Projet VBA: Monte Carlo and Riemann Simulation Ln (2)

Chenjie LI

January 28, 2024

1 Context

In this project, we want to estimate the value of ln (2) by two different methods: Monte Carlo and Riemann.

Firstly, we are going to explain what Monte Carlo and Riemann Method is; then we will explain the code that we implemented on VBA; finally, we are going to compare these two methods by plotting their convergence in function of number of iteration or subdivision.

Monte-Carlo method: Unlike a normal forecasting model, Monte Carlo Simulation predicts a set of outcomes based on an estimated range of values versus a set of fixed input values. In other words, a Monte Carlo Simulation builds a model of possible results by leveraging a probability distribution, such as a uniform or normal distribution, for any variable that has inherent uncertainty. It, then, recalculates the results over and over, each time using a different set of random numbers between the minimum and maximum values. In a typical Monte Carlo experiment, this exercise can be repeated thousands of times to produce many likely outcomes.

Monte Carlo Simulations are also utilized for long-term predictions due to their accuracy. As the number of inputs increase, the number of forecasts also grows, allowing you to project outcomes farther out in time with more accuracy. When a Monte Carlo Simulation is complete, it yields a range of possible outcomes with the probability of each result occurring.

Riemann method: In mathematics, a Riemann sum is a certain kind of approximation of an integral by a finite sum. One very common application is in numerical integration, i.e., approximating the area of functions or lines on a graph, where it is also known as the rectangle rule. It can also be applied for approximating the length of curves and other approximations.

The sum is calculated by partitioning the region into shapes (rectangles, trapezoids, parabolas, or cubic) that together form a region that is like the region being measured, then calculating the area for each of these shapes, and finally adding all these small areas together. This approach can be used to find a numerical approximation for a definite integral even if the fundamental theorem of calculus does not make it easy to find a closed-form solution. Because the region by the small shapes is usually not the same shape as the region being measured, the Riemann sum will differ from the area being measured. This error can be reduced by dividing up the region more finely, using smaller and smaller shapes. As the shapes get smaller and smaller, the sum approaches the Riemann integral.

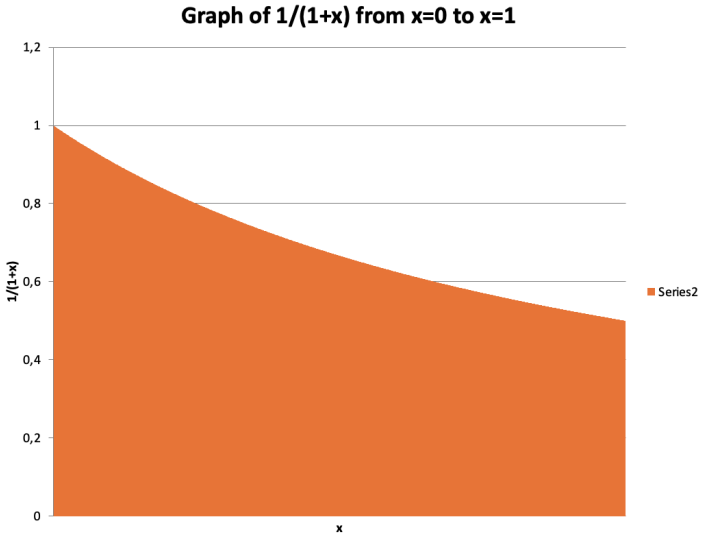
2 Implementation of Monte-Carlo Method

Different steps to implement the Monte Carlo integral: 1) Set up the predictive model, identifying both the dependent variable to be predicted and the independent variables (also known as the input, risk, or predictor variables) that will drive the prediction.

2) Specify probability distributions of the independent variables. Use historical data and/or the analyst's subjective judgment to define a range of likely values and assign probability weights for each.

3) Run simulations repeatedly, generating random values of the independent variables. Do this until enough results are gathered to make up a representative sample of the near infinite number of possible combinations.

$$\ln(2) - \ln(1) = \int_0^1 \frac{1}{x+1}\,dx$$




```
Function MonteCarloLn2Simulation(ByVal numIterations As Long) As Double
    Dim sum As Double
    Dim i As Long

    Randomize ' Initialize the random number generator

    For i = 1 To numIterations
        Dim x As Double
        x = Rnd() ' Generate a random number between 0 and 1

        ' Calculate the function value at x
        Dim f As Double
        f = 1 / (1 + x)

        sum = sum + f ' Accumulate the function values
    Next i

    ' Calculate the average of the function values
    Dim average As Double
    average = sum / numIterations

    ' Calculate the estimated value of ln(2)
    Dim ln2 As Double
    ln2 = average

    MonteCarloLn2Simulation = ln2
End Function
```

3 Implementation of Riemann Method

Different steps of Riemann Model:

1) Set up the Model: Formulate the mathematical model for the Riemann sum. This involves defining the function to be integrated, specifying the interval over which the integration will be performed, and determining the number of subdivisions or intervals.

2) Generate Random Samples: In the Riemann method, the random samples are not generated as in the traditional Monte Carlo method. Instead, the samples are obtained by dividing the interval into a specified number of subdivisions and evaluating the function at randomly selected points within each subdivision. These random points are used to calculate the function values for the Riemann sum.

3) Perform Calculations: For each randomly selected point within a subdivision, calculate the value of the function. Multiply this value by the width of the subdivision to obtain the contribution of that subdivision to the Riemann sum. Sum up the contributions from all the subdivisions to obtain the estimated value of the definite integral using the Riemann sum.

Let a and b be two real numbers with $a < b$. Let f be a piecewise continuous function on $[a; b]$ with values in R. For any $n \in N^*$, we call the Riemann sums of order n associated with f the sums:

$$s_n = \frac{b-a}{n} \sum_{k=1}^{n} f\left(a + k\frac{b-a}{n}\right)$$

$$= \frac{b-a}{n}\left[f\left(a + \frac{b-a}{n}\right) + f\left(a + 2\frac{b-a}{n}\right) + f\left(a + 3\frac{b-a}{n}\right) + \ldots + f\left(a + n\frac{b-a}{n}\right)\right]$$

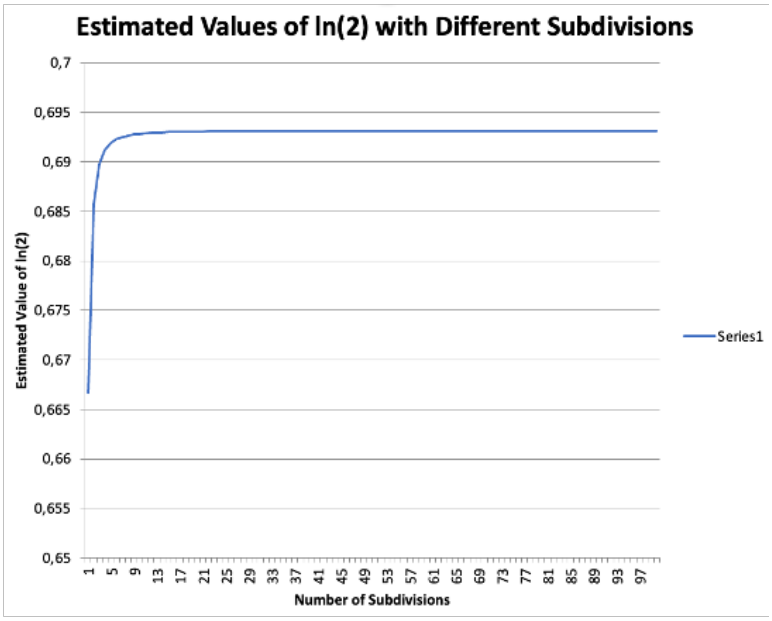
Theorem: Convergence of Riemann Sums The sequences $(r_n)_{n\in N^*}$ and $(s_n)_{n\in N^*}$ converge to the integral of f between a and b.

$$\lim_{n\to+\infty} s_n = \int_a^b f(t)\,dt$$

```
Function RiemannIntegralLn2(n As Integer) As Double
    Dim width As Double
    Dim sum As Double
    Dim x As Double
    Dim i As Integer

    ' Calculate the width of each subdivision
    width = (2 - 1) / n

    ' Initialize the sum
    sum = 0

    ' Evaluate the function at the sample points and calculate the sum
    For i = 1 To n
        ' Calculate the x-coordinate of the midpoint of the subdivision
        x = 1 + (i - 0.5) * width

        ' Evaluate 1/x at the sample point x
        sum = sum + width / (x)
    Next i

    ' Return the estimated value of ln(2) using the Riemann sum
    RiemannIntegralLn2 = sum
End Function
```

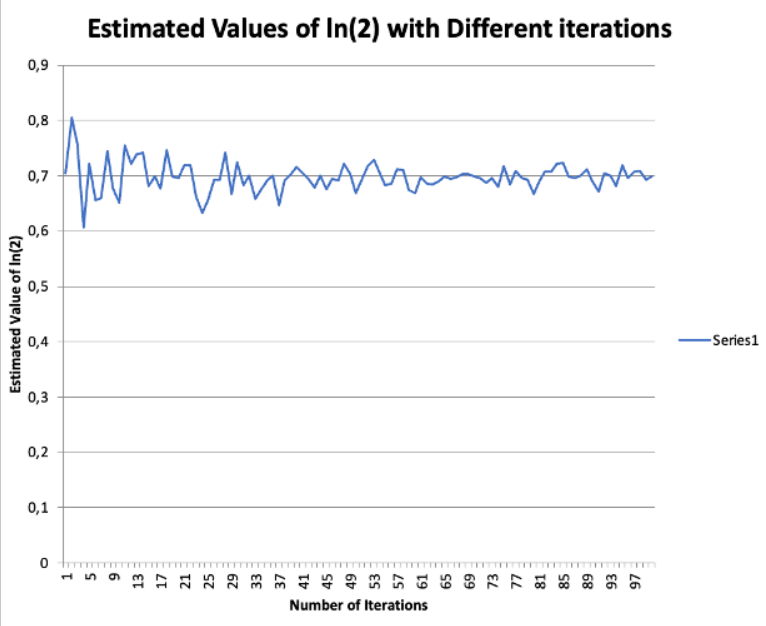
4 Interpretation of Riemann Method

As we can see in the graph that the more subdivisions there are, the more precis the result would be. And the Riemann method gives us an estimation of ln (2) which is close to the true value for more than 10 subdivisions.



5 Interpretation of Monte Carlo Method

We could conclude that the Monte Carlo method gives us an estimation more precise with more iterations but as it uses a probabilistic model, the results are still unstable even we have a great number of iterations.



6 Conclusion

-

Riemann Method:
Advantages:

1) Deterministic: The Riemann method provides a deterministic estimate of ln(2) by dividing the interval into subdivisions and evaluating the function at specific sample points.

2) Simplicity: The Riemann method is relatively straightforward to implement and understand, especially for simple functions like ln(x).

3) Accuracy with Sufficient Subdivisions: With a sufficiently large number of subdivisions, the Riemann method can provide accurate estimates of ln(2).

Inconveniences:

1) Limited Accuracy: The accuracy of the Riemann method heavily depends on the number of subdivisions used. As the number of subdivisions increases, the accuracy improves, but it can be computationally expensive for very high accuracy.

2) Limited Applicability: The Riemann method is primarily suitable for integrating functions, and estimating ln(2) is a specific use case. It may not be as versatile as the Monte Carlo method for estimating a wide range of values or solving complex problems.

3) Potential Bias: The Riemann method can introduce bias if the sample points within each subdivision are not representative of the overall function behaviour.

Monte Carlo Method:
Advantages:

1) Versatility: The Monte Carlo method is highly versatile and applicable to a wide range of problems beyond integration. It can be used to estimate ln(2) as well as other complex mathematical functions or simulate probabilistic scenarios.

2) Statistical Interpretation: The Monte Carlo method provides a statistical interpretation of the estimate, allowing for the quantification of uncertainty and confidence intervals.

3) Flexibility with Sampling: The Monte Carlo method allows for various sampling techniques, including random, stratified, or importance sampling, which can potentially improve accuracy.

Inconveniences:

1) Stochastic Nature: The Monte Carlo method relies on random sampling, which means the estimates can have inherent variability and may require many samples to obtain accurate results.

2) Computational Intensity: Monte Carlo simulations can be computationally intensive, especially when dealing with complex functions or systems, requiring significant computational resources and time.

3) Convergence Issues: Convergence to an accurate estimate can be slow, and the required number of samples can be difficult to determine in advance.

In summary, the Riemann method is simpler and deterministic, but its accuracy is highly dependent on the number of subdivisions used. On the other hand, the Monte Carlo method is versatile and provides statistical interpretations, but it requires many samples and can be computationally intensive. The choice between the two methods depends on the specific problem, accuracy requirements, and available computational resources.