University of Clermont Auvergne
ISIMA

Report

Monte Carlo Simulation and Confidence Intervals

Student
Kirill Aksenov

Professor
David Hill

Clermont Ferrand, 2018

Contents

1 Introduction

The Monte Carlo simulation method is still very popular. With it, a huge number of different studies are conducted. In this paper, we will consider the method of calculating the number π using the Monte Carlo method; we will also provide the algorithm for finding the confidence interval for the value found. In the end, the Fibonacci sequence will be considered, as well as its relationship with the number of ϕ.

2 Monte Carlo π evaluation

2.1 Algorithm

In first problem it was necessary to estimate π value, using Monte-Carlo method. In order to do this, the following algorithm was used:

1. Generate 2 random numbers in [0, 1] using Mersenne Twister. They will be interpretated as a x and y coordinates.
2. Check that $x^2+y^2 \leq R^2$, where $R = 1$. Meaning of this, that our random coordinates lie inside a circle.
3. We know, that the area S enclosed by a circle of radius R is $S = \pi * R^2$. In our case $S = \pi$.
4. Area can be estimated as Area $= \dfrac{\text{Number of points in circle}}{\text{Number of points}}$.
5. By the way, we are generating random variables that can lie only in positive quarter, thats why we should multiply our area value by four to obtain π.

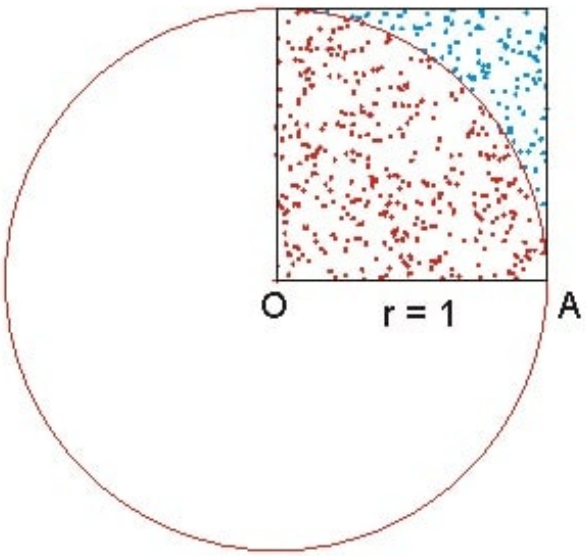

Figure 1: How Monte Carlo π estimation works.

2.2 Examples of Work

In table (1) it is possible to see dependence between number of points that used in Monte Carlo simulation and value of π.

Number of points	π value
10	2.8000000
100	3.2000000
1000	3.1240000
1000000	3.1447600
1000000000	3.1415455

Table 1: Examples of Work

It is easy to see and understand that increasing number of points in MC simulation leads to increasing quality if estimation. In next problem we will see, that it is also possible to increase quality by using N independent experiments.

Code

Listing 1: Example of Code that can evaluate MC.

```
double circle(double x, double radius) {
  return(pow(radius, 2) - pow(x, 2));
}

double find_pi(int64_t num){
  int64_t Num_In = 0;
  for(int64_t i = 0; i < num; i++){
    if (pow(genrand_real1(), 2) <
          circle(genrand_real1(), 1.0))
          Num_In++;
  }
  return( (Num_In / (double) num) * 4.0 );
}
```

3 N Independent experiments

In this problem it was necessary to propose a loop function, that compute N independent π evaluation experiments and that compute mean of all experiments. In table

(2) below possible to see how much does the value depend on number of points in each experiment and number of experiments.

Number of points	Number of Experiments	Mean π value
1000	1000	3.1447600
1000	100000	3.1412454
1000000	100	3.1412454
1000000	10000	3.1415627

Table 2: Dependency of quality and number of experiments.

It can be seen that an increasing number of experiments leads to an improvement in the quality of the assessment, just as it can be done by increasing the number of points in each experiment. If we look at second and third row in table (2) we can see, that they got same mean π value. It happens because our Mersenne Twister initialized once, when the program starts, so we got same values of each random value. Also there are same amount $1000 * 100000 = 10^8$ of points in both examples. That is why it does not depend on which parameter you need to increase (the number of experiments or the number of points in each).

4 Confidence Interval

4.1 Theory

In statistics, a confidence interval (CI) is a type of interval estimate, computed from the statistics of the observed data, that might contain the true value of an unknown population parameter. The interval has an associated confidence level that, loosely speaking, quantifies the level of confidence that the parameter lies in the interval. More strictly speaking, the confidence level represents the frequency (i.e. the proportion) of possible confidence intervals that contain the true value of the unknown population parameter. In other words, if confidence intervals are constructed using a given confidence level from an infinite number of independent sample statistics, the proportion of those intervals that contain the true value of the parameter will be equal to the confidence level.

Algorithm how to compute confidence interval for our task:

1. Compute sample mean:
$$\bar{X}(n) = \frac{1}{n}\sum_{i=1}^{n} X_i$$
2. Know we can compute sample variance:
$$S(n)^2 = \frac{n}{n-1}\hat{\sigma}^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \overline{x})^2.$$

3. The confidence radius at the $1-\alpha$ level is given by

$$R = -t_{1-\frac{\alpha}{2},n-1}\frac{S}{\sqrt{n}}.$$

4. Confidence interval will be equal to

$$\mathbb{P}\left(\bar{X} - t_{1-\frac{\alpha}{2},n-1}\frac{S}{\sqrt{n}} \leq \pi \leq \bar{X} + t_{1-\frac{\alpha}{2},n-1}\frac{S}{\sqrt{n}}\right) = 1-\alpha.$$

Code

Listing 2: Confidence interval evaluation.

```
pair<double, double> find_interval(int num, int num_of_att){
  vector<double> PIs;
  double t = 12.7062;
  for(int i = 0; i < num_of_att; i++){
    PIs.push_back(find_pi(num));
  }
  double mean = accumulate( PIs.begin(), PIs.end(), 0.0)
                    / PIs.size();
  double sq_sum = inner_product(PIs.begin(), PIs.end(),
                  PIs.begin(), 0.0);
  double stdev = sqrt(sq_sum / PIs.size() - pow(mean, 2));
  return( make_pair(mean - t*stdev/sqrt(num_of_att),
           mean + t*stdev/sqrt(num_of_att)) );
}
```

4.2 Examples of Work

Now, when we got working algorithm, we can compute confidence intervals for different cases. Lets look at the table (3):

Number of points	Number of Experiments	Confidence Interval
100	100	$2.95695 < PI < 3.33905$
1000	1000	$3.12446 < PI < 3.16506$
10000	1000	$3.13438 < PI < 3.14819$
100000	10000	$3.14088 < PI < 3.14221$
100000	100000	$3.14135 < PI < 3.14177$

Table 3: Dependency of quality and number of experiments.

As can be seen, increasing number of points in each experiment and number of experiments leads to constriction of confidence interval. Using 10^{10} points give utterly narrow CI, with can be used in scientific works. Main problems for this method of π estimation is slow convergence rate as well as high computation time.

5 Rabbit population simulation

Last task is about modeling Fibonacci sequence:

$$\begin{cases} X_i = X_{i-2} + X_{i-1}, \\ X_0 = 0, \\ X_1 = 1. \end{cases}$$

5.1 History

Fibonacci considers the growth of an idealized (biologically unrealistic) rabbit population, assuming that: a newly born pair of rabbits, one male, one female, are put in a field; rabbits are able to mate at the age of one month so that at the end of its second month a female can produce another pair of rabbits; rabbits never die and a mating pair always produces one new pair (one male, one female) every month from the second month on. The puzzle that Fibonacci posed was: how many pairs will there be in one year?

- At the end of the first month, they mate, but there is still only 1 pair.
- At the end of the second month the female produces a new pair, so now there are 2 pairs of rabbits in the field.
- At the end of the third month, the original female produces a second pair, making 3 pairs in all in the field.
- At the end of the fourth month, the original female has produced yet another new pair, and the female born two months ago also produces her first pair, making 5 pairs.

At the end of the n-th month, the number of pairs of rabbits is equal to the number of new pairs (that is, the number of pairs in month $n-2$) plus the number of pairs alive last month (that is, $n-1$). This is the n-th Fibonacci number.

5.2 Connection with the number of φ

In mathematics, two quantities are in the golden ratio if their ratio is the same as the ratio of their sum to the larger of the two quantities. Expressed algebraically, for quantities a and b with a $>$ b $>$ 0,

$$\frac{a+b}{a} = \frac{a}{b} \stackrel{\text{def}}{=} \varphi.$$

It is possible to see, that using Fibonacci sequence language:

$$\frac{F_{n+1}}{F_n} \to \varphi. \tag{1}$$

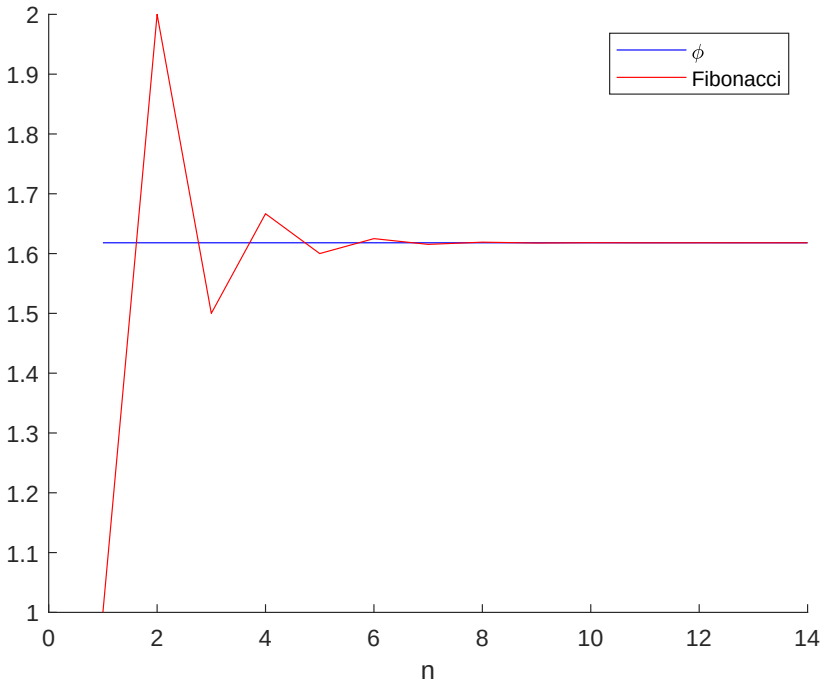

Figure 2: Convergence of (1)

It can be seen on example of golden spiral, which is a logarithmic spiral whose growth factor is φ, the golden ratio.

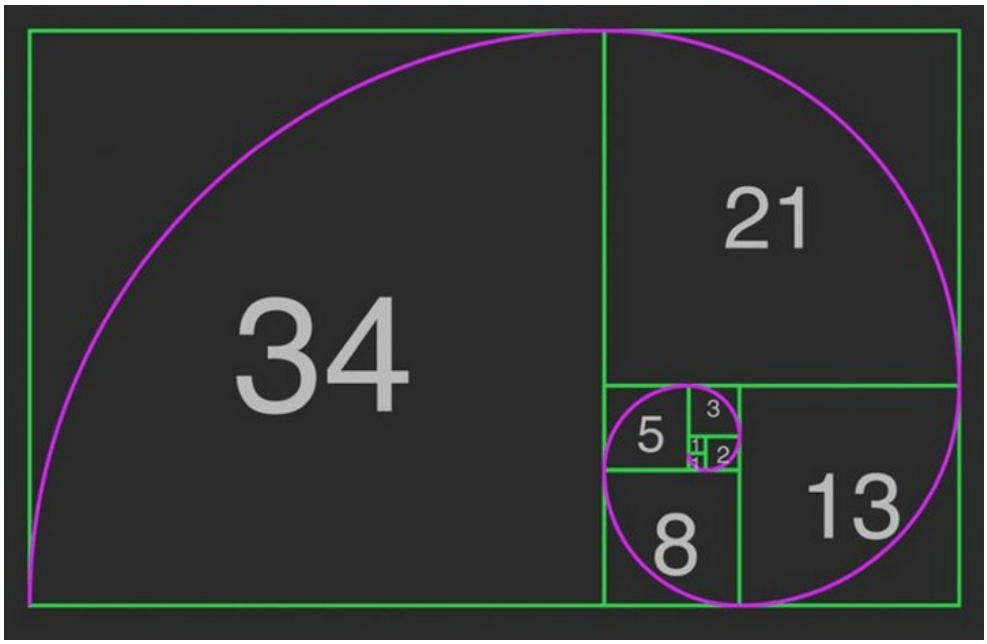

Figure 3: Approximation of the golden spiral

As can be seen, it is fits well Fibonacci numbers.

Code

Listing 3: Fibonacci sequence evaluation.

```
int Fibonacci(int n) {
  if (n == 0)
    return 0;
  if (n == 1 || n == 2)
    return (f[n] = 1);
  if (f[n])
    return f[n];
  int k = (n & 1)? (n+1)/2 : n/2;
  f[n] = (n & 1) ?
        (Fibonacci(k)*Fibonacci(k)
        + Fibonacci(k-1)*Fibonacci(k-1)) :
        (2*Fibonacci(k-1) + Fibonacci(k))*Fibonaccik);
  return( f[n]);
}
```

6 Conclusion

In this paper, we reviewed various techniques for working with the Monte Carlo method. I would like to note that this method has a huge advantage - it is its intuitive clarity. However, along with it come the disadvantages, such as:

- Very slow convergence (for the problem considered in this paper).
- High running time.

It is difficult to make an exact verdict. But one thing is certain: The Monte Carlo method is perfect for exploring stochastic modeling.