The Monte Carlo Method: Versatility Unbounded In A Dynamic Computing World
Chattanooga, Tennessee, April 17–21, 2005, on CD-ROM, American Nuclear Society, LaGrange Park, IL (2005)

A CRITICAL REVIEW OF THE WEIGHT WINDOW GENERATOR IN MCNP

J. Eduard Hoogenboom and Dávid Légrády
Delft University of Technology
Mekelweg 15, 2629 JB Delft, The Netherlands
j.e.hoogenboom@tnw.tudelft.nl; d.legrady@tnw.tudelft.nl

ABSTRACT

Weight windows are often used in Monte Carlo calculations to increase their efficiency. However, choosing suitable values for the upper and lower bounds of the weight windows at different spatial regions and energy groups is not straightforward. MCNP provides a weight window generator to determine these parameters from a Monte Carlo run. The definition of a particle importance and its implementation in MCNP is discussed as well as the method to convert them into weight window lower bounds.

In this paper the various assumptions and the implementation of the weight window generator in MCNP will be reviewed. It is concluded that the weight window generator does not really estimates the well-known importance function. Furthermore, it is shown that the choice of the weight window bounds inversely proportional to the importance function (or the quantity estimated in MCNP) is not the optimum solution.

This is numerically demonstrated for a simplified one-group two-direction transport model for which the adjoint function and other quantities can be calculated analytically. Moreover, coupling the Monte Carlo calculation to a minimization procedure insight is obtained in the optimum setting of the weight window bounds (for a given number of space bins). The results are compared with the MCNP solution using the weight window generator and when using the true importance function.

The limitations of the use of the importance function as a basis for weight window bounds are especially relevant in the case of a coupled neutron-photon problem. Due to the difference in averaged CPU time for a neutron and a photon history, it is advantageous to use a multiple of the photon weight generated in an (n,γ) reaction.

Key Words: Monte Carlo, weight window, importance function, optimization

1 INTRODUCTION

A weight window is generally used in a Monte Carlo calculation to control the statistical weight of a particle during its history in order to increase the efficiency of the calculation. A particle is split if its weight is above the upper bound of the weight window and a Russian roulette is played when the particle weight is below the lower bound of the weight window in such a way that after splitting or surviving the Russian roulette the particle weight falls within the weight window. The upper and lower bounds of the weight window may be space and energy or time dependent, and in some cases also direction dependent.

As it is generally difficult to obtain suitable values for the upper and lower bounds as a function of space and energy, the general purpose Monte Carlo code MCNP provides a weight window generator with which these bounds can be estimated from a Monte Carlo calculation. The driving principle is that these bounds should be inversely proportional to the importance (or adjoint) function at the space and energy bin considered [1,2]. This is accomplished by estimating the expected score or importance of particles. Then the weight window lower bound is set inversely proportional to the estimated importance (provided the estimate is non-zero) with a certain normalization.

2 REVIEW OF DEFINITION AND IMPLEMENTATION

2.1 Definition of importance

The MCNP manual [1] gives the following definition of importance: the importance of a cell can be defined as the expected score generated by a unit weight particle after entering the cell. The manual provides the following formula "after a little bookkeeping"

$$\begin{array}{c}\text{importance}\\ \text{(expected score)}\end{array} = \frac{\begin{array}{c}\text{total score because of particles (and}\\ \text{their progeny) entering the cell}\end{array}}{\text{total weight entering the cell}} \tag{1}$$

The term cell here means a part of the phase space, both in geometric space as in energy (or time). From the actual implementation in MCNP4C it becomes clear that the term entering the cell (both in the numerator as well as in the denominator) should read: entering the cell *for the first time*. Moreover, entering the cell means either being released from the source in that cell or entering the cell on its flight when crossing the surface encompassing the (geometric) cell, both with an energy in the range defined by the phase space cell. This function is not exactly equal to the usual definition: the expected value of detector response or score of particles leaving the source or a collision within the cell. In this last case a transport equation can be derived to mathematically define the importance function, which turns out to be an equation adjoint to the normal particle transport equation for the particle flux. In passing it can be noted that it is also possible to define the importance of particles entering a collision. This also leads to an equation adjoint to a transport equation but now for the so-called emission density [3].

The formulation of a transport equation defining mathematically the quantity described in Eq.(1) is by far not obvious because of the possibility that particles enter the cell through its boundary surface. Nonetheless it can be estimated in a Monte Carlo calculation. The actual implementation in MCNP4C (and very probable also for MCNP5) is that a geometric mesh is defined covering the whole geometry and a set of energy groups (or time intervals). Together these define the phase space cells. For each cell two different quantities are accumulated during the simulation of all particle histories. The first quantity is the accumulated weight of particles entering the cell for the first time (either as a source particle or entering it during transport to its next collision site and crossing the cell boundary). To keep track of which cells has been entered already for the first time for a particle during a certain history, a flag is set if a particle enters a cell in which it has not been yet. The second quantity accumulates the score values of particles that entered a certain cell for the first time. For a coupled neutron-photon problem these quantities are stored separately for neutrons and photons.

The user of MCNP can choose for geometric cells identical to the material cells defined in the input or to introduce an independent mesh system in either x-y-z or r-z-θ geometry. In the first case MCNP will signal when a particle crosses the surface defining the material cell and its weight will be accumulated if it has not been in this cell before. In the case of a separately defined mesh system, MCNP does not signal the possible crossing of a mesh boundary defining a mesh cell, unless this boundary surface coincides with a material cell surface. Only at a collision site or at a material surface crossing event it is checked in which mesh cell the particle is and counters are updated if the particle has not been in the cell before. As the particle does not change its weight flying from the mesh cell surface to the collision site or material surface crossing (except when using exponential biasing), this does not invalidate the algorithm. However, it is possible with small sized mesh cells that the crossing of a mesh cell is not notified and that the update of the particle weight entering that cell is missed. To limit this effect a provision is included in MCNP to restrict the flight path in case of using a mesh system for weight windows to a maximum of one mean free path at the current energy of the particle. This does not fully exclude the effect in all cases.

2.2 Relation between importance and weight window lower bounds

After estimation of the importances for each cell the weight window lower bounds in MCNP are assigned inversely proportional to the importance of that cell. The MCNP manual does not discuss this relation, although there is no rigorous theory leading to this result. A heuristic argument is that in phase space regions with high importance it is likely (relatively to other regions with lower importance) that the particle in that region will give a contribution to the score. Therefore the particle should not be killed by a Russian roulette and hence the lower bound of the weight window should be low. It may be even effective to split the particle and to accept the longer CPU time in order to maximize the score from this particle. This is just what the weight window tries to control. There are sometimes more theoretical arguments used to justify the inverse relation with the importance, but these arguments are not correct [3]. This becomes obvious from the fact that the effect of splitting and Russian Roulette essentially depends on the CPU time spend on the additional fragments from splitting or the time saved by not having to proceed with a particle killed by Russian Roulette. The CPU time is not taken into account in any of these arguments.

In Sect. 5 we will show very specifically the influence of difference in averaged CPU time for neutron and photons on the search for optimum weight control.

2.3 Practicalities

As the weight window lower bounds obtained with the weight window generator are themselves statistical estimates with an unknown standard deviation it is difficult to judge the reliability of the lower bound values. Therefore, the MCNP manual gives some advice and the code itself can signal lower bound values that are too much different from neighboring cell values so that the user can adapt certain values. It is also discussed that an iterative process can be followed in which the weight window bounds found in a previous run are used in a next run to transport more particles to important regions and to improve the statistical accuracy of the weight window bounds. However, apart from improving the weight window lower bound estimate for cells with relatively high importance, this procedure will deteriorate the estimates for cells with low importance, which may lead to setting the lower bound to zero if no score is made for a particle entering that cell for the first time, indicating in an MCNP run that the weight window cannot be used. As it is important to cut off histories in regions with low importance, because time is spent to a particle that has a low probability of ever contributing to the detector score, one can argue whether you always gain efficiency by performing iterations on the estimation of the weight window generation.

Neither discussed is the possibility of choosing a different source distribution for the Monte Carlo run in which the weight window generator is used. As it is not the intention of the Monte Carlo run using the weight window generator to get a good estimate of the detector response, but rather to get good values for the weight window setting, it is not necessary to use the actual physical source distribution in the weight window generator run. Therefore, one can choose such a source distribution (in space and energy) that more particles are started in cells for which one wants a good estimate of the weight window bounds.

3 A CASE STUDY FOR A SIMPLIFIED TRANSPORT MODEL

3.1 The one-group two-direction transport model

As it is in most cases almost impossible to derive theoretically the expression for the neutron flux distribution and for the adjoint function, we resort to a very simplified transport model that still contains the most essential properties of neutral particle transport for a Monte Carlo calculation. We therefore consider monoenergetic particles which can move only in two opposite directions [3]. Although a piecewise heterogeneity can be handled, we confine ourselves here to an infinite homogeneous medium. The transport equation for the particle flux $\phi(x)$ in this model with isotropic scattering (i.e. both directions are equally probable) is given by

$$\Sigma_t \phi(x) = S_1(x) + \frac{1}{2} \int \Sigma_s \mathrm{e}^{-\Sigma_t |x-x'|} \phi(x') dx'. \tag{2}$$

Here Σ_s and Σ_t are the scattering and total macroscopic cross section and S_1 is the source of first collisions. The major advantage of this model is that this integral equation can be transformed into a diffusion equation without approximation. From the differential equation analytical solutions can be easily obtained.

$$-\frac{1}{\Sigma_t} \frac{d^2 \phi(x)}{dx^2} + \Sigma_a \phi(x) = S(x). \tag{3}$$

3.2 Analytical Solution for a Source-Detector Problem

We consider the geometry of Fig. 1 with a flat source between $|x| \le x_S$. The solution for the collision density in the one-group two-direction model is

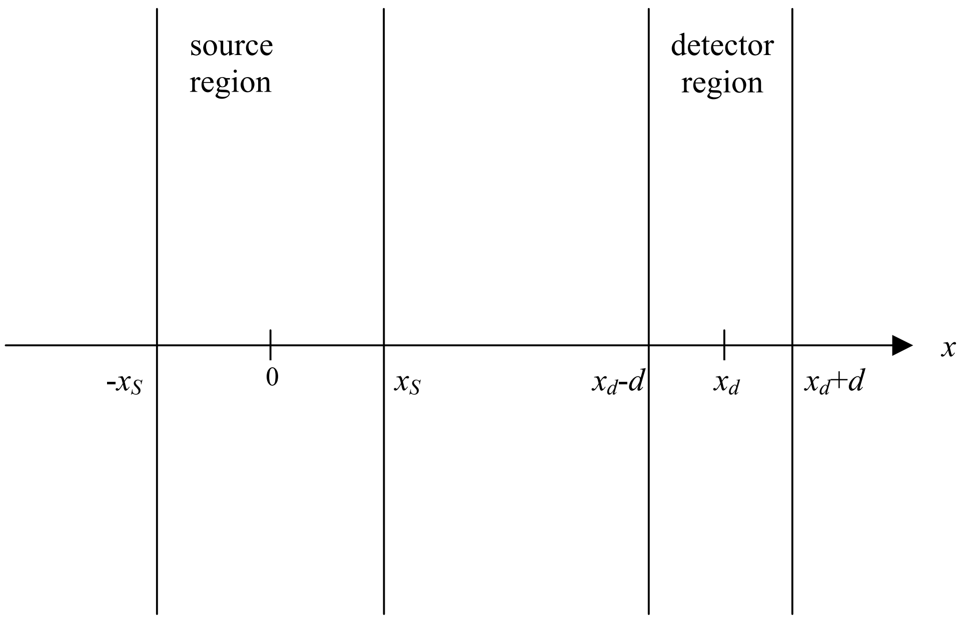

Fig. 1. Source and detector geometry

$$\phi(x)=\frac{1}{2x_S\Sigma_a}\begin{cases}1-\mathrm{e}^{-\sqrt{\Sigma_a\Sigma_{tr}}x_S}\cosh\sqrt{\Sigma_a\Sigma_t}x & |x|\le x_S\\ \sinh\sqrt{\Sigma_a\Sigma_t}x_S\;\mathrm{e}^{-\sqrt{\Sigma_a\Sigma_t}x} & |x|>x_S\end{cases}\tag{4}$$

For a homogeneous detector from x_d-d to x_d+d registering the averaged particle flux we have for the detector response R

$$R=\frac{1}{2d}\int_{x_d-d}^{x_d+d}\phi(x)dx=\frac{1}{2x_Sd\Sigma_a\sqrt{\Sigma_a\Sigma_t}}\sinh\sqrt{\Sigma_a\Sigma_t}x_S\sinh\sqrt{\Sigma_a\Sigma_t}d\;\mathrm{e}^{-\sqrt{\Sigma_a\Sigma_t}x_d}\tag{5}$$

As Eq.(3) is self-adjoint we can obtain the adjoint function $\phi^+(x)$ from an equation like Eq.(3) with the source $S(x)$ replaced by the detector response function $D(x)=1/(2d)$. This results in

$$\phi^+(x)=\frac{1}{2d\Sigma_a}\begin{cases}1-\mathrm{e}^{-\sqrt{\Sigma_a\Sigma_t}d}\cosh\sqrt{\Sigma_a\Sigma_t}(x-x_d) & |x-x_d|\le d\\ \sinh\sqrt{\Sigma_a\Sigma_t}d\;\mathrm{e}^{-\sqrt{\Sigma_a\Sigma_t}|x-x_d|} & |x-x_d|>d\end{cases}\tag{6}$$

One can easily verify that the detector response can also be obtained from $R=\int S(x)\phi^+(x)dx$. $\phi^+(x)$ is the adjoint function for particles leaving the source or a collision. It is also possible to define the adjoint function for particles entering a collision. Then the adjoint source function must be chosen equal to the detector response functions for particles leaving a collision, which becomes in this case

$$S^{+}(x)=\frac{1}{\Sigma_t}\frac{d^2D}{dx^2}-\Sigma_t D(x) \tag{7}$$

and the adjoint function becomes

$$\psi^{+}(x)=\frac{\Sigma_s}{2d\Sigma_a\Sigma_t}\begin{cases}\Sigma_t/\Sigma_s-\mathrm{e}^{-\sqrt{\Sigma_a\Sigma_t}d}\cosh\sqrt{\Sigma_a\Sigma_t}(x-x_d) & |x-x_d|\le d\\ \sinh\sqrt{\Sigma_a\Sigma_t}d\ \mathrm{e}^{-\sqrt{\Sigma_a\Sigma_t}|x-x_d|} & |x-x_d|>d\end{cases} \tag{8}$$

Fig. 2 displays the particle flux and both adjoint functions. Note that the adjoint function $\phi^{+}(x)$ is everywhere continuous, but its derivative is discontinuous at $x=x_d\pm d$, while $\psi^{+}(x)$ is discontinuous at $x=x_d\pm d$ and its derivative is everywhere continuous.

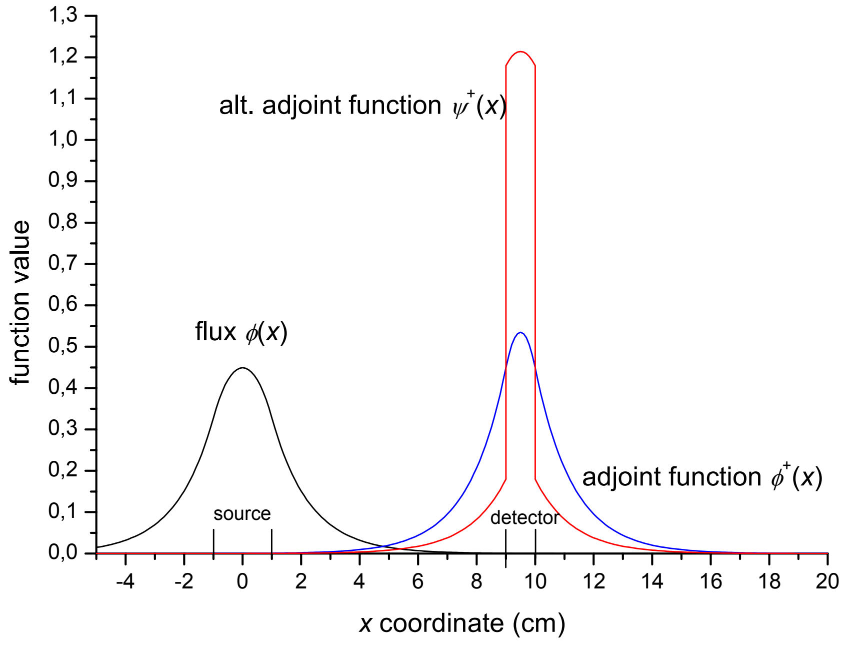

Fig. 2. Particle flux and detector adjoint functions

3.3 Numerical results

To get numerical results we made the following choices: x_s=1 cm, x_d=9.5 cm and d=0.5 cm. Hence the detector ranges from 9 to 10 cm. The total cross section of the medium was taken as 1 cm^{-1} and the scattering cross section as 0.4 cm^{-1}. Then from Eq.(5) the detector response is R = 4.6501 10^{-4} for a unit source.

To be able to introduce various options we developed our own relatively simple Monte Carlo program to treat the geometry of Fig. 1 and the two-direction model. To compare results with MCNP and especially the weight window generator in MCNP we made some minor modifications to MCNP to accommodate the two-direction model. To be able to use the code also in its original form we added an input variable determining whether the two-direction model should be used. If so, the direction of the particles after generating the source variables of a particle and after dealing with a collision was reset to the *+X* or *–X* direction depending on the component of the originally selected direction by MCNP along the *X*-axis. In order to run a one-group problem with the selected cross sections we composed a separate cross section library in ASCII format following the format defined in the MCNP manual for multigroup libraries. This cross section library needs only a few lines.

3.3.1 Reference calculations

Table I shows the results of the detector response estimate with our own program and from MCNP with the one-group two-direction model, each using 10^6 histories. In MCNP the standard variance reduction options are used, i.e. implicit capture and playing Russian roulette if the particle weight is below 0.25 with a survival weight of 0.5. The same was implemented in our own program.

Table I. Comparison of MCNP4C3 run with our own code

Code	Detector response	Standard deviation (%)	CPU time (min)	FOM (min^{-1})
MCNP4C3	$4.653\ 10^{-4}$	0.35	71.3	1145
own MC code	$4.666\ 10^{-4}$	0.35	7.96	10255

The table shows a satisfactory agreement, also with the theoretical value, which validates both the two-direction option in MCNP and the private Monte Carlo program. The column at the right shows the Figure of Merit (FOM), equal to the inverse of the product of the CPU time of the calculation and the square of the relative standard deviation. This quantity is a measure for the efficiency of the calculation when compared with other calculations for the same problem using different options to carry out the Monte Carlo simulation. The CPU times and hence the FOM values of both codes cannot be compared as the general purpose code MCNP has to deal with a lot of overhead in this very simplified case. The FOM values are used only for comparison with other runs with the same code.

3.3.2 Results with weight windows

Next we made an MCNP run with the weight window generator activated. To this end spatial regions are defined each with a width of 1 cm and ranging from x=-5 cm to x=+15 cm, so constituting 20 different weight window regions. The areas to the left and to the right of these regions are in fact also defined as weight window regions, but as the particle density is very low here, they do not play a significant role. The normalization of the lower weight window bounds was chosen as 0.25 for that part of the source region from x=0 to 1 cm. The value of 0.25 was chosen in accordance with the limit for playing Russian roulette in a standard MCNP run. The run for generating weight windows gives, of course, the same results as above, apart from the about 10 % longer CPU time. Fig. 3 shows the estimated importances by MCNP for the various regions. They can be compared with the theoretical values by averaging the values of Eq.(8) over each region. Taking into account the logarithmic scale of Fig. 3 there are clear differences between the two types of importance values. The MCNP-estimates are not symmetric with respect to the detector and are considerably lower at the right of the detector. Note that the MCNP-estimates are random variables with unknown standard deviation.

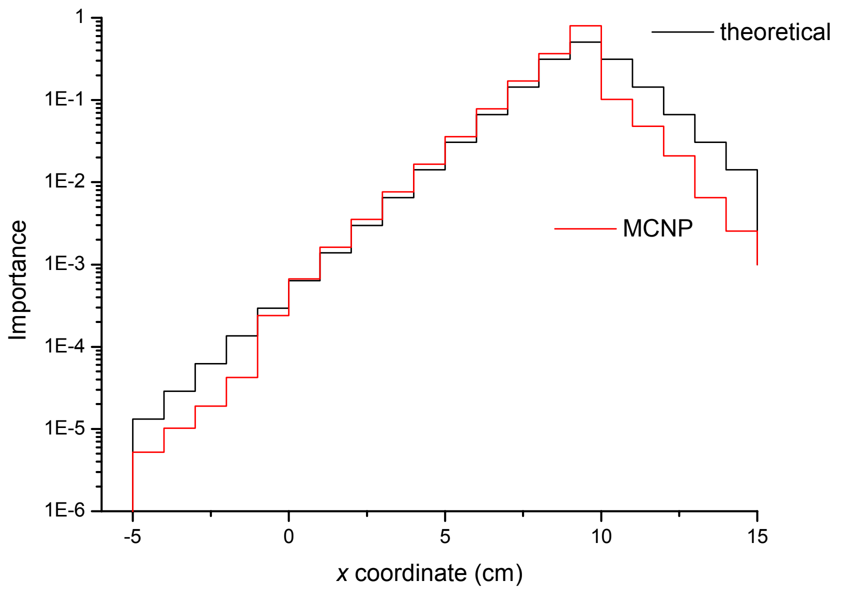

Fig. 3. MCNP-estimated and theoretical importances

An MCNP run was made with the weight window data generated in the previous run (row "with ww from wwg"). For a comparison these weight window lower bounds were also used in our own MC program. Table II shows the results. From a comparison with the FOM values from Table I for the same code it is clear that the weight window provides a big improvement in

Table II. Calculations using weight window lower bounds from the MCNP weight window generator and theoretical values

Code	Detector response	Standard deviation (%)	CPU time (min)	FOM (min^{-1})
MCNP4C3 with ww from wwg	$4.6520\ 10^{-4}$	0.037	145.9	49660
own MC code with ww from MCNP wwg	$4.6484\ 10^{-4}$	0.037	22.65	319,040
MCNP4C3 with ww from theory	$4.6511\ 10^{-4}$	0.038	136.4	50800

efficiency, about a factor of 43 for MCNP and about 31 for our own code.

The table also shows the results when the theoretical averaged importance values for each region are transformed to weight window lower bounds and used in the calculation (see row "ww from theory"). If the statement is true that the adjoint function is the best basis for determining the weight window parameters, this should give a further improvement in FOM, which is indeed the case but only marginally.

4 NUMERICAL OPTIMIZATION OF WEIGHT WINDOW PARAMETERS

If we ask for the optimum distribution of the weight window parameters over the various cells, we can consider the weight window lower bound for each geometric cell as an independent variable. We keep the ratio of the upper to the lower bound as a fixed value of 5. The weight of the particle after survival of the Russian roulette is taken as 2 times the lower bound. These are default values in MCNP. For the geometry dealt with in Sect. 3 there remain 20 independent variables to be optimized. Because of the limited number of variables it should be possible to look for numerical optimization of these parameters. To that end we made around our own Monte Carlo code a shell for minimization of a function, with a call to the Monte Carlo subroutine to provide a function value. An obvious choice for the function value is the product of the CPU time of the Monte Carlo run and the square of the relative standard deviation in the estimated detector response, i.e. the inverse FOM.

However, coupling a Monte Carlo calculation to an optimization procedure presents special problems, as the outcome of the Monte Carlo calculation is a stochastic quantity. The minimization routine will normally take decisions on basis of small differences in function values due to slightly different input variables but the stochastic nature of the function value will obscure differences due to input values.

To overcome this problem we took several measures in the Monte Carlo calculation: for every run the same initial random number for the histories was used. Moreover, as in MCNP, for a history the same starting random number, derived from the initial random number, in successive calculations was used. This introduces correlations between successive calculations with different weight window parameters, but in this case this is just what we need to see the effect of varying weight window parameters. Moreover, the CPU time of a certain Monte Carlo run is neither a fully deterministic quantity. Repetition of the same Monte Carlo run with the same random numbers show small differences in CPU time. A test with 10 identical runs show a standard deviation in the CPU time of 0.3 %. Although this negligible for normal applications, it deteriorates the minimization procedure Therefore, instead of the computer CPU time to calculate the (inverse) FOM, we took the averaged number of collisions N_c, which was also calculated during the Monte Carlo calculation. It is a reasonable assumption that the CPU time is proportional to the averaged number of collisions. The CPU time will also depend on the number of surface crossings, which take some time to handle and hence from the geometry. But in our case, where the geometry is fixed when looking for optimum weight window parameters, that is not a problem. The averaged number of collisions for a Monte Carlo run with a finite number of histories is itself a random variable, but only dependent on the sequence of random numbers used in the Monte Carlo run, just like the estimated detector response and its relative standard deviation, while the CPU time turns out to have an independent source of fluctuations. Hence, we used as the function value to be minimized

$$f = \sigma_{rel}^2 . N_c \tag{9}$$

At first we tried a standard minimization routine from the NAG library for the case that no information is available about derivatives. However, it turned out that such routines are not suitable to solve our problem. Therefore, we programmed our own algorithm to change one or more parameters after having obtained a new function value from a Monte Carlo run with a given set of parameters. Initially guessed values of all parameters must be supplied and the function value for this case is determined, as well as for all cases where the parameters are successively changed a small amount. From this information it is determined which parameters should be changed to find a lower function value. The smaller the change in function value is, the smaller the changes in one or more parameters is chosen. As happens frequently in such minimization problems, one can get stuck in a local minimum. Therefore, starting with the parameters from such a local minimum, the whole process can be restarted with larger changes in the parameters than used when approaching the local minimum. This may make it possible to step out of the local minimum and find an even lower function value. As an alternative one may restart the minimization procedure with one or more of the parameters found for the minimum changed rather arbitrarily. This often leads to a new and somewhat different optimum set of parameters, but with a function value hardly different from the previous local minimum.

Because the problem at hand is a non-linear problem with respect to the weight window parameters, no guarantee can be given that the absolute minimum will ever be reached. On the other hand, we are not interested in finding the absolute minimum as long as other local minima give a function value that is only marginally larger than the absolute minimum. An additional gain in efficiency of say 1 % will seldomly be seen as relevant. Because of the physics of the problem we can expect that if the weight window parameters deviate much from a ‘reasonably good’ solution, no smaller function value will be obtained. Hence, we may be satisfied with a certain solution, even if we don’t know whether it gives the absolute minimum for the function value.

We started with the weight window lower bounds for each cell as obtained from the weight window generator in MCNP. First we searched for the optimum value of the normalization constant determining the lower bounds from the estimated importance values. This is in fact a search for the optimum value of the lower bound in the reference source cell, which was chosen in our case as the source cell from x=0 to 1 cm. We took a few different starting values to circumvent local minima. This resulted in an optimum normalization constant 2.6447 times higher than the choice in the weight window generator, i.e. the weight window lower bound in cell 6 becomes 0.6612 instead of 0.25. It should be noted that although this is a drastic change with respect to the MCNP-based lower bounds, the function value and hence the efficiency of the Monte Carlo calculation only changes about 4 %. This means that the choice of the weight window lower bound in the reference (source) cell is not critical, at least as long as it remains within a certain, rather wide, range.

Next, we tried to use this distribution of lower bounds as a starting point for optimization of all the lower bounds in the 20 cells. However, this gave only very small changes in de lower bounds and a very small improvement in the function value. Therefore, we chose a stepwise optimization process, starting with only one parameter to be optimized, namely the weight window lower bound of the cell with the detector (cell 15). Using the optimum value from this case, we started a two-parameter optimization for the weight window lower bounds of cells 14 and 15, and so on. As many iterations have to be performed, especially when 20 parameters must be optimized, we did all the Monte Carlo runs with 10^5 histories to limit the execution time. The changes in the spatial distribution compared to those of the MCNP weight window generator are small as is the change in function value. However, this does mean, other than the overall normalization value, that lower bounds in a cell can be changed rather arbitrarily without serious penalty. A change in one or more of these values may lead to a serious loss in efficiency.

Table III. Comparison of different weight window distributions with our own MC code and 10^5 histories

Case	Detector response	Standard deviation (%)	CPU time (s)	Number of collisions N_c	$1/f$ (s^{-1})
ww from MCNP wwg	$4.68\ 10^{-4}$	1.18	1.35	7.285	0.0098
ww from MCNP wwg with optimum renormalization	$4.60\ 10^{-4}$	1.38	0.98	5.187	0.0102
best ww distribution found	$4.55\ 10^{-4}$	1.36	0.99	5.186	0.0105

5 A COUPLED NEUTRON-PHOTON PROBLEM

Another type of problem to discuss the basis of the MCNP weight window generator is a coupled neutron-photon problem. In such a problem the importance for all spatial regions and energy groups is estimated both for neutrons and photons. Then the weight window lower bounds are calculated using the same normalization factor for neutron data and photon data, derived from the requirement that the weight window lower bound for source particles, hence neutrons, is a given value, for instance 0.5. This seems reasonable because the ratio of photon and neutron importances is conserved.

In the MCNP input one should indicate for which cell the normalization constant of the inverse importance should be determined to get a given weight window lower bound. However, the neutron energy group for which this value should be arranged cannot be entered by input. In fact, MCNP chooses that energy group for which the estimated importance is maximum (still restricted to the source cell given by input). Even if the source energy distribution is restricted to a single weight window energy group, there is no guarantee that the estimated adjoint is maximum for that source energy group. It often happens that the maximum occurs for the energy group just below the source energy group. This means that the source energy group is not assigned the value aimed at. Especially if the importances for the different energy groups differ a lot, this may introduce unwanted effects, as the source particle weight may not fall in the weight window assigned to the source energy group. This leads to unwanted splitting or Russian roulette of source particles. It would therefore be desirable to input not only the spatial source cell for which the weight window lower bound should get a specified value, but also the energy group for which this should apply.

To illustrate the effect of the weight window based on importance values we took a borehole logging problem [6] with a neutron source and two photon detectors in the logging tool. The borehole logging problem is known to be a difficult problem for Monte Carlo variance reduction [7]. In this case we used a separate mesh system to define the cells with weight windows. Then the weight window lower bounds should be entered from a separate file, which can be generated by the MCNP weight window generator. However, we used a deterministic calculation with the discrete ordinates code DORT [7] in r-z geometry to calculate the importance function for the 'far' detector. Then the importance functions were converted to weight window lower bounds in the proper file format. This has the advantage that a detailed weight window map can be obtained, both in geometry and in energy with a smooth behavior. In this case we used a 175-neutron group and a 45-photon group calculation, with 44 radial mesh intervals and 79 axial mesh intervals. This will not be possible from the weight window generator, as the more cells are used, the worse the statistical reliability will be of the estimated weight window lower bounds and especially many zero values will be recorded in those cases where MCNP does not get a score for particles entering that cell for the first time.

Table IV shows the photon flux at the far detector from a standard MCNP calculation without using weight windows and when using the weight windows derived from the DORT calculation. An increase in efficiency of a factor of more than 9 is obtained. Note that for the photon weight window lower bounds the same normalization constant is used as for the neutron lower bounds, as would have been the case when using the MCNP weight window generator. Next the weight window lower bounds for photons are all decreased by a factor of 10. Then another factor of 1.5 in efficiency gain is obtained. This is caused by the fact that photons are more frequently split after their generation, because of the lower bounds of the weight window for photons. This also increases the computer time, but the CPU time to process a photon is much smaller than that for neutrons, so the net effect is positive.

Table IV. Efficiency of MCNP calculations using importance functions from DORT

Case	Detector response	Standard deviation (%)	CPU time (min)	FOM (min^{-1})
without ww	1.989 10^{-6}	9.2	13.06	9.05
with ww	1.963 10^{-6}	3.7	8.62	84.7
with ww x 0.1 for photons	1.852 10^{-6}	2.8	10.12	125.1
with ww; generated photon weight x 10	1.953 10^{-6}	1.5	13.00	346.5

The processing time of a photon is much smaller than that of a neutron because a photon, on the average, has a larger mean free path and hence, less collisions than a neutron. This can be illustrated by running the case without weight windows as a neutron problem only. Then the CPU time is 11.64 min. So, the processing time for photon histories is only 12 % of that of a neutron. These numbers will change somewhat when applying weight windows. Anyway it is useful to use more photons. In some Monte Carlo codes it is possible to use each generated photon a number of times, to be entered by input. As an equivalent we introduced in MCNP the possibility to multiply the generated photon weight at a neutron collision by a factor given by input and renormalizing the detector response by that factor. The result is also shown in Table IV and provides an efficiency gain factor of 4 with respect to the case with straight forward weight windows and a factor of 38 with respect to the standard case.

6 CONCLUSIONS AND RECOMMENDATIONS

In this paper the use of the importance function of a particle for determining the value of the lower bound for a weight window is discussed. It is shown that the weight window generator does not actually estimate the generally accepted adjoint function, as it also counts particles entering a cell for the first time by crossing one of its boundary surfaces from an adjacent cell. However, it turns out in our case study that the differences are not important for the efficiency gain to be obtained when applying the weight window parameters from the weight window generator results. It has also been shown that the choice of a value of the weight window lower bound in a source cell to normalize the calculation of weight window lower bounds from the estimated importance values is not very critical. We believe that these conclusions have a wider application than the case studied here.

From an example of a coupled neutron-photon problem it is shown that the same normalization of the weight window lower bounds for neutrons and photons is far from an optimum solution. Because of the smaller CPU time needed for processing a photon history, the weight window bounds for photons should be chosen much lower than follows from the MCNP weight window generator. It is also shown that providing a new option in MCNP to multiply the generated photon weight by a factor to be entered by input and normalizing the tally result by this factor can lead to a big additional improvement in efficiency.

For an energy dependent calculation (not only coupled neutron-photon problems) it is recommended to provide an input option in MCNP to specify not only the spatial cell for which the weight window lower bound must be assigned a specific value, but also to specify the (source) energy group.

7 ACKNOWLEDGMENTS

Part of this research was performed in the framework of the European Union project Improved Radiation Transport Modelling for Borehole Applications.

8 REFERENCES

[1] Judith F. Briesmeister, “MCNP – A General Monte Carlo N-Particle Transport Code, Version 4C, report LA-13709-M, Los Alamos Scientific Laboratory (2000).

[2] T. E. Booth and J. S. Hendricks, “Importance Estimation in Forward Monte Carlo Calculations”, *Nucl. Technol./Fusion* , **5**, 90 (1984).

[3] J. E. Hoogenboom, “Monte Carlo Zero-Variance Schemes: Theory, Demonstration and Practical Consequences,” *Proceedings of MC2003*, Gatlinburg, April 6-10, 2003, CD-ROM paper 049.

[4] I. Kodeli, et al., “Generation and Performance of a Multigroup Coupled Neutron-Gamma Cross-Section Library for Deterministic and Monte Carlo Borehole Logging Analysis,” *Proceedings of PHYSOR2004*, Chicago, April 25-29, 2004, CD-ROM.

[5] R. P. Gardner and L. Liu, “Monte Carlo Simulation of Neutron Porosity Oil Well Logging Tools: Combining the Geometry-Independent Fine-Mesh Importance Map and One-Dimensional Diffusion Model Approaches,” *Nucl. Sc. Eng.* **133**, 80 (1999).

[6] W. A. Rhoades et al., “DOORS 3.2, One-, Two- and Three-dimensional Discrete Ordinates Neutron/Photon Transport Code System, CCC-650”, Radiation Safety Information Computational Center, Oak Ridge National Laboratory (1998).