Monte-Carlo Techniques: The detection of a 0.663 MeV gamma-ray

Update: January 14, 2021

Contents

1 Introduction

By now, you probably know that as a scientist, just doing an experiment (i.e. taking data) is not generally enough. We also need some kind of theory to go with the data, even if it just serves as a guide for our lab work. This is true even if you find yourself to be an "experimentalist" in the "experimentalist" vs "theorist" aspects of physics.

To this point in your education, usually a curve fit serves as your theory. We all know how satisfying it is to see the smooth line of a theoretical equation go right through a set of data points. We extract the fit parameters and can often come to some firm conclusions from them.

In this lab, we'll investigate where a theory might come from for very complicated data, like a Cs137 gamma ray spectrum shown here.

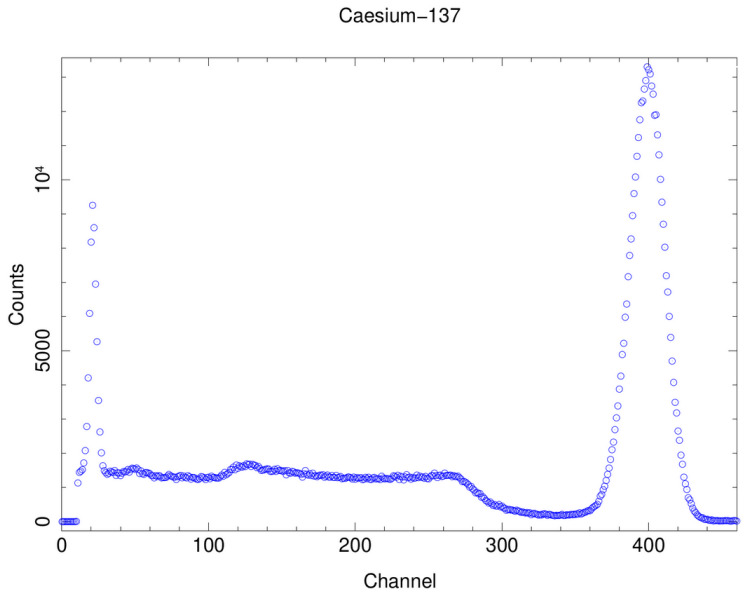


Figure 1: Cs137 energy spectrum.

Would you expect a to see a single equation form a smooth line that passes through all of these data points, predicting all of the peaks and valleys of this data? That is, do you think you can find a function $f(E)$ that allow us to plug in some energy E, and have it reproduce every hill and valley in Fig. 1?

No. It turns out your only hope to make a theoretical recreation of such a spectrum is to simulate the entire process of what happens when a gamma ray goes into a detector. If you

run the simulation for 1000s of simulated gamma rays, maybe you'll be able to produce a *simulated* spectrum that shows some agreement with Fig. 1.

The simulation will likely involve multiple steps in how a gamma ray interacts with a detector, that you know are all physically correct, each driven by their own theory (and likely an equation). Maybe stringing all of these smaller theories together can collectively produce the Cs137 spectrum.

You know for instance that when a gamma ray goes into a detector, it may interact with an electron. And when it does, it will either Compton scatter or photoelectric absorb (if $E < 2m_0c^2$).

This work will take you through the creation of a computer simulation that will allow you to create a simulated Cs137 spectrum, gamma ray by gamma ray. Such simulations are very instructive too. As you string the theories together in your code, you'll have the opportunity to gain a deeper understanding of the processes involved.

1.1 Monte-Carlo Simulations

Tracking a gamma-ray through a detector is complicated. One obvious question is: how does the gamma ray "decide" what to do? Even if you are armed with your theories of Compton scattering, photoelectric absorption, and pair production, what now?

Well it turns out that a complicated system like this is driven in a simulation by random numbers to serve as the "chance of nature." It is true, that when a gamma-ray approaches an material full of electrons, it may pass right through, Compton scatter, or photoelectric absorb. We don't know what it'll decide to do, so we we pull random numbers to decide for us. This actually means we will internally roll a dice to decide what the gamma ray will do, and if we do this enough times, maybe a gamma ray spectrum will evolve. Keep in mind though, that the likelihoods of the possible processes are not all equal, so we'll have to weight them carefully based on physics (which is a part of our overall theory).

The word "Monte-Carlo" comes from the name of a casino in Monaco. It refers to the use of random number to simulate the chance of nature in computer code (and "chance of nature" obviously has a lot to do with gambling). Also, 'inspirations for numerical computing can spring from unlikely sources...and who does not feel at least a faint echo of glamor in the name "Monte Carlo" method?"[1]

[1]Press, *Numerical Recipes in C*, Sect. 7.6, p. 304.

2 Monte Carlo Warm-up

2.1 Your software's random number generator

You may not know that the software you use to do your data analysis (Matlab, Python, etc.) can generate random numbers. (Have you every use a computer-generated random number for anything in the past?)

Go into your data analysis software and write some code to sample 1000 uniform random numbers between 0 and 1. (Matlab: `rand`, Python: `import random` then `random.random()`.) Histogram these numbers. What does the histogram look like and why? And, why would a random number between 0 and 1 be useful in simulating the chance of nature?

2.2 A simple Monte Carlo simulation to find π

It really is by no means obvious that anything sensible could come out of random numbers on a computer[2]. So, before delving into the gamma ray simulation, let's write a simple Monte-Carlo simulation to estimate π–yes π–*using random numbers*. Doing so will give you a flavor of the considerations of a Monte-Carlo simulation.

To begin, think about how you would provide a numerical estimate for π. Would you look up some infinite series and start computing and summing terms? Maybe you'd even tempt yourself with Ramanujan's formula? Let's take a Monte-Carlo approach instead.

Here's our model, which is a square with side $2r$ with a circle of radius r inscribed within it as shown here:

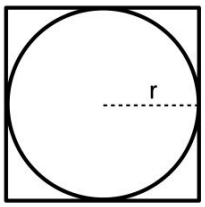


Figure 2: A circle of radius r inscribed in a square of side $2r$.

Why is this our model? Well, let's imagine this was a dartboard up on a wall and blindfolded, you started throwing darts at it after being spun around a few times. (Your dart throwing

[2]Koonin, *Computational Physics* p. 185

is the randomness of nature.) If you count the number of darts that fall within the circle (N_C) vs the total number thrown (N_T), you'd have an estimate of π. Why?

The area of the square is $(2r)^2$ and that of the circle is πr^2. When done throwing N_T darts, you can imagine the dart board full of little holes where the darts struck. You could now count up the number that fell within the circle's area vs the total number thrown at the whole area (the square's area). The ratio of the two is is $\pi r^2/(4r^2)$ or just $\pi/4$. So $N_C/N_T = \pi/4$ or $\pi = 4N_C/N_T$. So all you have to do to estimate π is take the ratio of N_C to N_T and multiply by 4. See why the areas are important in our model?

To simulate dart throws, we'll set $r = 1$, so the square size will be 2 and the radius of the circle 1. We'll find a two random numbers, each between -1 and 1. One will be the x coordinate of where a dart landed, and the other the y coordinate. This assumes the square and circle are both centered at $(0, 0)$. You'll frequently have to scale your software's built in random number function (that returns numbers from 0...1) into some other range. In this case, if u is a uniform random number for $0 \leq u \leq 1$, and we need v uniform on $-1 \leq v \leq 1$, we can use $v = 2u - 1$. (Why? $2u$ will be between 0 and 2, and if you subtract 1 from such numbers, they'll all lie between -1 and 1.)

Your logic then is to find many (x, y) pairs and count how many fall within the circle. Checking for darts that fell within the circle requires some logic too. Here we'll see if the distance of a hit (x, y) point from the origin is ≤ 1. Your code may resemble this pseudo-code:

```
Nc=0
NT=100
for i=1 to NT
    x=2*rand-1
    y=2*rand-1;
    if sqrt(x^2+y^2) <= 1 then
      Nc=Nc+1
    end-if
end-for
print Nc/NT
```

See the string of theories needed even for this? You should tweak `NT`, the total number of darts thrown. As it grows, the estimate for π should improve. This is of course at the expense of computing time and energy; try `NT` of 10,000,000 or more! It may be interesting to plot the current approximation for π as a function of (every thousand) iteration number.

You may sometimes hear of simulations requiring days to run on super-computers. Parameters equivalent to `NT` (a total number of runs needed for some reason) are usually why.

Congratulations on creating your first Monte-Carlo simulation. You computed π using *ran-*

dom numbers!

3 Part I: Monte Carlo Simulation for Gamma-Rays in the small-detector regime

Let's get going then on a Gamma-ray simulation. Like the simulation for π, let's start by looking at a model for our gamma ray simulation.

Suppose a gamma-ray enters a NaI(Tl) detector and encounters an electron. At the 0.663 MeV energy of a Cs137 gamma-ray, it can undergo one of two processes, 1) a Compton scatter (CS) or 2) Photoelectric absorption (PA).

Ultimately a gamma-ray's energy is deposited into the detector via electrons the gamma-ray interacts with. The interaction results in the gamma-ray giving the electron kinetic energy (KE), which contributes to the detector's output. In other words, the moving electrons create the detector's signal.

A PA will transfer all of the gamma-ray energy to KE of an electron and end the gamma-ray's life.

A CS will transfer some KE to an electron with the rest of the energy going into a lower energy gamma-ray. In an actual detector, this lower energy gamma-ray would go off and find another electron to interact with. Here however, this will also end the gamma-ray's life.

Allowing on only on CS assumes the gamma-ray now leaves the detector. This is the so called "small detector" regime, meaning there isn't enough detector material around to cause another CS or PA. At this point, this regime keeps the programming logic straightforward and the detector physics mostly right, while still allowing one to see how CS contributes to the overall spectrum. Your instructor's entire simulation of this for example, is less that 50 lines of Matlab. We'll relax the small-detector regime in Part II.

3.1 Simulation Steps

3.1.1 Initialization

Here are a few steps to initialize your simulation.

- Create an empty array you can use to store numbers in, one by one. We'll call this the "energy array." It will be a list of gamma-ray energies that were deposited into the detector via electron KE, as your simulation runs.

- Assign an initial gamma-ray energy, $E = 0.663$ MeV, the energy of a Cs137 gamma-ray.
- Decide how many gamma-rays you want to simulate, called N. Let's start with $N = 1000$.
- Let call c our current gamma-ray count. Set $c = 1$ to start.

3.1.2 Simulation steps

Here are the steps in your simulation.

1. Your gamma-ray has energy E. Assume it has now encountered an electron.
2. What will it do? CS or PA? These are probabilistic events, governed by the following equations. The probability of CSing is given by

$$P_{CS}(E) = 1.04713 \cdot e^{0.23e^{-0.5E}} \tag{1}$$

   and that for PAing

$$P_{PA}(E) = 1.01158 \cdot 10^{132e^{-28E}}. \tag{2}$$

   Here, $P_{CS}(E)$ is the probability to Compton scatter, and $P_{PA}(E)$ is the probability for a photoelectric absorption. Note that both are functions of the gamma-ray energy, E (in MeV).
3. To determine which will happen, compute a uniform random number r for $0 \leq r \leq 1$. If

$$r < \frac{P_{CS}(E)}{P_{CS}(E) + P_{PA}(E)}, \tag{3}$$

   then the gamma-ray will CS. If not it will PA.[3]
4. If the gamma-ray will PA:
   (a) Throw this energy into your energy array.
   (b) Increment c. If $c > N$ stop.

[3]This technique, of seeing if a random number is less than some needed distribution is known as the "Metropolis algorithm."

(c) If not, reset $E = 0.663$ (start again with a new gamma-ray).

(d) Loop back to Step 1.

5. If you are at this step, your gamma-ray is going to Compton scatter. This will result in a lower energy gamma-ray with energy E' where

$$E' = \frac{E}{1 + \frac{E}{0.511}(1 - \cos\theta)}. \tag{4}$$

But at what angle θ will you use?

6. To find the CS angle, use the Klein-Nishina Cross section. This is the probability that a gamma-ray will scatter at some angle θ, which is

$$P_{KN}(E,\theta) = \left(\frac{1}{1+\alpha(1-\cos\theta)}\right)^2 \left(\frac{1+\cos^2\theta}{2}\right) \left(1 + \frac{\alpha^2(1-\cos\theta)^2}{(1+\cos^2\theta)[(1+\alpha(1-\cos\theta)]}\right), \tag{5}$$

where $P_{KN}(E,\theta)$ is the probability that a gamma-ray of energy E will scatter through an angle θ. Note that $\alpha = E/0.511$.

7. To find the angle, choose an random angle θ for $0 \leq \theta \leq 2\pi$. Now, find a random number q for $0 \leq q \leq 1$. If $q < P_{KN}(E,\theta)$, your gamma-ray will scatter at your chosen angle θ. If not, keep choosing new values for θ and q until $q < P_{KN}(E,\theta)$. (This is the Metropolis algorithm again.)

8. Now that you have your scattering angle θ, compute the new gamma-ray energy, E' from Eqn. 4 above.

9. Compute the KE of the Compton scattered electron, which is $0.663 - E'$. Throw this KE into your energy array.

10. Set $E = 0.663$ to start over again with a new gamma-ray (this is a "small detector" simulation, where we only allow one CS per gamma-ray).

11. Increment c. If $c \leq N$ loop back to Step 1.

When $c \geq N$, make a histogram of your energy array (with 100s of bins at least). Compare your histogram with your actual Cs137 spectrum.

3.1.3 Your histogram

Hopefully in your histogram, you'll see the mono-energetic 0.663 MeV photopeak and the lower energy CS events. The power of working out the code in this simulation is that it forced you to consider the CS and PA processes that a real gamma-ray would undergo.

You also see a void between the photopeak and the start of the CS. This is sometimes called the "Compton Valley" and the high energy limit of the Compton events is called the "Compton Edge." You should convince yourself why the valley and edge exist in this "small detector" regime: there is no physical process in this model that would generate gamma rays in that energy region, if only one Compton scatter is allowed.[4]

3.1.4 Realism

Let's add a little realism to your energy-array, by dithering the energies a bit (i.e. simulate the detector's non-zero resolution). We'll dither the energies assuming random errors in the detector's ability to determine a gamma-ray energy. How?

1. Find a normally distributed random number w, with $\bar{w} = 0$ and $\sigma_w = 1$ (Matlab: `randn`, Python: `random.gauss`).
2. Compute $E_{dither} = E + 0.05wE$ from each energy E in your energy array, which assumes a 5% dither spread.
3. Show a histogram of the dithered energy-array. How does it compare to the actual Cs137 spectrum you took in your lab work?

3.2 Visualize P_{PA} and P_{CS}

Plot P_{PA} and P_{CS} vs E for $0.100 \leq E \leq 1.0$ on the same plot with a linear y-axis. Make another with a logarithmic (base 10) y-axis. From these plots, discuss the general probability of what a gamma-ray—as a function of its energy—is most likely to do when it encounters an electron. What does it mean where P_{PA} and P_{CS} cross?

[4] see Fig. 10-2, p294 in Knoll, "Radiation Detection and Measurement" 2nd ed.

4 Part II: Monte Carlo Simulation for Gamma-Rays in a larger-detector regime

In this part, we'll relax the "small detector" regime and allow multiple Compton scatters. This will produce an even more realistic Cs137 spectrum. The small-detector regime assumed that after one CS, the gamma-ray leaves the detector and will no longer contribute to the overall signal. If we want to allow for more CSs to occur we'll have to add three more considerations to our simulation: 1) impose a simulated detector size 2) consider *if and where* a gamma-ray will interact within the detector and 3) keep track of the gamma-ray's position as it lives.

4.1 Detector size

A typical NaI detector is cylindrical in shape, so it is natural to use cylindrical coordinates for containing the detector volume, as shown here.

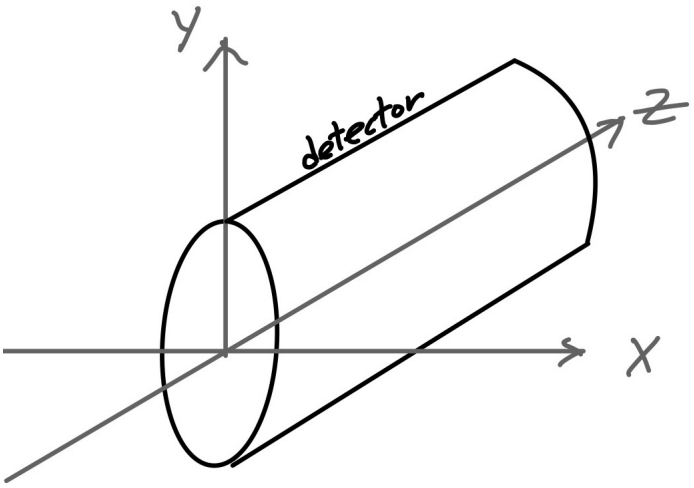


Figure 3: A cylindrical detector imposed on a cartesian coordinate system.

You can assign any length and radius to the detector as needed. We'll assume our initial gamma ray starts at $(0, 0, 0)$ and is heading straight along the $+z$-axis.

4.2 If and where it interacts

In the "small detector" regime, we always assumed the gamma would immediately interact with an electron. A PA would cause absorption and would end the gamma's life. A CS would deposit some energy into the detector, and the scattered gamma would immediately leave the detector, also ending its life.

Extending this work to finite sized detector, we'll have to now figure out where the gamma ray will interact. This will allow you to use a parameter you experimentally found in the gamma-ray attenuation lab: the mass attenuation coefficient.

As gamma-rays get attenuated through some material, the drop off in intensity is exponential, as in

$$I(z) = I_0 e^{-\lambda z}, \tag{6}$$

where λ is known as the "linear attenuation coefficient." For gamma-rays, λ is not constant and will depend on the gamma-ray's energy, so $\lambda = \lambda(E)$.

In this work, this equation can be interpreted as the probability that a gamma-ray will interact in some material, after traversing it by some depth z. In this case, the normalized probability distribution of an interaction or $f(z)$ would be

$$f(z) = \lambda e^{-\lambda z}, \tag{7}$$

and the cumulative probability of interacting after traversing some distance z would be $F(z)$ or

$$F(z) = \int_0^z f(z)dz. \tag{8}$$

Filling things in, we get

$$F(z) = \int_0^z \lambda e^{-\lambda z}dz, \tag{9}$$

which is an integral we can do to get

$$F(z) = 1 - e^{-\lambda z}. \tag{10}$$

Now, in the spirit on Monte-Carlo simulations, $F(z)$ will be a uniform random number r for $0 \leq r \leq 1$. Solving for z, we get

$$z = \frac{-1}{\lambda} \ln(1 - r). \tag{11}$$

This equation will tell us this: if we choose a uniform random number r on $0 \leq r \leq 1$, z will be the depth from the gamma-ray's current position it will interact next, given a material's linear attenuation coefficient of λ.

4.2.1 What is λ?

Equation 11 above is straightforward enough. It we choose a uniform r on $0 \leq r \leq 1$, we can find z, the distance the gamma-ray will travel in the detector before it interacts. But what do we put in for λ? You found λ for lead and aluminum in a previous lab, but what about for the NaI detector?

It turns out that the probability of interaction formulas above, namely P_{PA} and P_{CS} in Eqns. 1 and 2 are related to how far a gamma-ray will travel before interacting. Each formula is actually just an approximation for the mass-attenuation coefficients at energies near 0.663 MeV. They were taken from Knoll[5] and have units of cm^2/g. For the probability calculations, units and anything absolute isn't of concern because we always used them in ratios.

To get λ, the relationship is

$$\lambda(E) = \rho_{NaI}(P_{PA}(E) + P_{CS}(E)), \tag{12}$$

where ρ_{NaI} is the density of NaI or 3.67 g/cm^3.

Note that λ in Eqn. 6 would include any process that would remove a gamma-ray from an initial beam. In the case of a 0.663 keV gamma-ray, this would be PA and CS, so it seems logical that λ in Eqn. 12 would include the sum of both processes occurring, scaled by the density of material through which the gamma-ray is traveling.

This also means that $\lambda(E)$ has units of cm^{-1} meaning the probability of removing a gamma-ray from a beam, per cm of travel. So, Eqn. 11 should really be recast as

$$z = \frac{-1}{\lambda(E)} \ln(1 - r), \tag{13}$$

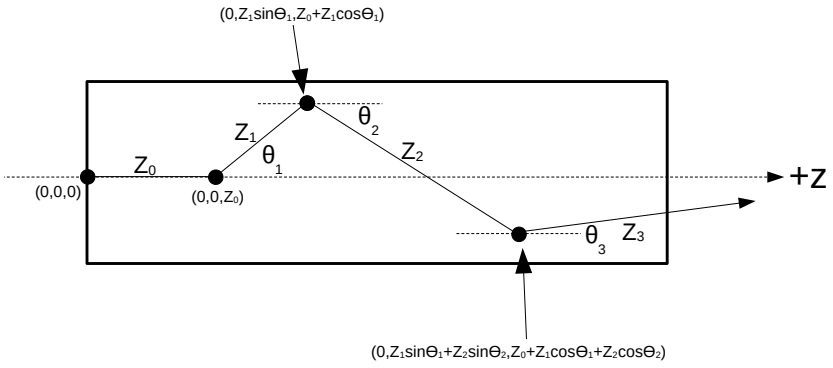
to emphasize the origin of $\lambda(E)$ from Eqn. 12.

[5]G. Knoll, " Radiation Detection and Measurement," 2nd ed., p. 51

4.3 Gamma-ray's position

Equation 13 is key to working a detector size into our simulation. It allows us to compute where the next gamma-ray interaction is likely to occur.

Here's an example of tracking a gamma-ray's multiple CSs through a detector, where we'll assume that the gamma-ray stays within the same $x = 0$ plane through its interactions.



Note the gamma-ray starts at $(0, 0, 0)$. We predict the first interaction depth z_0 from a random number and Eqn. 13. This will bring the gamma-ray to the position $(0, 0, z_0)$. If a CS is deemed to occur, we find the angle at which it scatters using the technique from Sect. 3.1.2 above. We'll now find another penetration depth z_1 from Eqn. 13. The CS angle will be θ_1, which will bring the gamma-ray to the position $(0, z_1 \sin\theta_1, z_0 + z_1 \cos\theta_1)$. If another CS occurs at angle θ_2, we compute still another penetration depth z_2, which will bring the gamma-ray to position $(0, z_1 \sin\theta_1 + z_2 \sin\theta_2, z_0 + z_1 \cos\theta_1 + z_2 \cos\theta_2)$.

This process continues until the gamma-ray finds itself outside of the bounds of the detector or PAs. Each of the CSs deposits energy into the detector in a running sum, which is logged in the event of an eventual PAs. If the gamma-ray leaves, the step-by-step CS electrons deposit energy into the detector (and contribute to the signal), with the leaving gamma-ray taking the un CSed energy with it. Note: when a gamma-ray "escapes" the detector, it can generate an "escape peak" in the spectrum. Escape peaks are indicative of recurring gamma-rays escaping the detector and not depositing all of there initial energy into it.

We find it amusing that the CS angle also takes on a geometrical use here, in advancing the gamma-ray's position through the detector.

4.4 Updated simulation steps

Here are the steps in your simulation that will now take into account a finite detector size. This section is an update version of Sec. 4.4. You should mostly be able to use your existing code from the small-detector simulation.

The major difference here is in how you keep track of the energy a gamma-ray ultimately contributes to your histogram. In the small detector case, it was either 0.663 for a direct PA, or the electron KE from a single Compton scatter.

In this larger-detector regime, you'll have to keep track of a running total of energies a single gamma-ray deposits as it traverses through your detector. Each CS event will add to this running total. That is, each CS does not contribute its individual energy to the signal; *the sum* of them all do.

This running total is only added to your histogram when the gamma-ray finally PAs or leaves the detector.

4.4.1 Steps

0. Initializers.
   (a) To begin, make a new gamma-ray with energy $E_0 = 0.663$.
   (b) Place it at position $(0, 0, 0)$.
   (c) Start a new variable called $E_{tot} = 0$, which is the total or "running" amount of energy the gamma-ray will deposit into the detector
   (d) You'll also need a variable to keep track of the current gamma-ray's energy. Call this variable E_g and set $E_g = E_0$ to start. E_g will change as a gamma-ray Compton scatters.
   (e) Decide how many iterations you want to run, N.
1. Assume the gamma-ray has an energy E_g.
2. Compute how far the gamma-ray will travel before an interaction using Eqn. 13. Base this on energy E_g.
3. Update the gamma-ray's position in the detector.
4. If the gamma-ray has left the detector, log the running energy E_{tot}, and start over with a new gamma-ray (go to Step 0).

5. Assume the gamma-ray has now encountered an electron. Determine if it will CS or PA based on energy E_g.
6. If PA, add the current gamma-ray energy, E_g to E_{tot}, then log E_{tot} and start over with a new gamma-ray (go to Step 0).
7. If CS, compute the scattered gamma-ray energy E' based on E_g. Also compute the KE of the Compton scattered electron, which is $E_g - E'$. Add this electron KE to your E_{tot}.
8. Update the gamma-ray's energy by setting $E_g = E'$.
9. Increment c. If $c \leq N$ loop back to Step 1.

4.5 Histogram to see the results

Once again, make a histogram of the E_{tot} values that you logged. Hopefully, you'll see a strong photopeak feature at 0.663 MeV as usual. There will also be a broad lower energy structure of CS events. But now, you should see that the possibility of multiple CS events will fill in the "Compton valley," which was left totally empty in the small detector regime.

Also as before, add some realism to your energy-array, by dithering the energies a bit as described above.

4.6 Optional: Plotting gamma-ray trajectories

As your code runs, you could compile (x, y, z) coordinates for a given gamma-ray's life in the detector. Making a plot of these (x, y, z) points and connecting them with lines would allow you to visualize the gamma-ray's life as it entered, interacted with, then left the detector. Perhaps other lines could be drawn that would show the bounds of the detector itself. Plotting the trajectories of 1000s of gamma-ray paths on the same plot might make an interesting graphic!

5 Part III: The very large detector regime

To this point, you've simulated a small and larger detector. What if the detector was very, very large (in reality: expensive, in theory: infinitely large), so that any incoming gamma-ray could never escape? Can you think what your spectrum might look like? Think again: a gamma-ray can never escape. What would your spectrum look like?

Adapting your simulation into the very-large regime, can be done pretty easily in two ways:

- Find the logic in your larger-detector code that checks if a gamma-ray has escaped. See Step 4 in Sect. 4.4.1. Tweak your code somehow so the logic of "it escaped" always fails, and re-run your code.
- You must have variables that set your detector's size. Make these values ridiculously large.

5.1 Histogram to see the results

Make a histogram of the E_{tot} values that you logged. Use this to describe a detector that is so large, a gamma-ray can never escape.[6]

6 Monte Carlo Integration

Again quoting Koonin, "it really is by no means obvious that anything sensible could come out of random numbers on a computer." Well, you've just used them to compute π and reproduce a gamma-ray spectrum. Let's close our study of Monte-Carlo techniques and use random numbers for one more thing: finding the answer to an integral.

Recall finding π by counting the number of "darts" that hit inside of a circle vs. inside of a surrounding square. Doing integrals with random numbers follows the same logic. Suppose in Fig. 4 the function you wish to integrate, $f(x)$ is shown by the black curve.

The integral of $f(x)$ from the lower to upper bound for which it is plotted in Fig. 4, is estimated as the area A multiplied by the fraction of random points that fall below the curve $f(x)$.

Come up with some function you'd like to integrate, and do so over some range of your choosing using this Monte Carlo method. Verify your Monte Carlo answer (somehow).

[6] see Fig. 10-3, p295 in Knoll, "Radiation Detection and Measurement" 2nd ed.

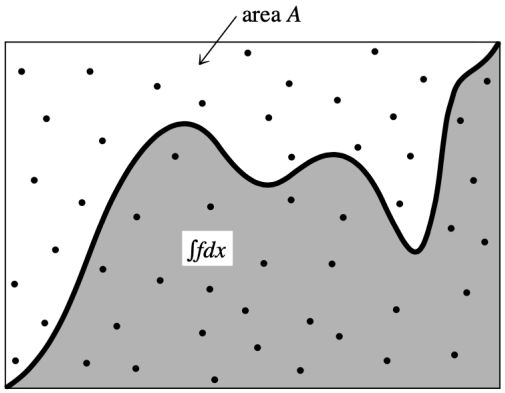


Figure 4: The epitome of using random numbers to do an integral. From Press, *Numerical Recipes in C*, Sect. 7.6, p. 306.