Computing the area of the Mandelbrot set

Stochastic Simulation: Assignment I

E.L. Bakels 12362980

L.H. Bijman 15211312

S. Gijsbers 12785985

In this article the area of the Mandelbrot set is estimated using Monte Carlo methods. Three different sampling techniques are compared: pure random sampling, Latin hypercube sampling and orthogonal sampling. Finally, we introduce a function for importance sampling and test the impact on convergence.

1 Introduction

The Mandelbrot set is a fractal based on the following recursive rule: $z_{n+1} = z_n^2 + c$, where $c \in \mathbb{C}$, $z_0 = 0$ and $z_i \in \mathbb{C}$ for all i-values considered. A complex number c is part of the Mandelbrot set if the sequence z_n does not diverge when n goes to infinity. When plotted on the complex plane, the set shows dynamic, self-repetitive behavior (see Figure 1). The set was introduced by Brooks and Matelski in 1978 (Brooks & Matelski, 1981), and Benoit Mandelbrot published the first high-quality visualizations of the set in 1980 (Mandelbrot, 1982). In this paper we will investigate the area of the Mandelbrot set on the complex plane (A_M). Douady and Hubbard have shown that the set is fully connected (Douady & Hubbard, 1982), therefore it is meaningful to discuss the area of the set. Furthermore, the set is compact and therefore the area is measurable (Ewing & Schober, 1992). The dynamic behavior around the edges of the set, as seen in 1b, makes it hard to determine A_M. Infinitesimally small changes in c can determine whether the sequence diverges or converges. The set is fully contained within the circle of radius two from the origin (Branner, 1989), so A_M is bounded by 4π. Here we will approximate the area of the Mandelbrot set using the Monte Carlo method with random sampling, Latin Hypercube sampling, and orthogonal sampling. We will compare the Monte Carlo areas to an area computed by pixel counting. Lastly, we will investigate the impact of importance sampling on the rate of convergence of the area calculation.

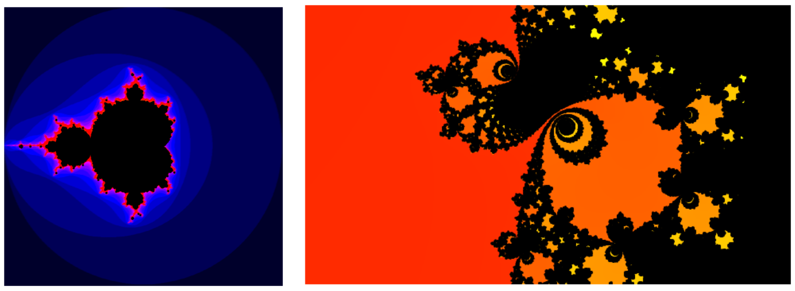

(a) (b)

Figure 1: The Mandelbrot set on the complex plane ($100 \cdot 100 = 10^4$ points are plotted in both figures). The colors represent the number of iterations before reaching the divergence criterion (see 2.1). Brighter blue and red colors denote slower divergence than darker colors. The black points denoted as part of the Mandelbrot set do not reach the threshold before the amount of maximum iterations (i). For Figure 1a we have $[-2, 2] \times [-2, 2]$, and $i = 200$, for 1b bounds $[-0.750222, -0.749191] \times [0.031161, 0.031752, 2]$, and $i = 500$.

2 Methods and theory

2.1 Mandelbrot set

To determine whether a point is part of the Mandelbrot set, we use the circle with radius 2 around the origin. Since the Mandelbrot set is contained within this disc, we know a point is not part of the set when $\|z_n\|_2 = \sqrt{a_n^2 - b_n^2} \leq \sqrt{4} = 2$ after $n < i$ iterations. For any given $c \in \mathbb{C}$, if the $\|z_n\|_2 < 2$ during the i iterations, c is considered part of the Mandelbrot set. Whenever the z_n value went out of the bounds, the complex number was not part of the Mandelbrot set, and the number of iterations it took to diverge (before $\|z_n\|_2 < 2$) was recorded.

For Figure 1, the implementation was visualised using coloured dots in a complex plane from -2 to 2 in both real and imaginary axes. Black dots were used to represent complex numbers that are part of the Mandelbrot set. The other complex numbers use a continuous colour scheme, where their recorded number of iterations determined the specific colour. Complex numbers that are close to the Mandelbrot set tend to use more iterations and thereby show slower divergence. Points that are far from the Mandelbrot set diverge almost immediately. To visualise this difference, the colour scheme was divided into groups. Because the majority of points in our complex plane diverge immediately, these points were allocated to two colour groups. The nearby points formed the third group.

After implementing the iterations and visualising the Mandelbrot set with points, the area of the Mandelbrot set was estimated with pixel counting. All the black points in the Mandelbrot set divided by the total point in the plane gives the fraction of the area that is part of the Mandelbrot set. Multiplying this fraction with the area of the plane gives a good estimate of the area A_M of the Mandelbrot set, and will be used to compare the results in the following experiments. We used a grid of 10000×10000 points for pixel counting.

2.2 Monte Carlo integration

Monte Carlo integration provides a way to estimate the area underneath a curve (that is hard to determine analytically) using the expected value. In general, we are interested in estimating the integral $\mu = \int f(x)p(x)\mathrm{d}x$, where $f(x)$ is the function of interest and $p(x)$ its probability density function. By the central limit theorem, μ can be estimated by the expected value of $f(x)$ when sampling from the distribution $p(x)$. We have

$$\mathbb{E}_p(f(x)) = \frac{1}{N}\sum_{i=1}^{N} f(x_i) \tag{1}$$

where N is the number of independent samples and x_i are sampled from $p(x)$. This estimator converges to μ (Oates, Girolami, & Chopin, 2017). For the Mandelbrot set, $f(x)$ can be seen as the function that determines whether a point is in the set or not. Therefore, we can estimate the area by randomly sampling in the $[-2, 2] \times [-2, 2]$ grid and calculating the expected number of points that are in the set over the total number of points sampled (as in equation 1). For a predetermined maximum number of iterations i and a number of samples s, this yields an estimate $A_{i,s}$ of the area A_M. Because there is a finiteness of i and s and a limited computing power available, i and s were balanced so that the errors of them are comparable. For this the convergence behaviour from $A_{i,s}$ to A_M was investigated using the error $(A_{j,z} - A_{i,z})$ was calculated as a function of j computing all $A_{j,z} \forall j < i$. Hereby, the lowest i that caused a minimal error was chosen to use in further experiments. In addition to this, different sample sizes were investigated in the same matter (with $z < s$), and also the lowest sample size that shows minimal error was chosen for further experiments. After obtaining a good estimate for both s and i, the area is determined using three methods: random sampling, Latin hypercube sampling, and orthogonal hypercube sampling. Lastly, importance sampling was applied to improve the convergence.

2.3 Sampling methods

We use three different sampling methods to sample points from the complex plane:

i) **Pure random sampling** involves selecting samples randomly from the complex plane, where each complex number has an equal chance of being chosen (Ross, 2022). To implement this sampling method we used the random function of Python's numpy library.

ii) **Latin Hypercube sampling (LHS)** is a quasi-random sampling technique, that is used for stratifying the sample space. It divides the sampling plane into a grid, whereby the samples are taken from different places in the grid, with one sample taken from each row and column, thereby creating a more equal distribution (Ross, 2022). To implement this sampling method we used the LatinHypercube function of Python's scipy.stats.qmc library with dimension = 2.

iii) **Orthogonal sampling** is an expansion of LHS. It divides the sampling plane into equal sub-domains, while ensuring an equal allocation of samples within each domain. The goal of the orthogonal sampling method is to reduce correlation among sampling dimensions, counteracting the inherent negative correlation found in LHS (Ross, 2022). To implement this sampling method we used the LatinHypercube function of Python's scipy.stats.qmc library with dimension = 2 and group = 2.

2.4 Importance sampling

After obtaining a good estimate for both s and i and comparing the three methods, we aim to improve the convergence rate by implementing importance sampling. This technique is mostly used to reduce the variance between runs. We can multiply $\mu = \int f(x)p(x)\mathrm{d}x$ with $\frac{g(x)}{g(x)}$, where g is another probability density function. We then obtain

$$\mu = \int \frac{f(x)p(x)}{g(x)} g(x)\mathrm{d}x = \mathbb{E}_g\left[\frac{f(x)p(x)}{g(x)}\right] \tag{2}$$

If $g(x)$ is chosen in such a way that it is large when $f(x)p(x)$ is large, the variance is decreased (Ross, 2022). The choice of $g(x)$ can increase convergence as well, since sampling from more likely values can lead to a faster convergence.

For the Mandelbrot set, we can see $f(x)$ as the function that determines whether a point is part of the Mandelbrot set (after i iterations and for s samples) and $p(x)$ a uniform distribution over the complex plane. To determine $g(x)$, we divide the complex plane in three blocks (see Figure 2). It is likely that most of the points in the middle block are part of the Mandelbrot set, and unlikely that points from the outer block are part of the set. In the middle block, we roughly captured the boundary of the set. We determine $g(x)$ in such a way that it samples relatively much in the inner block and little in the outer block. In Table 1 we show the probabilities of sampling in the different blocks based on their area and after importance sampling. We sample the largest part of the points in the second block, but compared to the area the number of points in the inner block increases the most.

Table 1: Renewed probability of sampling within different regions of the complex plane for importance sampling. P_{area}(sampling) is the probability of sampling within a block based on its area, and P_{imp}(sampling) is the probability of sampling with importance sampling.

Block	P_{area}(sampling)	P_{imp}(sampling)	Factor
1. (inner)	$1/25 = 0.04$	0.1	$0.1/0.04 = 2.5$
2. (boundary)	$11/25 = 0.44$	0.7	$0.7/0.44 \approx 0.76$
3. (outer)	$13/25 = 0.52$	0.2	$0.2/0.52 \approx 0.385$

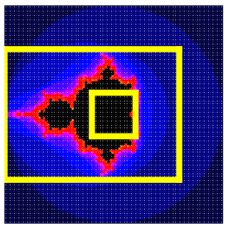


Figure 2: In this picture of the Mandelbrot set, the yellow lines represent different blocks for importance sampling. The edge of the Mandelbrot set (thick red edge) is roughly between the yellow lines. This sample region will have a higher density of points in the experiment.

2.5 Comparing the convergence rates

In our experiment, the convergence to max iterations and to the max sample size is investigated, as was mentioned in section 2.5. To compare the convergence rates of the different sampling methods, there will be examined at which point the data appears to converge to the particular equilibrium value. There is expected that because of the implementation of importance sampling, the data converges more rapidly to max interactions and max sample size. For every method, the first mean value that falls between the 95 confidence interval of the last point is chosen. This means that the mean of this particular point (i) corresponds to the last point (n). To test if there is convergence between point i and point n, the equation of sample variance is used:

$$S^2 = \frac{\sum_{i=1}^{n}(x_i - \bar{x})^2}{n-1} \tag{3}$$

In this equation, the means for every point from i until n is compared with the sample mean. If the $S**2$ is low, there is a low variance between the points, which means after the chosen point i the data seems to converge.

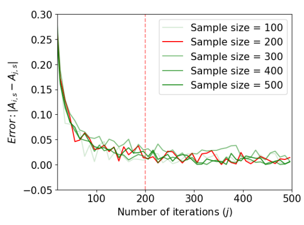
3 Results

3.1 Mandelbrot set

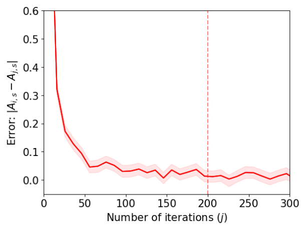
Firstly, we implemented the Mandelbrot set and created the images shown in Figure 1. In Figure 1b we can see that the boundary of the Mandelbrot set shows complex and dynamic behavior. We can see that the points at the border of the complex plane diverge faster (indicated by their darker color). The number of iterations before the threshold was met (n) is very low. The red points near the boundary diverge slowly, indicated by a large n, closer to the maximum amount of iterations set. Furthermore, we observe in Figure 1b that there are no disconnected regions within the set, therefore the area can be computed.

3.2 Monte Carlo integration

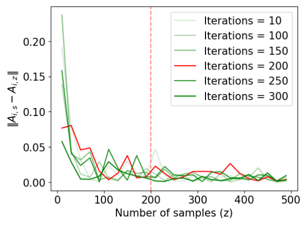
In this section, we implement Monte Carlo integration with random sampling. The results are shown in Figure 3.



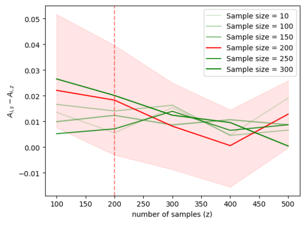
(a)



(b)



(c)



(d)

Figure 3: Convergence of Monte Carlo integration for different numbers of maximal iterations (3a,3b) and for different sample sizes (3d). The red lines show the values that were chosen for further analysis. The average and confidence interval is shown over 1000 runs.

In Figure 3a and 3b we can see that the error does not decrease by a lot after $i = 200$. This is highlighted for $s = 200$ in 3b, where we can see that the confidence interval does not decrease as j increases. Therefore, we choose $i = 200$ as value for the maximum amount of iterations. In Figure 3d we can see that this is similar for the sample size (z). Therefore we set $s = 200$ for the further analysis. We have chosen i and s such that they are the lowest possible (to decrease computation time) while causing the minimal error. In the following sections, we perform similar tests for the different sampling techniques. We can determine whether the different properties of the sampling methods influence the convergence behavior.

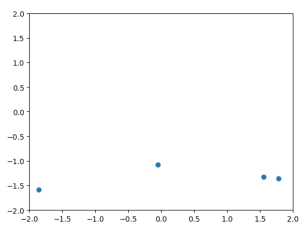
3.3 Sampling techniques

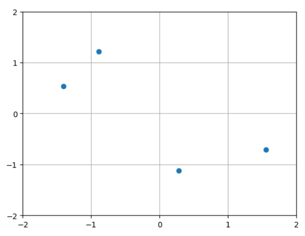
In the following section we compare the estimated area for different sampling methods: random sampling, Latin hypercube sampling, and orthogonal sampling. Furthermore, we compare the estimated area to the area obtained by pixel counting. In Figure 4 the difference between the sampling methods is visualized. In Figure 4b we can see that the Latin hypercube sampler does not sample twice in the same row or column. In Figure 4c we see that this goes for orthogonal sampling as well. Furthermore, the space is divided in equal sub-spaces and no more than one point is sampled from each subset.

Table 2: T-test for pure random sampling, Latin hypercube sampling, and orthogonal sampling tested against the pixel counting strategy and the area with large i and s (1000 runs, 289 samples, 200 iterations). Pixel area value = 1.525, for Area$_{i,s}$ we have $i = 500$, $s = 841$

Sampling technique	**Mean area**	**St. dev.**	**t-test for pixel counting**		**Area$_{i,s}$**	**t-test**		**Confidence interval**	
			t	***p***		***t***	***p***	*low*	*high*
Random	1.514	0.280	1.435	0.151	1.501	-1.374	0.170	1.496	1.531
Hypercube	1.508	0.264	-2.189	0.029	1.509	-0.219	0.827	1.491	1.524
Orthogonal	1.504	0.264	-2.664	0.0078	1.513	-1.172	0.241	1.487	1.520

We denote that these visualizations are for explanatory reasons, and the grids shown do not necessarily coincide with the grids made by the Python function.


(a) Pure random


(b) Latin hypercube

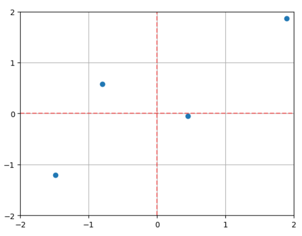

(c) Orthogonal

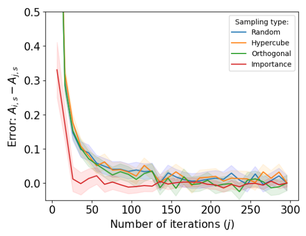
Figure 4: Visualisation of the different sampling techniques with four samples taken. The red lines in 4c indicate the division of the plane into equal subdomains.

In Table 2 we show the outcomes of statistical analysis of the areas determined with the different sampling techniques with $i = 200$ and $s = 289$. The sample size changed to this number because orthogonal sampling requires the square of a prime. In the first section, we compare the areas to the area obtained by pixel counting with 10.000×10.000 points (1.525). We observe that the mean of the area obtained with random sampling is not significantly different from the pixel counting area. This is the case for Latin hypercube sampling and orthogonal sampling. this is an indication that Latin hypercube sampling and orthogonal sampling are different from random sampling. No significant difference between the estimated values and the pixel counting area is advantageous as it implies comparable performance.

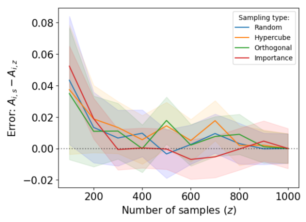
In the second section of the table, we compare the estimated area to the area determined by the same sampling method with $i = 500$ and $s = 841$. None of the means are significantly different for a high amount of iterations and sample sizes. This shows that the estimated area values for the different sampling techniques do not perform significantly worse then the same techniques with higher iterations and sample size. Two-sample sample t-tests showed that the difference in estimated area for none of the sample methods is significant (Random - Latin Hypercube: $t(1998) = 0.504$, $p = 0.615$, random - orthogonal: $t(1998) = -0.830$, $p = 0.407$, orthogonal - Latin Hypercube: $t(1998) = -0.0336$, $p = 0.737$).

3.4 Importance sampling

In this experiment, importance sampling was implemented (as described in 2.4) to further improve the convergence rate of the Monte Carlo approach. The results for different numbers of iterations (i) and different numbers of samples (z) are shown in Figure 5a and Figure 5b respectively. In Table 3 we show the variance for the different methods.



(a)



(b)

Figure 5: Error of the estimation for the area for different values of the maximum iterations (j, 5a, sample size = 200) and for different sample sizes (z, 5b, $i = 200$).

Table 3: Variance for different values for the maximum iteration and for the different numbers of samples.

Sampling type	Iterations			Samples		
	Var	j	Var(after j)	Var	z	Var(after z)
Random	5.439e−2	136	1.113e−4	1.779e−4	300	2.286e−5
Latin Hypercube	5.640e−2	106	1.924e−4	1.293e−4	400	4.567e−5
Orthogonal	5.554e−2	106	2.285e−4	1.144e−4	400	4.334e−5
Importance	4.600e−3	36	5.810e−5	3.112e−4	300	1.282e−5

We observe that the error for Monte Carlo integration combined with importance sampling is decreases for lower numbers of iterations (j) than the other sampling methods (5a). In Table 3 we see that the estimated area is below the upper bound of the confidence interval of importance sampling with 296 iterations when $j = 36$. This value is lower than for the other sampling techniques. Furthermore, the variance between runs is decreased by a factor of 10 (both overall and for the values larger than j).

We do not observe a faster decrease in the error for the different number of samples (5b. Furthermore, the variance of importance sampling is similar to the variance for the other methods (3), as is the value z for which the estimated area is below the upper bound of the confidence interval of the estimated area with 1000 samples. We denote that in our experiment, z increases in steps of 100, so the difference between 300 and 400 is one step.

We found an the average of the estimated area with importance sampling of 1.586, with $i = 200$ and $s = 289$. This area is significantly larger than the pixel counting area ($t(1998) = 10.00$, $p < 0.001$). Furthermore, we found that the area is significantly larger than the area computed with importance sampling with $i = 500$ and $s = 842$ (1.567, $t(1998) = 3.104$, $p = 0.002$). This means that there is no indication that the average of the estimated area with importance sampling converges to the area as determined with many iterations and many samples. Two-sample independence t-tests showed that the average for importance sampling is significantly different than the average of the other sampling methods (random - importance: $t(1998) = 6.765$, $p < 0.001$, orthogonal - importance: $t(1998) = 8.010$, $p < 0.001$, orthogonal - Latin Hypercube: $t(1998) = 7.626$, $p < 0.001$).

4 Conclusion and discussion

In section 3.2 we observed that an increase in the maximum number of iterations does not decrease the error when it is above approximately 200. This is an indication that it is unlikely for a complex number to diverge if $\|z_i\| < 2$. We also observed that an increase in number of samples does not decrease the error. However, this could be the case when larger numbers are tested.

In section 3.3 we observed that only random sampling does not significantly differ from the area calculated with pixel counting. This result is unexpected, since we expected the Hypercube and orthogonal samples to be more accurate and therefore closer to the value determined by pixel counting. One explanation for this result is that the area determined by pixel counting might be biased. As the boundaries show infinitesimal differences, this means that, even with a 10.000×10.000 grid, many points in the boundary area are skipped. Furthermore, we used the $[-2, 2] \times [-2, 2]$ plane to sample from. The Mandelbrot set is contained within a smaller area of this set. Sampling between these limits only can lead to a more accurate estimation of the area, both with pixel counting and the other (pseudo-)random sampling methods. The average estimated area of none of the sampling methods was found to differ significantly. This indicates that the sampling methods do not differ significantly when applied in this context. One possible explanation is $[-2, 2] \times [-2, 2]$ plane, where points outside of the Mandelbrot set are sampled with relatively large probability (see Table 1). For these points, the difference between the sampling methods has a smaller effect.

In section 3.4 we observed the effects of importance sampling. The decrease in error for a lower maximum number of iterations can be explained by the fact that our version of importance sampling samples relatively many points inside the Mandelbrot set. Since those points converge, a lower cut-off value does not increase the error of the estimation. Thereby, importance sampling in the way it was implemented here can decrease the runtime by decreasing the maximum amount of iterations needed per point. The variance for the number of iterations is decreased by a factor 10 with this type of importance sampling. For the number of samples needed, the variance is not reduced with importance sampling. One explanation for this is that our implementation of importance sampling samples more points in the boundary of the Mandelbrot set. Since there is much variance in that area between points, this might increase the need for more samples. Another way to implement importance sampling is by increasing the importance of the boundary of the Mandelbrot set. It is beneficial to sample more points there, since the stark difference between convergent and divergent points in that area has a great influence on A_M. Our approach could be altered by increasing the probability of sampling in block2, or by making the boundaries between the different blocks more specific to the boundaries of the set. One should take into account the above concern about the increase of variance when sampling more in the boundary region.

The area found with importance sampling (1.586) was significantly larger than the areas found with the other methods. Furthermore, the estimated area for importance sampling with larger i and s significantly differs from the area with $i = 200$ and $s = 200$. A possible explanation is that a higher value of i is needed to determine whether the points in the boundary of the Mandelbrot set converge or diverge.

Overall, we conclude that in our implementation, there is no significant difference between the three sampling methods discussed. Importance sampling shows significant difference with all sampling methods and could reduce the running time by increasing the convergence for lower iteration numbers. However, this should be done with care as the estimated area with a lower iteration number might not be an accurate predictor for the area with larger iteration numbers.

References

Branner, B. (1989). The mandelbrot set. In *Proc.symp.appl.math* (Vol. 39, pp. 75–105).

Brooks, R., & Matelski, J. P. (1981). The dynamics of 2-generator subgroups of psl (2, c). In *Riemann surfaces and related topics* (Vol. 1).

Douady, A., & Hubbard, J. (1982). Itération des polynômes quadratiques complexes. *CR Acad. Sci. Paris Sér. I Math.*, *294*, 123–126.

Ewing, J. H., & Schober, G. (1992). The area of the mandelbrot set. *Numerische Mathematik*, *61*(1), 59–72.

Mandelbrot, B. B. (1982). *The fractal geometry of nature* (Vol. 1). WH freeman New York.

Oates, C. J., Girolami, M., & Chopin, N. (2017). Control functionals for monte carlo integration. *Journal of the Royal Statistical Society Series B*, *79*(3), 695–718.

Ross, S. M. (2022). *Simulation.* Academic Press.

5 Work distribution

Author	**Surviving Lines**	**Commits**	**Files**	**Distribution**
Esther Bakels	2479	10	2	47.3/19.2/18.2
Sacha Gijsbers	1942	25	7	37.0/48.1/63.6
Loes Bijman	824	17	2	15.7/32.7/18.2