P5

edgardjfc

March 2022

1 Introduction

For this assignment we studied the behaviour of the Monte Carlo method of obtaining random numbers based on a given formula. The formula in this case must be normalized in order to give results that are evenly spread. The objective for the assignment was to use the given base code by our teacher Dr. E. Shaeffer [2] and use it to obtain statistics on the margin of error that the approximation of our software has compared to the approximation of Wolfram Alpha.

2 Code

```
generator = GeneralRandom(np.asarray(X), np.asarray(Y))
start = 3
finish = 7
chunk = 50000
ammounts = [500, 5000, 50000]
wolfram = 0.04883411112604931084
ae = []
se = []
dec = []
points = []
```

With this initial part of the code we set up the variables that we'll use for the experiment along with the different lists we'll use to save the data for each particular figure we'll produce.

```
def part(replica):
    V = generator.random(chunk)[0]
    return ((V >= start) & (V <= finish)).sum()

def compare_strings(a, b):
    a = str(a)
    b = str(b)
    if a is None or b is None:
        return 0

    size = min(len(a), len(b))
    count = 0

    for i in range(size):
        if a[i] == b[i]:
            count += 1
        else:
            break
    return count
```

Here we define our functions in order to generate one random number within the specified range of our initial conditions. The second function is used to compare the string of text and see how many digits of accuracy it has compared to the results from Wolfram Alpha.

The second formula was taken from the code worked on by a classmate with the alias FeroxDeitas [1].

3 Results

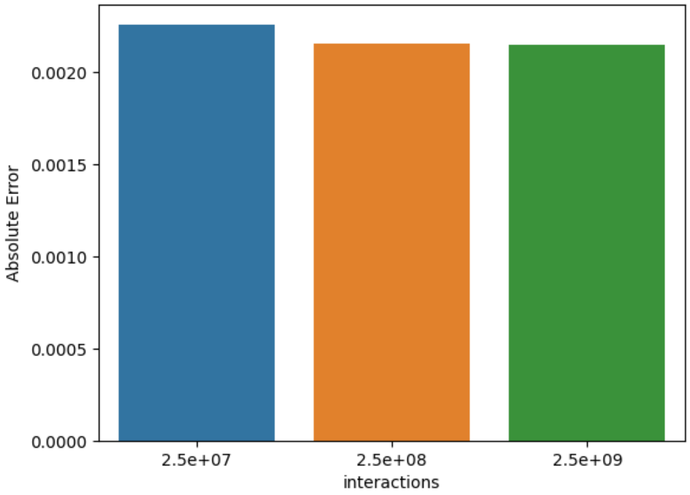

Figure 1: Absolute Error

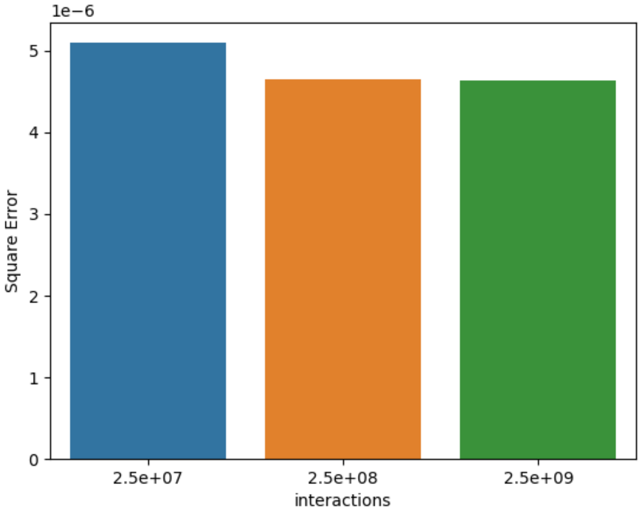

Figure 2: Square Error

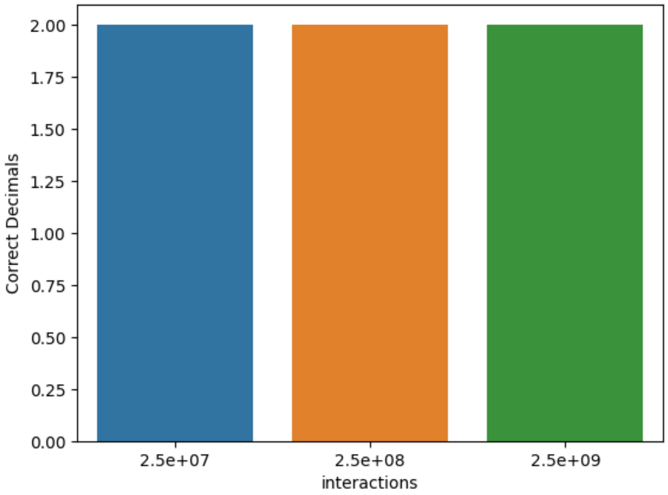

Figure 3: Accurate decimals

4 Interpretation

We can observe in fig. 1 and fig. 2 how the margin of error decreases as the amount of interactions increases. This makes sense, since the number of interactions give us a more accurate approximation towards the Wolfram Alpha value. However, in fig. 3 we can observe how the difference between these number of interactions is not significant enough to mark a difference in the figure, where all the correct digits are 2 in every experiment.

5 Conclusion

We can conclude that with an increasing amount of interactions it makes it a more accurate approximation towards whatever it is being calculated.

References

[1] J.FeroxDeitas. *GitHub,pasos$_a$leatorios.py*. URL: `https://github.com/FeroxDeitas/Simulacion-Nano/blob/main/Tareas/P5`.

[2] E. Schaeffer. *GitHub,MonteCarlo*. URL: `https://github.com/satuelisa/Simulation/tree/master/MonteCarlo`.