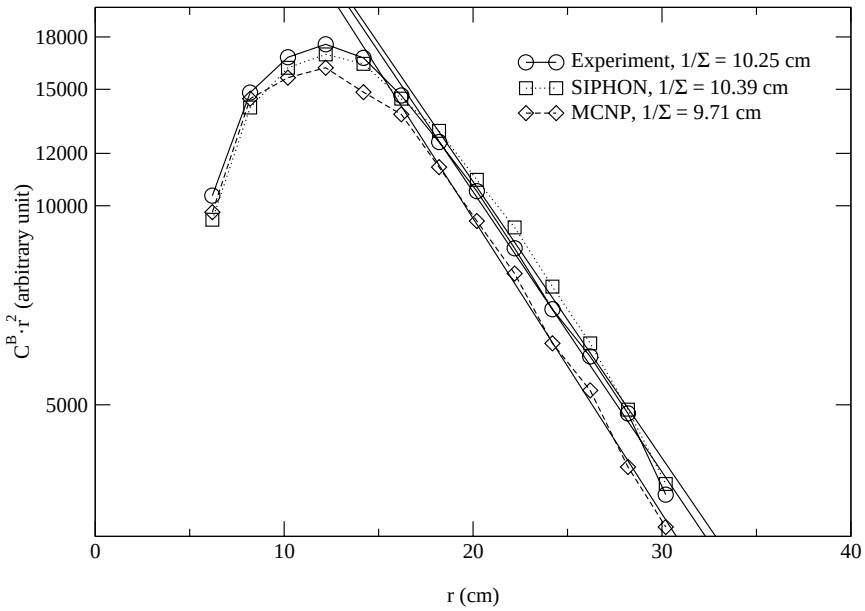

Figure 4: Plot of $C^B \cdot r^2$ against r to determine and compare $1/\Sigma$

6. Conclusion

It was seen from the plot in fig.4 that all curve have the same shape, though they are not exactly aligned. The determination of $1/\Sigma$ using the relation $C^B \sim e^{-\Sigma r}/r^2$ or

$r^2 \cdot C^B \sim e^{-\Sigma r}$ at r > 13 cm, was found to be 10.39 cm for SIPHON, 9.71 cm for MCNP and 10.25 cm for the experiment. These values agree with 10.8 cm from the work of Mirza, N.M., et al.[3].

It should be noted that the Monte-Carlo efficiency of SIPHON is rather good, as shown by the figure of merit (fom) in the third column of table 2. However, there are somewhat difference between the result of SIPHON and MCNP, which, unfortunately, we have not analysed in detail yet.

Acknowledgements

We would like to thank the free software community who contributed to every software we have used, especially BLITZ++, in our work.

References

[1] Lux, I. and L. Koblinger, "Monte Carlo Particle Transport Methods: Neutron and Photon calculations", CRC Press, Inc., Boca Raton, Florida. 517 p. (1990)

[2] Cashwell, E. D. and C. J. Everett, "Monte Carlo Method for Random Walk Problems", Pergamon Press, London. 153 p. (1959)

[3] Mirza, N. M., S. M. Mirza and M. Iqbal, "Determination of Mean Squared Slowing-down Distance for Am/Be Neutrons in Water using BF_3-Detector", *Radiat. Phys. Chem.* vol. 48 no. 4 pp. 413-417 (1995)

[4] Kludge, H. and K. Weise, "The Neutron Energy Spectrum of a ^{241}Am-Be(α,n) Source and Resulting Mean Fluence to Dose Equivalent Conversion Factors", Radiation Protection Dosimetry, vol. 2 No. 2 pp. 85-93 (1982)

[5] Briesmeister, J. F., "MCNP—A General Monte Carlo N-Particle Transport Code Version 4B", Radiation Information Computational Center (RSICC), Oak Ridge, U.S.A., 709 p. (1997)

[6] Eckel, B., "Thinking in C++", Prentice Hall, New Jersey, 813p. (1995)

[7] Goldstein, H., "Classical Mechanics", Addison-Wesley, U.S.A., pp.164-165. (1980 2nd Ed.)

[8] Foley, J. D., et al., "Computer Graphics: Priciples and Practice", Addison-Wesley, U.S.A., pp.712-714. (1992)