# Monte Carlo Simulations

Tristan Hodgson

Thursday, 16 February 2023

## 1 What is the Monte Carlo Method

If I tossed a coin five times, and it came up heads on all five times, you might assume that I am lucky since the probability is only about 3%. However, if I tossed a coin 100 times and got 100 heads, now you would likely accuse me of cheating since the chance of that happening is about $8 \times 10^{-31}$. The reason for this is that you are, intuitively, seeing the result of something called the Law of Large Numbers, which means that as the number of coin tosses increases, the proportion of heads should tend towards the probability of getting a head (i.e. 0.5). It is this law that a rather interesting piece of statistical analysis uses to allow us to vastly simplify a large number of calculations across a range of fields.

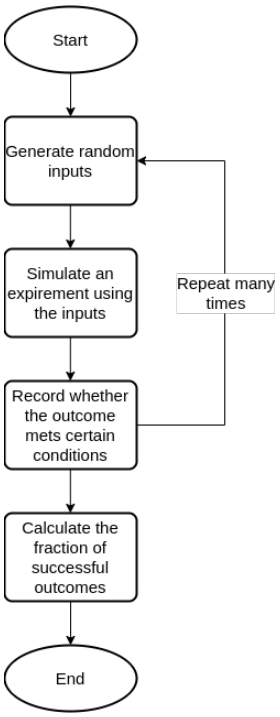


Figure 1: A flowchart of the basic process that we follow when performing a Monte Carlo simulation.

A Monte Carlo simulation is a repetitive process of random generation of inputs, simulation and evaluation, as shown in the flowchart in figure 1. It is, however, perhaps best understood through an example. Figure 2 shows a graphical representation of a Monte Carlo simulation to determine the value of $\pi$. A computer randomly generates numbers between -1 and 1 for both the x and y coordinates, then plots these points and colours them blue if it is within the circle and red if now. It determines whether a point is inside or outside of the equation by seeing whether the logical expression shown in equation 1 is true or false.

$$x^2 + y^2 \leq 1 \tag{1}$$

I like this example because it shows the relative simplicity of Monte Carlo simulations and how such simulations are always an approximation. For example, when we used 30,000 samples (or, in other words, repeated the experiment 30,000 times), as shown on the graph, we achieved a value of $\pi = 3.1376$, at a percentage error of just 0.127%, increasing the number of samples would decrease this error but increase the amount of computation required[1]. While this is good, it is certainly far from the best method of calculating $\pi$ as computers are capable

[1]Please note that this value is dependant on the specific run, due to the use of random numbers, replicating this result with the code below will likely yield slightly different results.

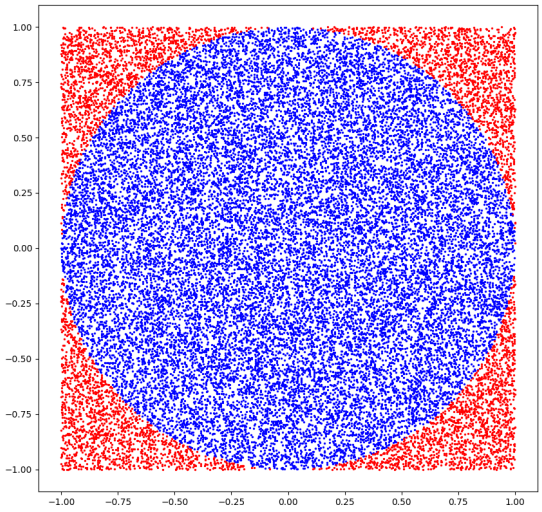


Figure 2: A graph of the random points generated by the algorithm to find pi; the red points are outside of the circle, the blue points are on or within the circle.

of far higher degrees of accuracy using other methods[2]. As such, Monte Carlo simulations are seldom used to calculate $\pi$, outside of demonstrations of the theory as we have done, and instead are used to find approximate solutions where finding exact solutions would be impractical.

## 2 History and Early Uses

Scholars debate the earliest use of the Monte Carlo method; one notable early use is approximating the value of the definite integrals involved in the nuclear chain reaction of the Atomic Bombs on the Manhattan project. These calculations are similar to the example above where we calculated $\pi$ since both use an inequality to determine whether a specific coordinate is inside or outside the function and then calculate the proportion of points inside compared to outside (see equation 2).

$$y \leq f(x) \tag{2}$$

Using this method allowed the scientists on the Manhattan project to approximate the probability that a neutron produced from a uranium atom undergoing fission would collide with another uranium atom to cause a chain reaction of nuclear fission, leading to a nuclear explosion. By simplifying these calculations, using the Monte Carlo method, scientists could calculate the probability of a chain reaction occurring.

Its use in the Manhattan Project is interesting because the scientists did not know the value (unlike when we used it to find the value of $\pi$). As a result, they had to trust that they had done their Monte Carlo simulation correctly and that it would give them a valid approximation.

## 3 Modern Uses

The most prevalent modern use of the Monte Carlo method centres around uncertainty calculations. When taking any measurement, there is an associated uncertainty; for example, if you read a temperature using a thermometer, you cannot measure the difference between 21.4°C and 21.5°C (unless you have a very expensive thermometer); hence, if you read off 21.5°C then we don't know if the temperature is actually 21.4°C. Thus, we have an uncertainty in our value.

Traditionally, if you are calculating with uncertainty, you can do what's known as a worst-on-worst calculation; by picking the worst possible value within the uncertainty range for each input and using these values for the simulation. However, with the Monte Carlo method, what we can do instead is randomise the inputs within their uncertainty range, run our simulation and then repeat.

[2]Google found 100 trillion digits of $\pi$ and even inputting just the first term of the equation they used gave me $\pi$ correct to 10 decimal places, the maximum number my calculator will show, in under a second

The advantage of the Monte Carlo method over the worst-on-worst effect method is that the Monte Carlo method is better at dealing with situations that have a low margin of error since continually taking the worst possible value will result in a scenario with an exceptionally low probability, where a Monte Carlo simulation will show all likely cases.

It is perhaps easier to see quite what this means with an example. In weather prediction, we use a technique called ensemble forecasting, which is a form of Monte Carlo simulation. Let's return to our thermometer; let's say that this thermometer is accurate to the nearest degree hence all values for 20.5°C to 21.5°C will be read as 21°C. We might have a couple hundred of these thermometers over a country feeding back for us to predict the temperature in the future. Since all the thermometers will have the same uncertainty, this will quickly add up, making our final prediction little more than an educated guess. So what we do is we do a Monte Carlo simulation and randomise all of the temperature values between the uncertainty range (in our case, between 20.5°C and 21.5°C) and run our simulations using these results, repeat this many times, and take an average of our results.

While predicted weather is far more complicated than this, the principle of ensemble forecasting is one that weather agencies across the world use and is arguably the main reason why the Met Office can claim to be able to predict tomorrow's temperature to the nearest 2°C 92% of the time.

Similarly, NASA uses Monte Carlo simulations for their trajectory data due to the low margin of error in space flight. While NASA could spend billions of dollars and tens of thousands of hours of engineering time reducing the uncertainty on their measuring equipment so they could use the worst-on-worst method, they are much better off simply using a Monte Carlo simulation to see whether the trajectories are likely to work.

# 4 Limitations

The computational intensity of this method is vast–the Met Office has a supercomputer called Cray XC40 which can do 14,000 trillion arithmetic operations per second. While the simulations involved in weather prediction are complicated, and this does account for a lot of this computing power, another major reason for the use of powerful computers like Cray is the need to perform every step of the calculation multiple times with slight variations in the inputs. The use of these vastly expensive supercomputers can limit the accessibility of these tools, at least in weather prediction (though as we showed earlier simple models can run on even the most basic of computers).

Moreover, as you are no doubt aware, weather prediction is not always correct. In weather prediction, this error caused by using an approximation is small (especially since using exact values is impossible since this would necessitate the removal of uncertainty from our measurements) since even in the more life-and-death field of storm prediction, the errors involved can be offset by simply being slightly overcautious and erring on the side of warning too often rather than too rarely. However, as we saw with our approximation of $\pi$, the error of our results can be high, particularly if we use a low number of samples. Therefore, we cannot implicitly trust the data that comes out of a Monte Carlo simulation: we should always try to verify the results through another method in conjunction with a Monte Carlo simulation when it is critical that the output is correct.

# 5 Conclusion

Monte Carlo simulations are undoubtedly a valuable method to scientists across all fields because of their capability to reduce uncertainty and simplify calculations. While there are limitations to its use, where it is applicable, it has affected the lives of billions and is a tool that is not talked about enough in the wider maths community.

# 6 Code

```
from random import randint as rand
import matplotlib.pyplot as plt
import numpy as np


def gen_values(n):
    summary = []
    for i in range(0, n):
        # Generate the x, y coords between -1 and 1
        x = rand(-1000, 1000)
        x = x/1000
        y = rand(-1000, 1000)
        y = y/1000

        # Is the point in the circle
        circle = x**2 + y**2
        if circle > 1:
            in_circle = False
        else:
            in_circle = True
        coord = [x, y, in_circle]
        summary.append(coord)
    return summary


def plot(values):
    x_true = []
    y_true = []

    x_false = []
    y_false = []

    for i in values:
        # Prepare the coords for plotting
        if i[2] == True:
            x_true.append(i[0])
            y_true.append(i[1])
        else:
            x_false.append(i[0])
            y_false.append(i[1])
    # Make the plot square
    f = plt.figure()
    f.set_figwidth(10)
    f.set_figheight(10)
    # Plot the results in the circle
    plt.scatter(x_true, y_true, color='blue', s=2)
    # Plot the results outside the circle
    plt.scatter(x_false, y_false, color='red', s=2)
    # Show the plot
    plt.show()


def calculate_pi(values):
    # Numerate the number of points in and out of the circle
    true_num = 0
    false_num = 0
    for i in values:
        if i[2] == True:
            true_num += 1
        else:
            false_num += 1
    # Calculate the value of pi
    pi = true_num / (true_num + false_num)
    pi = pi * 4
    return pi

# Normal mode, graph and calculate using 1 value for the number of points
data = gen_values(3*10**4)
print(calculate_pi(data))
plot(data)
```

# 7 Sources

**Applying Monte Carlo Simulation to Launch Vehicle Design and Requirements Analysis by J.M. Hanson and B.B. Beard p.g. 1-5** https://ntrs.nasa.gov/citations/20100038453 **p.g. 1-5** A massive document all about the use of Monte Carlo simulations at NASA, including detailed descriptions of what it is, how it is used and some basic history.

**Firmament–The Hidden Science of Weather, Climate Change, and the Air that Surrounds Us by Simon Clark** Contains, amongst a vast wealth of other things, an excellent description of how and why ensemble forecasting work along with the birth of the understanding of chaos theory in climate science.

**Monte Carlo Simulation by Science Direct** **https://www.sciencedirect.com/topics/economics-econometrics-and-finance/monte-carlo-simulation**
**https://www.sciencedirect.com/topics/neuroscience/monte-carlo-method** Two articles which summarise a lot of information about Monte Carlo simulations and how they are used in the real world.

**6. Monte Carlo Simulation by MIT OpenCourseWare**
**https://www.youtube.com/watch?v=OgO1gpXSUzU** Contains an excellent description of the distinction between the Law of Large Numbers and the Gamblers fallacy.

**A detailed proof of the Chudnovsky formula with means of basic complex analysis – Ein ausführlicher Beweis der Chudnovsky-Formel mit elementarer Funktionentheorie by Lorenz Milla**
**https://arxiv.org/abs/1809.00533** Contains the equation that Google used to calculate 100 trillion digits of $\pi$, only using 1 term got me at least 10 digits correct (my calculator doesn't show more that 10 digits).

**Monte Carlo method applied to approximating the value of $\pi$ by nicoguaro.**
**https://en.wikipedia.org/wiki/Monte_Carlo_method#/media/File:Pi_30K.gif** An amazing GIF of how the percentage error of the approximation for the value of $\pi$ changes as the number the number of points increases.

**Bayesian Statistics with Hannah Fry by Matt Parker** https://www.youtube.com/watch?v=7GgLSnQ48os The video that I used as a jumping off point for my research, contains a very good example that I didn't touch where Matt repeatedly throws balls at tables (and missing a lot).

**Even more pi in the sky: Calculating 100 trillion digits of pi on Google Cloud by Emma Haruka Iwao** **https://cloud.google.com/blog/products/compute/calculating-100-trillion-digits-of-pi-on-google-cloud** The annoucment post of how Google calculated $\pi$ to 100 trillion digits.