Computing of the Area of the Mandelbrot Set

Kaixin Hu 11129417

November 2015

Abstract

This article explores some properties of the Mandelbrot set. Firstly, a colourful pictures of the fractal is plotted. Secondly, Monte Carlo integration method is adopted to analyze the area A_M of the Mandelbrot set. Two sampling method is implemented, that is Pure random sampling and Latin hypercube sampling. Further, the convergence behaviour of $A_{i,s} \rightarrow A_M$ for i $\rightarrow\infty$ and s$\rightarrow\infty$ is analysed under the two sampling methods respectively.

1 Introduction of Mandelbrot set

Mandelbrot set is the set obtained from the quadratic recurrence equation

$$Z_{n+1} = Z_n^2 + C \tag{1}$$

with $Z_0 = C$, where points C in the complex plane for which the orbit of Z_n does not tend to infinity are in the set. That is of all the C in the complex plane in the Mandelbrot set, after iterations of the complex quadratic polynomial $Z_{n+1} = Z_n^2 + C$ remains bounded.

Mandelbrot set is a fractal[2], exhibiting self-similarity. And the limit at which points are assumed to have escaped can be approximated by a number instead of infinity. The magnitude of Z will either stay equal to or below 2, or it will eventually surpass 2. Once it surpasses 2, it will increase forever. So the set is bounded by a circle of radius 2, centered at the origin of the complex plane. Thus, the area is certainly less than 4π. Indeed, the area is much less than that. The left-most extent of the set ends with the spike at x = -2, and the right side extends out to approximately x = 0.47. The top and bottom are at approximately y = 1.12, respectively. [1] So in this article, the region for analysis of the Mandelbrot set area is lined out by x=-2,x=1 and y=+ - 1.5 on the complex plane, that is a square with an area of 9.

2 Method

2.1 Visualization of Mandelbrot set

Mandelbrot set can be created by coloring points depending on how quickly they diverge to $r_{max} = 2$.

First, a square is outlined by x=-2,x=1 and y=+ - 1.5 on the complex plane, and then the real part of the complex is chosen starting from -2 to 1 with 1/1000 interval, for each of these 1000 real parts, the imaginary part is chosen starting from -1.5 to 1.5 with 1/1000 interval. Thus, there 1000*1000 grids. Second, iterate each chosen complex numbers through the quadratic recurrence in Equation(1) for 1000 times, and calculate the magnitude of Z in every iteration to see whether it surpasses 2 or not. During the iteration, the time steps it takes and minus 1, to surpass 2 is assigned to the position defined by (real part,imaginary part) in the two-dimensional system, otherwise the value in the (real part,imaginary part) is 999. Finally, 1000*1000 points is colored with different colors ranging from blue to red by the values, with high value colored close to red and low values colored closed to blue.

2.2 Monte Carlo Integration Method

Monte Carlo Integration Method is used to calculate the area of Mandelbrot set.

Suppose that on the complex plane, the complex $C = x + y * i$ is uniformly distributed in the square with area 9, bounded by x=-2,x=1 and y=1.5,y=-1.5.That is, (x,y) is a random point in the rectangle region specified in figure1. So according to the Monte Carlo Integration Method, the probability of this random point in this region falling in to the Mandelbrot Set can be determined by:

$$\begin{aligned} &P\big\{(x,y) \text{ in the area of Mandelbrot Set}\big\} \\ &= P\big\{(x,y) : |Z_n| < 2 \text{ within certain length of steps}\big\} \\ &= \frac{\text{Area of Mandelbrot Set}}{\text{Area of the rectangle region}} = \frac{A_{i,s}}{9} \end{aligned}$$

i: the number of iterations in ref

s:the number of samples drawn

In this article, X and Y are sampled independently and uniformly distributed over (-2,1) and (-1.5,-1.5) respectively. Thus (X,Y)'s density function is constant in the square, consequently (X,Y) is uniformly distributed in the square. Define

$$I = \begin{cases} 1 & \text{(x,y)in the Mendelbrot set} \\ 0 & \text{else} \end{cases}$$

Then, $A_{i,s} = E(I) = \iint_{Square} p(x,y)\,dx\,dy$=P$\big\{$(x,y) in the area of Mandelbrot Set $\big\}$*Area of the rectangle region=9 * $P\big\{(x,y) : |Z_n| < 2$ within certain

length of steps$\}$. Thus the point is to calculate the probability of a random point in the rectangle falling into the Mandelbrot set. This article will use two kind of sampling method in the Monte Carlo simulation. That is the Pure random sampling and Latin hypercube sampling.

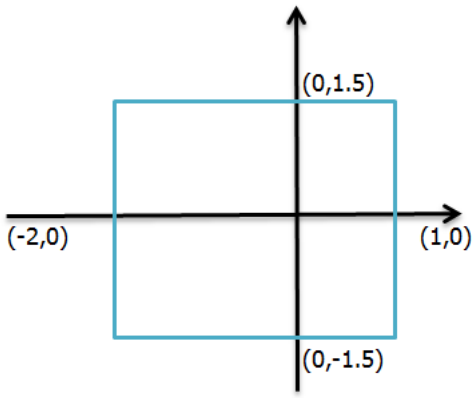

Figure 1: The square region

For every combination of the number of iterations and samples, a certain number of Monte Carlo Simulation is run. And the number run is determined by the $d = S/(k)^{0.5}$, samples' mean standard deviation. In this article, d is set to 0.01, so the mean of $A_{i,s}$ is at least 95 percent certain that will not differ from the mean calculated by 0.0196. The process is as follows:

First, for certain i and s, k=20 $A_{i,s}$ is generated.

Second, the generating process is not stopped until the d is less than 0.01.

Third, when the simulation is stopped, the mean of $A_{i,s}$ is calculated.

2.3 Sampling Method

2.3.1 Pure random sampling

Python is used, and uniform.rvs() function is adopted to generate the float random numbers. For the real part, the random numbers is generated uniformly between (-2, 1), and for the imaginary part the random numbers is generated uniformly between (-1.5, 1.5).

2.3.2 Latin hypercube sampling

In pure random sampling, new sample points are generated without taking into account the previously generated sample points. Thus large amount of samples are required to achieve a reasonably accurate random distribution, i.e.small enough standard error, in the Monte Carlo Simulation. While Latin Hypercube sampling improves this disadvantage by using a evenly sampling method.

Specifically in this article, with a certain number of samples, s, set, the probability function is split evenly into s parts, and then within each segment, the uniform.rvs() function is implemented for s-th times, resulting in s random numbers in all. And then these random numbers is shuffled and then implemented to the Equation(1).

2.4 Analysis of the convergence of A_M

Python is used to implement the Monte Carlo integration method. And the convergence behavior of is $A_{i,s}$ is analysed as i and s increases respectively when set the other equal, as showed in figure 2. And in Pure random sampling and Lain Hypercube sampling is implemented respectively in each analysis.

	convergence behaviour as i increases		convergence behaviour as s increases	
	Pure random sampling	Latin hypercube sampling	Pure random sampling	Latin hypercube sampling
number of samples	10000	10000	[500,50500]with intervals of 500	[500,50500]with intervals of 500
iterations	[100,5000]with intervals of 50	[100,5000]with intervals of 50	1000	1000
d=S/(k)^0.5	0.01	0.01	0.01	0.01

Figure 2: Analysis

2.4.1 Analysis of the convergence of $A_{i,s}$, with s set constant, i increases

In this analysis, s, i.e.the number of samples, is set equal to 10000, and i, i.e. the number of iterations, is set to the range of 100 and 5000(both included) with an interval of 50, as showed in the 2nd and 3rd column of figure2. So there are 99 combinations of set s and increasing i. Then a graph of the mean of $A_{i,s}$, i.e. $A_{i,10000}$, over each number of iterations is plotted. In each of the 99 run, d is set to 0.01, so the means of $A_{i,s}$ are at least 95 percent certain that will not differ from the mean calculated by 0.0196.

Then, the value of $A_{i+50,10000} - A_{i,10000}$ is calculated for each i to further investigate the convergence behavior. Those values are plotted on an graph over each number of iterations.

2.4.2 Analysis of the convergence of $A_{i,s}$, with i set constant, s increases

In this second analysis, s is set to the range of 500 and 50000(both included) with an interval of 500, and i is set equal to 1000, as showed in the 4th and 5th column of figure2. So there are 100 combinations of set s and increasing i. Then a graph of the mean of $A_{i,s}$, i.e. $A_{1000,s}$, over each number of samples is plotted. In each of the 100 run, d is set to 0.01, so the means of $A_{i,s}$ are at least 95 percent certain that will not differ from the mean calculated by 0.0196.

Then, the value of $A_{1000,s+500} - A_{1000,s}$ is calculated for each s to further investigate the convergence behavior. Those values are plotted on an graph over the each number of samples.

2.4.3 Analysis of the loops it needs for a certain confidential level

Since Latin Hypercube sampling has the advantage of simulating a more steady random distribution over the Pure random sampling. As mentioned above, in each run of the Monte Carlo Simulation, d is set to 0.01 to control the sample standard deviation. So the number of loops it takes to get the mean of $A_{i,s}$ will differ in the two sampling method.

First, when s is set equal to 10000, the number of loops for these two sampling methods is plotted over the number of iterations. The range of i (the number of iterations) is the the same as mentioned above, i.e. from 100 to 5000(both included) with an interval of 50.

Second, when i is set equal to 1000, the number of loops for these two sampling method is plotted over the number of samples. The range of s (the number of samples) is also the same as mentioned above, i.e. from 500 to 50000(both included) with an interval of 500.

3 Result and Discussion

3.1 Visualization of Mandelbrot set

Treating the real and imaginary parts of each number as image coordinates, the number of iterations as values, points' color differs according to how rapidly the sequence diverges, as Figure 3 shows. As the colorbar shows, With the color red, points never diverge, thus representing the Mandelbrot set. with dark blue, points surpasses 2 from the first iteration, and with light blue around the boundary of the set, they diverge after several iterations.

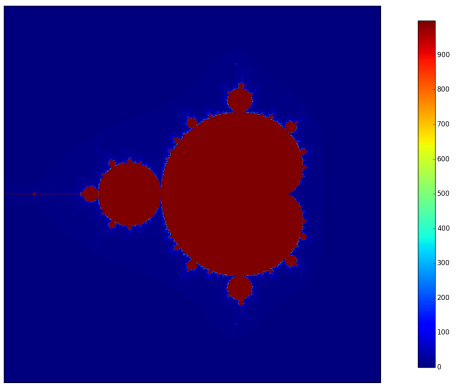

Figure 3: Mandelbrot set

3.2 Analysis of the convergence of A_M

3.2.1 Analysis of the convergence of $A_{i,s}$, with s set constant, i increases

As showed in figure 4, under both sampling methods, when s is set equal to 10000, and i increase, the area of Mandelbrot set gets close to 1.51 from around 1.54 at first. This is reasonable, for some complex numbers will surpass 2 after a large number of recurrence in Equation(1). But these two graphs indicate that after i is increased to 1000, the mean of area is fluctuating around 1.51 steadily. So $A_{i,10000}$ converge to 1.51 as in increase from 100 to 1000. After i increases up to 1000, as showed in figure 4, with at least 95 percent certainty, the actual area of Mandelbrot set will not differ from the 1.51 by 0.0196.

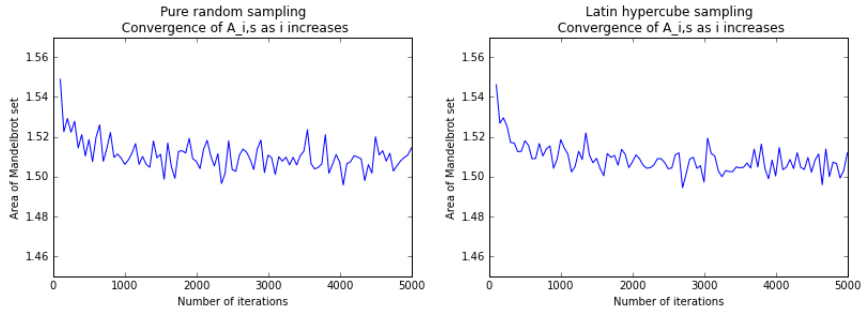

Figure 4: Convergence of $A_{i,10000}$ as i increases

Figure 5 compares the convergence behavior of A under the two sampling methods, by plotting the difference of $A_{i+50,10000}$ and $A_{i,10000}$ over i. It shows that the outcomes are similar, so it indicates Latin hypercube sampling doesn't show its advantage in this aspect, thus the number of samples is drawn large enough in order to investigate the influence of different number of iterations on the convergence.

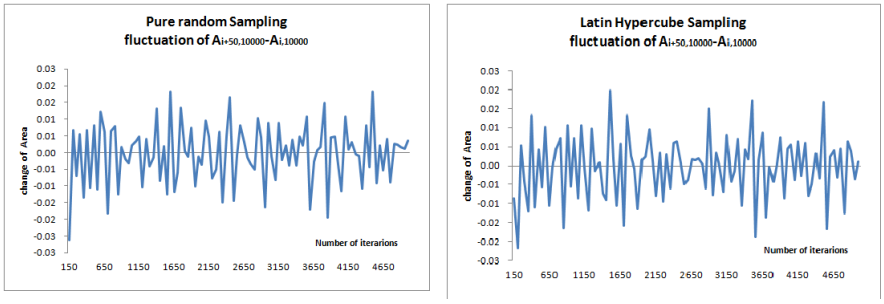

Figure 5: Fluctuation of $A_{i+50,10000} - A_{i,10000}$

But when taking into consideration the simulation it need to run to get the

the same 95 percent certainty of the mean of A, Latin hypercube sampling need obviously less loops, as showed in Figure 6. This is reasonable, as mentioned in the method part in this article that this method adopts a evenly sampling method, thus improving the simulation effiency.

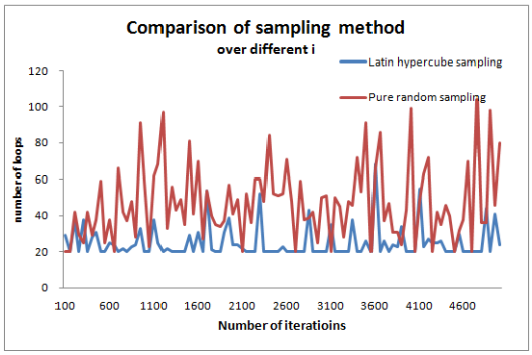

Figure 6: the number of loops for the 95 percent certain as i increses

3.2.2 Analysis of the convergence of $A_{i,s}$, with i set constant, s increases

As showed in figure 7, under both sampling methods, when i is set equal to 1000, and s increases, the area of Mandelbrot set fluctuates around 1.51, with at least 95 percent certainty that the actual area of Mandelbrot set will not differ by 0.0196. Both graphs in figure 7 show the convergence behavior, but not as obvious as in figure 4 where i increases from 100 to 1000.

There are several differences between the outcomes under Pure random sampling and Latin hypercube sampling.

As showed in figure 7 and figure 8, When Latin hypercube sampling is implemented, the mean of A becomes steady earlier than when Pure random sampling is implemented, it starts to get steady around s=2000, and since then the absolute of difference of $A_{1000,s+500}$ and $A_{1000,s}$ also lows down close to zero. While when Pure random sampling is implemented, it keeps relatively fluctuating until s increases to around 45000. What's more, the absolute of difference of $A_{1000,s+500}$ and $A_{1000,s}$ is larger than that in Latin hypercube sampling. This is reasonable, because just as mentioned in the method part in this article Latin hypercube method reduces the number of samples needed by using a evenly sampling method.

When taking figure 9 into consideration, it shows that Latin hypercube method also has the advantage of less loops to reach the 95 percent certain when the number of samples is relatively small, i..e. below around 10000 in this case.When the sample increases both method generate steady outcomes, as it showed the line is flat asymptotically to loops=20.

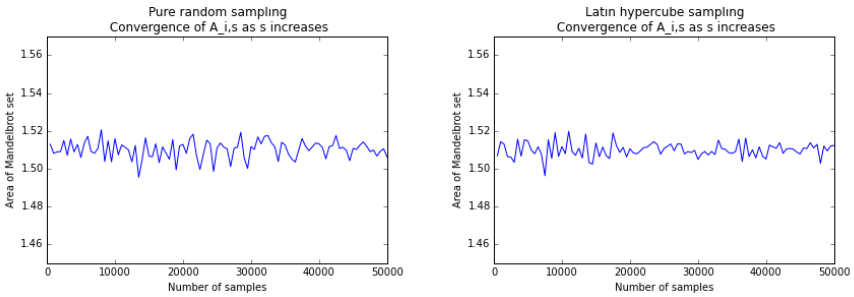

Figure 7: Convergence of $A_{1000,s}$ as s increases

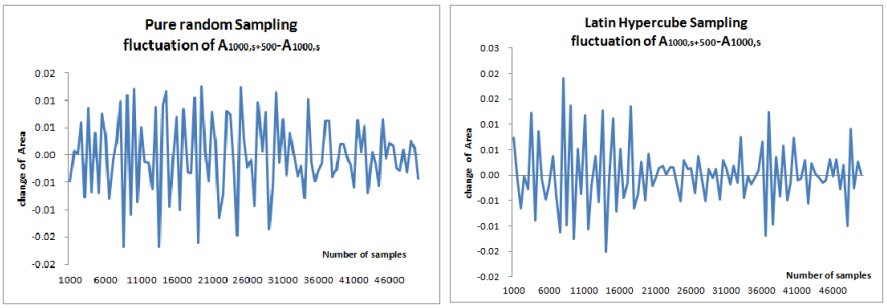

Figure 8: Fluctuation of $A_{1000,s+500} - A_{1000,s}$

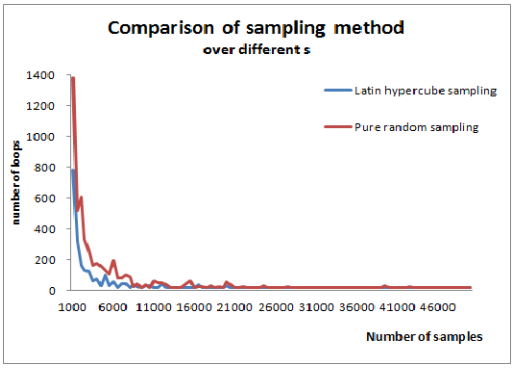

Figure 9: the number of loops for the 95 percent certain as s increases

4 Conclusion

Implementing the Monte Carlo integration method to calculate the area of the Mandelbrot set yields an estimation of $A_{i,s}$ of the area A_M for each combination of i and s. When setting one of the variables equal, and increasing the other to investigate the convergence behaviour of $A_{i,s}$, the mean of $A_{i,s}$ for the 95

percent certainty converges to around 1.51+-0.0196.

In this article, both Pure random sampling and Latin hypercube sampling method are implemented respectively, with the latter the Monte Carlo simulation efficiency is improved when the number of samples small or medium.

References

[1] Kerry Mitchell. A statistical investigation of the area of the mandelbrot set, 2001.

[2] Mitsuhiro Shishikura. The boundary of the mandelbrot set has hausdorff dimension-2. *Astérisque*, (222):389–405, 1994.