Monte Carlo Integration

Saad Mehmood

August 26, 2024

---

1 Introduction

Monte Carlo integration is a numerical integration method that uses probability and a large enough sample size to compute the area under a curve. This method belongs to a broader class of methods known as the "*Monte Carlo Methods*" that all rely on probability and large sample sizes to compute whatever we want.

2 Problem setup

Suppose we have some curve that follows the function $f(x)$ and we wish to find the area under this curve.

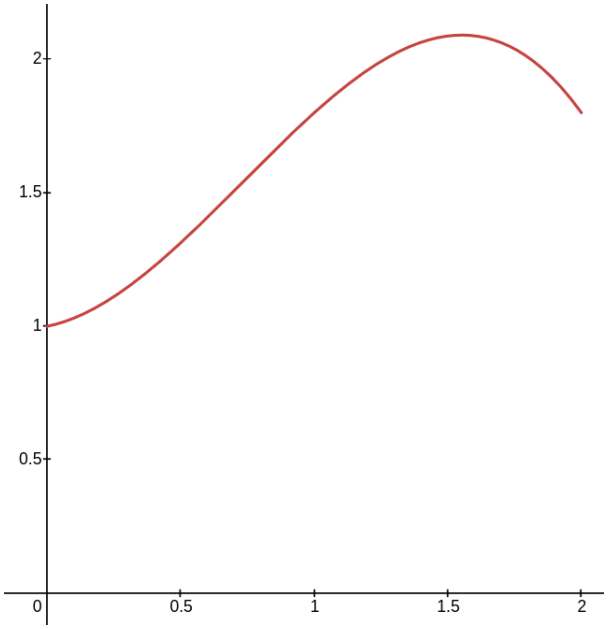

Figure 1: Some arbitrary curve given by the function $f(x)$

Normally we would just integrate the function over our required domain, Eq. (1), but in many cases integrating the function is not possible. For these situations we numerically solve the problem by various different methods. One of which is the Monte Carlo method

$$I = \int_a^b f(x)dx \tag{1}$$

3 Monte Carlo Integration

We begin by enclosing our curve in rectangle. Keeping in mind as the method relies on generating large amount of data points the rectangle has to be as close to the curve as possible.

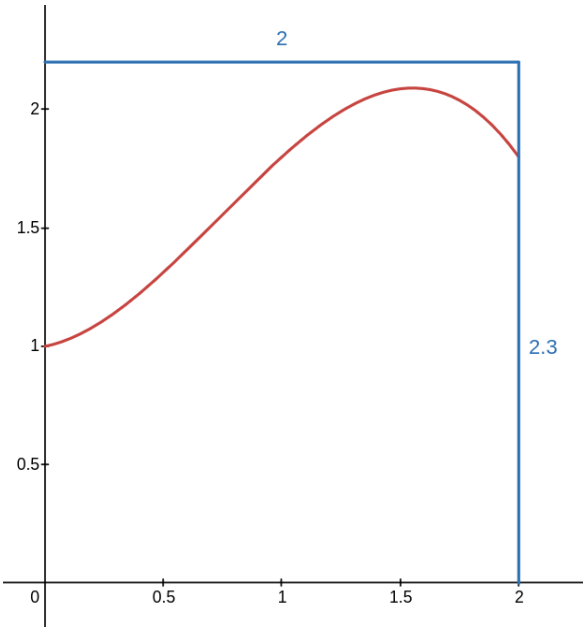

Figure 2: Enclosing the curve in a rectangle of dimensions (2×2.3)

The ratio between the area of the rectangle, (A), and the area under the curve, (I), would be constant if we do not change the dimensions of the rectangle and the domain of the curve.

$$\frac{A}{I} = C \tag{2}$$

This constant, C, is also linked to another ratio of numbers that is, the total number of points generated, N, and the number of points generated under the curve, n. For the sake of visualisation let us generate a few points in our rectangle.

Considering this fact we can now denote the constant C as;

$$\frac{N}{n} = C \tag{3}$$

Looking at Eq. (2) and Eq. (3) we can see that the two ratios are the same and hence we get the following expression;

$$\frac{A}{I} = \frac{N}{n} \tag{4}$$

Solving Eq. (4) for I we get an expression for the area under the curve in the following form,

$$I = \frac{An}{N} \tag{5}$$

And since we know that the area of a rectangle is just length times it's height the equation simplifies into,

$$I = \frac{(l \times h)n}{N} \tag{6}$$

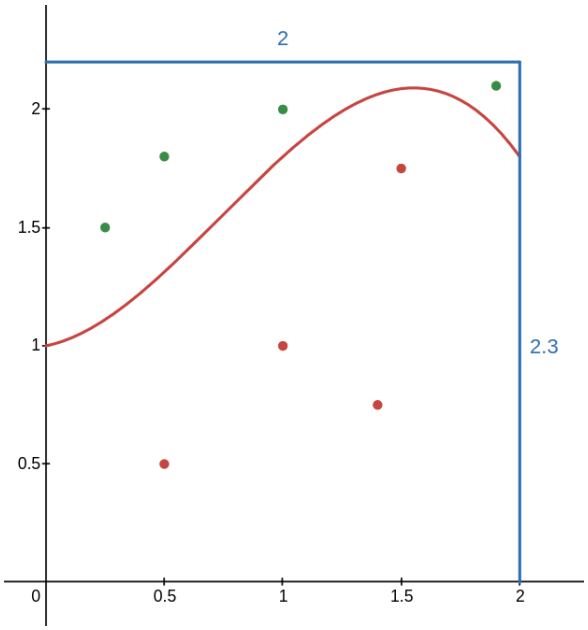

Figure 3: A set of points generated inside the rectangle

Finally from Eq. (6) to find the area under the curve we only need to count the number of points that are under the curve. But it can't be that easy, right?. Since this is a numerical method this is just an approximation of the actual area. For the case shown in Fig. (3) the area under the curve we get from Eq. (6) will be no where close to the actual area as the sample size is too small for the approximation to be good enough. Hence we have to generate many more points and consequently count many more points under the curve to get the answer, a task that is neither possible nor practical for a human to achieve meaning full results. But for a computer it is easy.

4 Algorithm

Since we now know how the method works and we also have a mathematical model. We will now come up with an algorithm that will help us calculate the area.

The steps that we must follow are as following;

1) Identify the function of the curve, $f(x)$

2) Identify the dimensions of the rectangle.

3) Generate a set number of points, N, within the rectangle. Each point will have an X_n and a Y_n value.

4) if the y value of the of the generated point, Y_n, is smaller than the value of the function at that x point, $f(X_n)$. Add 1 to the number of points under the curve.

5) Using the dimensions of the rectangle and the ratio of $\frac{n}{N}$ to find the area under the curve.

5 Example problem

Lets us now test our algorithm on a test problem. Suppose we have some curve that follows the function $f(x)$;

$$f(x) = e^{-2x} \tag{7}$$

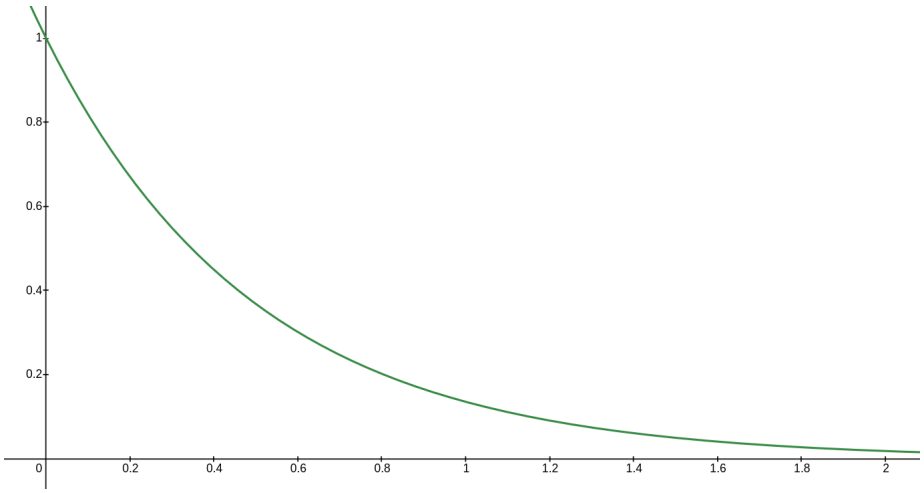

Figure 4: Plot of the function e^{-2x}

Let us find the area under this curve for domain $[0, 2]$. The maximum value of the curve in this domain is easy to figure out and is $y = 1$. Hence we will construct a rectangle of dimensions (2×1), which would look something like this,

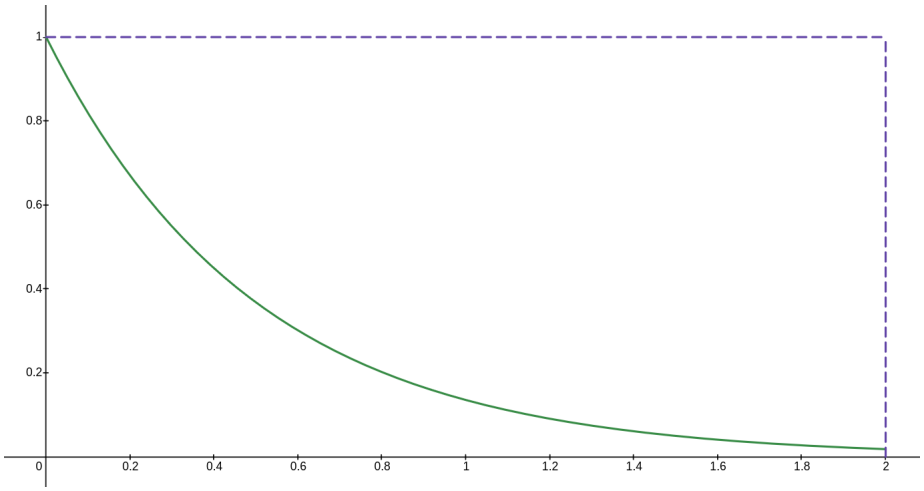

Figure 5: The curve enclosed by the a rectangle with dimensions (2×1)

The area of the rectangle therefore would be 2. For simplicity let us generate a mere 100 points and see what ratio $(\frac{n}{N})$ do we get. After generation the answer we get for the area under the curve is,

$$I = 0.46 \tag{8}$$

If we were to go again we would get a different answer as the point generation is random. But we increase the number of points significantly we would approach a fairly consistent answer. As a matter of fact, Table. (1) shows how for each run the area calculated changes, but when we increase the number of points the calculated area stays close to the actual area for each run.

No. of Points	Actual Area	run 1	run 2	run 3	run 4	run 5	Avg. Area
100	0.49084	0.54	0.58	0.4	0.5	0.44	0.492
1000	0.49084	0.48	0.52	0.5	0.442	0.484	0.4852
10000	0.49084	0.4922	0.4746	0.477	0.4956	0.4866	0.4852
100000	0.49084	0.49662	0.49418	0.48638	0.4885	0.4909	0.49136
1000000	0.49084	0.49049	0.490982	0.492294	0.489042	0.490516	0.49066

Table 1: Different results for different total number of points using monte Carlo Integration.

6 Computational problems

Due to the large number of points required for this method, it is not very efficient to use it reliably. Furthermore languages such as `Python` struggle to use this method efficiently. Hence to use this method it is best to use low level languages such as `C++, C, Fortran, F#`. The algorithm for remains the same for each language but the methods used vary on language used as not all languages have libraries that help with the random point generation.