The Curse of Monte Carlo

They attributed this hasty wedding to the Prince's dread of seeing accomplished an ancient prophecy, which was said to have pronounced that the castle and lordship of Otranto "should pass from the present family, whenever the real owner should be grown too large to inhabit it."

— Horace Walpole, *The Castle of Otranto*

The following is a series of guided exercises about Monte Carlo integration. Its purpose is to illustrate one of the uses of probabilistic thinking in overcoming computational obstacles.

Numerical Integration

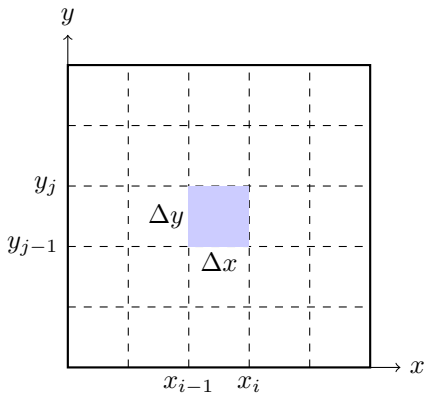
The scientific method demands that any scientific endeavor conclude with concrete results. In practice this means that your results must usually be numerical. An architect or engineer would need to provide measurements for her construction crew, and a scientist must provide predictions that can be put to the test. The prevalence of integrals in science and the fact that some integrals don't have analytic solutions naturally means that a suitable method for the numerical evaluation of integrals had to be developed.

In elementary courses of calculus, integration is usually defined as finding the area under the graph of a bounded function $f : [0, 1] \to \mathbb{R}$. This is usually done by cutting up the domain into n pieces at equidistant points $x_0, \ldots, x_n$, and then summing up the areas of the rectangles with base of $\Delta x = x_i - x_{i-1}$, and a height of $f(x_i)$. To be specific,

$$\int_0^1 f(x)\,dx = \lim_{n\to\infty} \sum_{i=1}^{n} f(x_i)\Delta x$$

where $\Delta x = 1/n$. This suggests that we can use the partial sums $\sum_{i=1}^{n} f(x_i)\Delta x$ as numerical approximations to the true integral. This actually works pretty well for integration in one variable, with some minor modification (take numerical analysis for further details).

What about higher dimensions? In two dimensions, we typically want to integrate our function $f : [0, 1]^2 \to \mathbb{R}$ over a rectangle $[0, 1]^2$. Typically, we partition each side of this rectangle into n pieces as we did above, resulting in the partition points $x_0, \ldots, x_n$ on the x-axis, and $y_0, \ldots, y_n$ on the y-axis. Consequently, the distance between neighboring points on either axis is going to be $\Delta x = \Delta y = 1/n$.



We approximate the volume of the region under the graph of f as the sum of the volumes of rectangular prisms whose base has area $\Delta x \Delta y$ and whose height is $f(x_i.y_i)$. That is,

$$\int_{[0,1]^2} f(x)\,dx\,dy = \lim_{n\to\infty} \sum_{i=1}^{n}\sum_{j=1}^{n} f(x_i, y_j)\Delta x \Delta y.$$

We can estimate this value by a partial sum $\sum_{i=1}^{n}\sum_{j=1}^{n} f(x_i, y_j)\Delta x\Delta y$.

1. *Suppose we want to estimate the integral using the partial sum above. At how many points do we need to evaluate f? In other words, how many sample points (x_i, y_j) are there?*

Similarly, to perform numerical integration in d-dimensions, we must partition each of the d-sides of the hypercube $[0,1]^d$ into n intervals. That is, we cut up the x_i-axis along equidistant points $x_{i0}, \ldots, x_{in}$. We approximate the volume under the graph of $f : [0,1]^d \to \mathbb{R}$ by summing up the volumes of the hyperrectangles with base $\Delta x_1 \cdots \Delta x_d$ and height $f(x_{1i_1}, \ldots, x_{di_d})$. That is,

$$\begin{aligned}&\int_{[0,1]^d} f(x_1, \ldots, x_d)\,dx_1 \cdots dx_d \\ &\quad = \lim_{n\to\infty} \sum_{i_1,\ldots,i_n=1}^{n} f(x_{1i_1}, \ldots, x_{di_d})\Delta x_1 \cdots \Delta x_d\end{aligned}$$

Therefore, the process of numerical integration along this idea will look something like

1. Divide each axis into n equidistant points: $\{x_{ij} : i = 1, \ldots, d, j = 0, \ldots, n\}$.
2. Find all sample points $\mathcal{S} = \{(x_{1i_1}, \ldots, x_{di_d}) : 1 \le i_1, \ldots, i_d \le n\}$ for suitably large n.
3. Evaluate the sum $\sum_{\mathbf{x}\in\mathcal{S}} f(\mathbf{x})\Delta x_1 \cdots \Delta x_d$.

This strategy has several flaws. Improving upon it is the goal of this article.

2. *For a fixed n, at how many points do we need to evaluate f? That is, how many sample points do we have? How does the number of sample points change when we increase the dimension?*

3. *In your favorite programming language, write a program that numerically integrates the* 10*-dimensional* $\mathrm{Beta}(\alpha, \beta)$ *density. That is,*

$$f(x_1, \ldots, x_d) = \prod_{i=1}^{d} \frac{x_i^{\alpha_i - 1}\,(1 - x_i)^{\beta_i - 1}}{B(\alpha_i, \beta_i)},$$

over the rectangle $[0,1]^d$ *where* $d = 10$*,* $B(\alpha, \beta)$ *is the Beta function. Pick some values for* $\alpha_1, \ldots, \alpha_d$ *and* $\beta_1, \ldots, \beta_d$ *at random and use a library to evaluate the Beta function. Since this is a probability density function, it should always evaluate to one. See how close you can get it to one by increasing the number of partition points on each axis (i.e., increasing* n*). Do you run into any difficulties?*

Monte Carlo Integration

From your solutions to the problems of the previous section, you may have noticed that the number of sample points increases *exponentially* as we increase the dimension. In fact, in my implementation of this "naive" algorithm, I was not able to integrate the 10-dimensional Beta density because I could not store all of the sample points! This is an example of *the curse of dimensionality*. The remainder of this article explains an approach for beating the curse of dimensionality (in this particular case) using probability.

Recall that if $\mathbf{x}$ is a uniform random vector on $\Omega = [0,1]^d$, then the expectation $\mathbb{E}f(\mathbf{x})$ is equal to $\int_\Omega f(\mathbf{x})\, d\mathbf{x}$. This suggests that if we can estimate the mean of the random vector $f(\mathbf{x})$, then we can estimate our desired integral. The obvious thing to do is to take several samples and to compute the sample mean. That is, generate i.i.d. uniform random vectors $\mathbf{x}_1, \mathbf{x}_2, \ldots$ on Ω, so that we can use the sample mean $\frac{1}{n}\sum_{i=1}^n f(\mathbf{x}_i)$ as an estimator for $\mathbb{E}f(\mathbf{x})$. This approach is called *Monte Carlo integration.* The law of large numbers guarantees that as we increase n our estimate will approach the mean as well. But what makes this better than the previous method?

4. *Show that* $\mathbb{E}[\frac{1}{n}\sum_{i=1}^n f(\mathbf{x}_i)] = \int_\Omega f(\mathbf{x})\, d\mathbf{x}$*. In statistics terminology, this means that our estimator is unbiased.*

Last time we needed a very large number of sample points to get a good estimate. If we wish to compare this new method with the last one, we need to find the number of sample points we need to get good "error". Typically error is measured using squared difference, which in this case would be $(\frac{1}{n}\sum_{i=1}^n f(\mathbf{x}_i) - \mathbb{E}f(\mathbf{x}))^2$. Since our method has some randomness involved, the error that we can *expect* to see is given by the expectation of this loss. This quantity is typically called the *risk* or L^2-error

$$R = \mathbb{E}\left[\left(\frac{1}{n}\sum_{i=1}^n f(\mathbf{x}_i) - \mathbb{E}f(\mathbf{x})\right)^2\right].$$

You will now investigate how large n has to be for R to be sufficiently low.

5. *Suppose that* f *is bounded by* M*, i.e., that* $|f(\mathbf{x})| \leq M$ *for all* $\mathbf{x} \in \Omega$*. Show that* $\operatorname{Var} f(\mathbf{x})$ *is bounded by* M^2*.*

6. *Show that* $R = \operatorname{Var}\left(\frac{1}{n}\sum_{i=1}^n f(\mathbf{x}_i)\right) \leq \frac{M^2}{n}$*. This shows that, as* $n \to \infty$*, the error goes to zero at about the same speed that* $1/n$ *goes to zero.*

The last exercise is very important. Notice that the dimension d plays no part in it. This means that no matter what the dimension is, the new *Monte Carlo* method will converge to the true answer in about the same time. That is, we have managed to beat the curse of dimensionality in the case of numerical integration.

7. *Redo exercise 3 but use the Monte Carlo method instead. Compare their performance, and how easy each one was to implement. (Hint: your Monte Carlo implementation shouldn't be longer than 3 lines).*

8. *Use your program from the previous exercise to estimate the volume of the d-dimensional unit ball for* $d = 1, \ldots, 100$.

9 (Improper integrals)**.** *The methods of this section were focused on integration over bounded sets, exemplified by* $[0,1]^d$. *Propose and implement a Monte Carlo strategy for integrating functions on all of* $\mathbb{R}^d$.