Monte Carlo Exercise 1

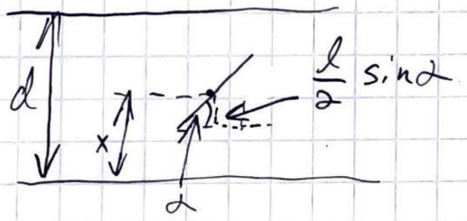
$\boxed{1}$

Buffon's needle

l - needle length
d - distance between lines $\quad l << d$

A needle of size l is thrown across a floor with uniform parallel lines with distance d. what is Phit of a line?

It can be illustrated as:



α - is the angle of the needle with respect to floor lines

x - is the position of the needle with respect to closest line

the needle crosses a line if $x \leq \frac{l}{2}\sin\alpha$
and $\frac{l}{2}\sin\alpha$ is the projection of $\frac{l}{2}$ onto the "y" axis

now the probability density function of x is $\frac{2}{d}$ as both halves are the same
and the probability density function of α is $\frac{2}{\pi}$ as again both halves of the circle are the same.

we need both probabilities, hence taking the product while using $x \leq \frac{l}{2}\sin\alpha$ and integrating.

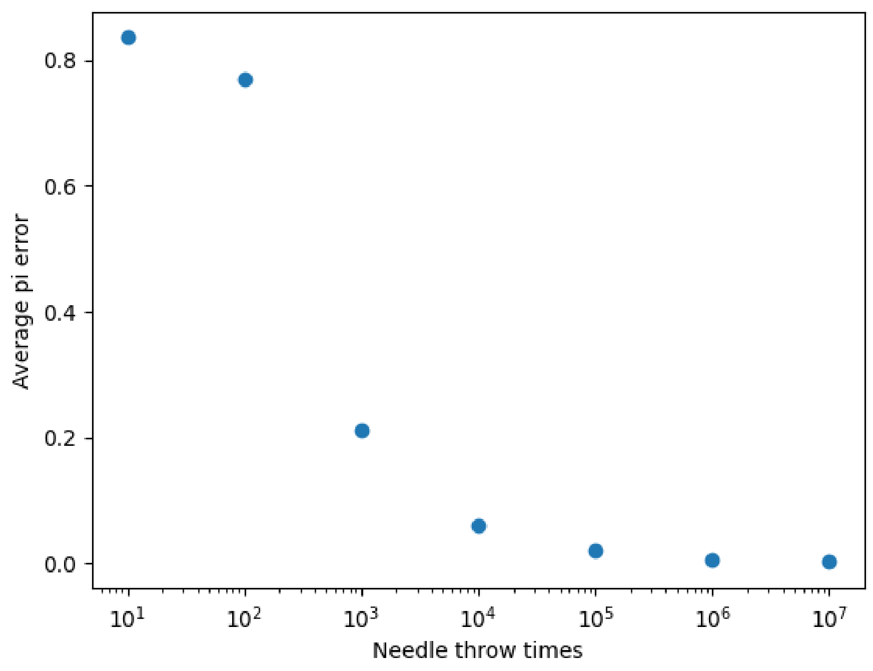
$$\int_0^{\pi/2} d\alpha \int_0^{l\sin\alpha} dx \, \frac{4}{\pi d} = \int_0^{\pi/2} d\alpha \left(\frac{4}{\pi d} x \Big|_0^{\frac{l}{2}\sin\alpha} \right) = \int_0^{\pi/2} d\alpha \, \frac{4}{\pi d} \, \frac{l\sin\alpha}{2}$$

$$= \frac{4l}{\pi d} \frac{\cos\alpha}{2} \Big|_0^{\pi/2} = \underline{\underline{\frac{2l}{\pi d}}} = \text{Phit} = \frac{\text{num of hits}}{\text{num of throws}}$$

now to find π we can $\pi = \dfrac{2 \cdot l(\text{num of throws})}{d \cdot (\text{num of hits})}$

2

Running it with N = 100 indeed took quite a while, especially for n = 10^7.
Here are the results of the average error from the numpy defined pi:



Values plotted are:
[0.83591741, 0.7705188, 0.2124989, 0.05913755, 0.02042019, 0.00633254, 0.00207057].

Seems that at no point the average error is smaller than 0.0001 here. For 10^7 we get 0.00207057.
But, looking at the trend of the graph, where the error becomes ~3 times smaller each step I would say that at 10^10 we would have the wanted error.

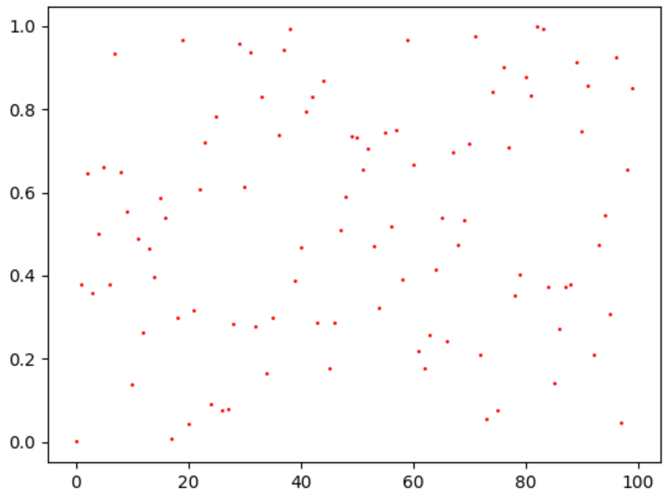
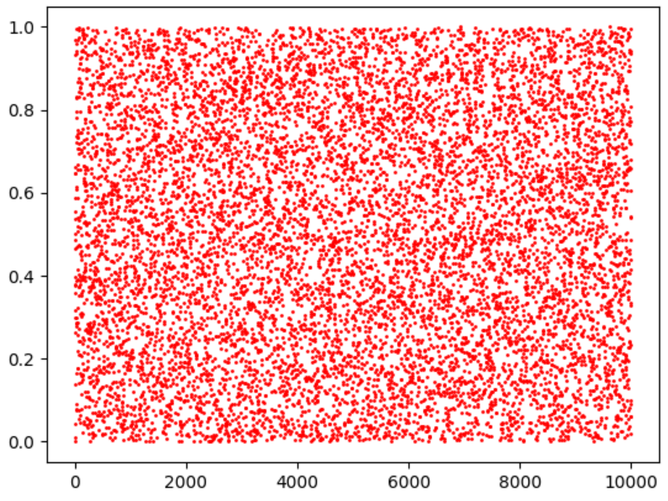
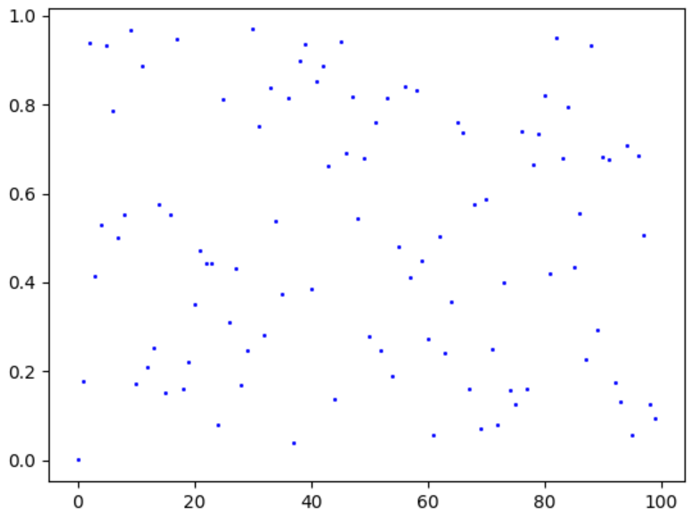
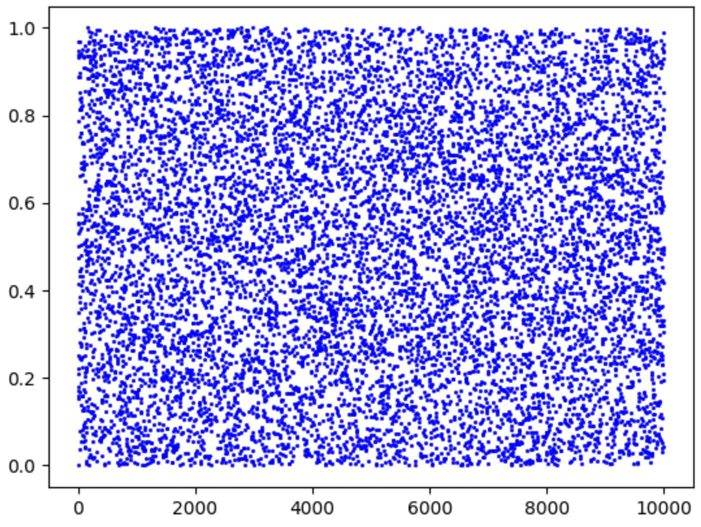
3

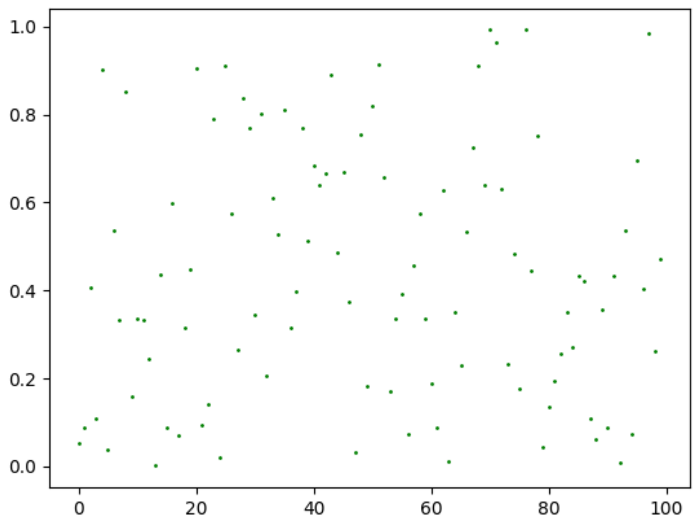
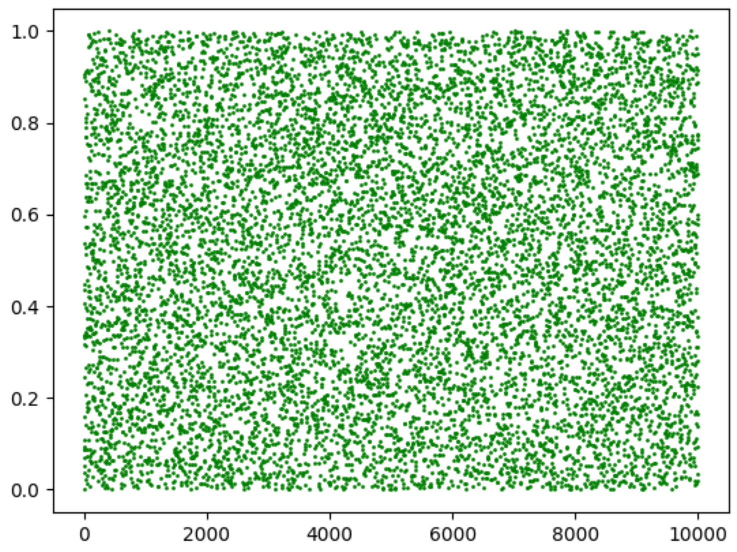
B: I defined a “repeat” as an event where we encounter a series of 10 repeater number Using the func find_repeat which can be used for all 3 types I generated numbers and saved them in an array (before to a dictionary, but the array has better performance) until I hit 10 consecutively repeated numbers.

LCG (a = 587 c = 1019): The repeat period is 31690 (minus 10). It is much smaller than the theoretical period which is simply *m* in this case.
Park-Miller: I have run it for a very long time, while trying to optimize performance for faster repetition testing and I got to 710000000 generated numbers.
The theoretical period of Park-Miller is 2^31 - 1 which I have evidently not reached. Maybe if I would run it for the whole night.
Twister: This one should have a very large period which I found online to be (2^19937 – 1)

4

Here are the plotted LCG (red) , PM (blue) and Twister(green) schemes for 100 and 10000 points

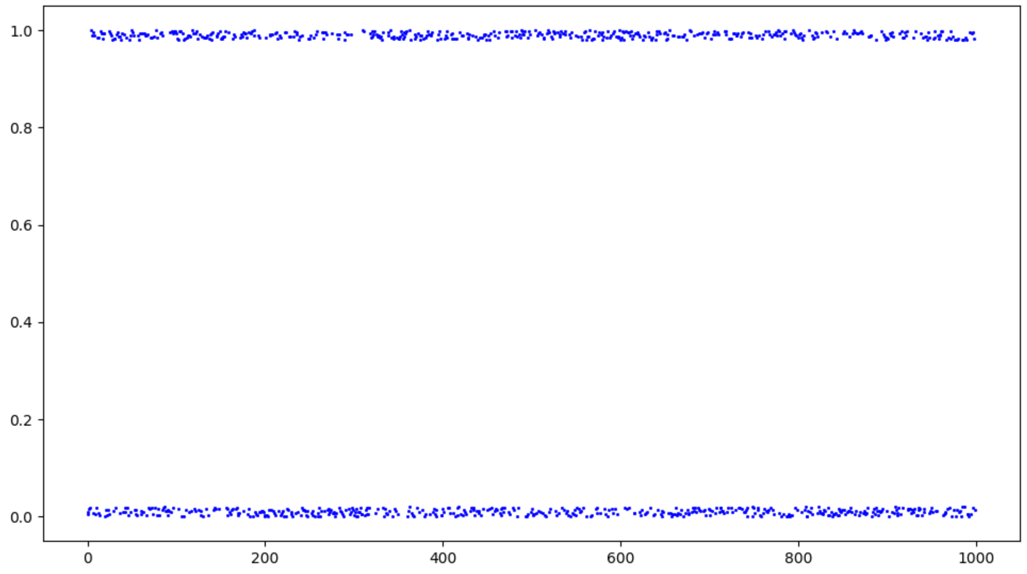



At the 100 points versions, I think that already the Park-Miller has a nice scatter (density) compared to the basic LCG.

When comparing the 10000 points plots I notice that at LCG and PM we see a not-small amount of “bald” areas and condensed areas. Then only the twister scheme (which is python’s random here) seems to cover the entire spectrum well enough.

Plotting a smaller range:

Now when I plot only at the range of < 0.02 and > 0.98 (I hope I understood correctly the assignment) I get the following scatter:



The problem I see with it is that we have again areas with smaller and higher density of probability. At class it was mentioned that this happens, it tends to generate numbers in the same “neighborhood”

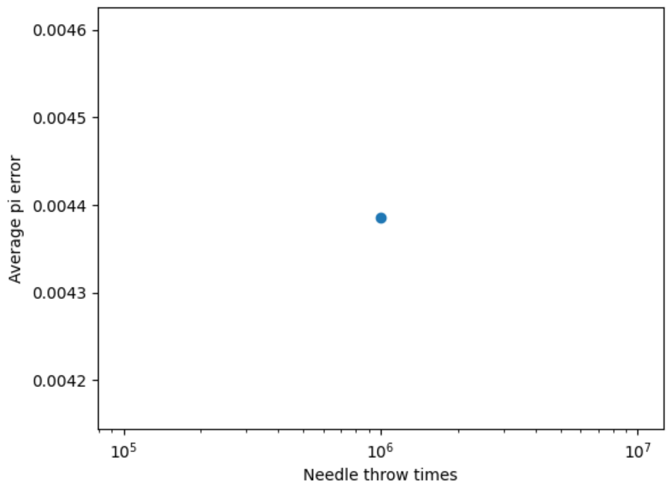
Changing the seed didn’t help, now I have just different positions for the different densities.

Using LCG for buffon’s:

Using our basic LCG from the lcg.py file, the generation is much slower than before, and we are testing it only for the 10^6 group.

I got 0.00438522 as the average error, which is actually better than the previous results (0.00633254) using python’s random.random().

I don’t know why, maybe the seed I chose was somehow better for this case?


Average pi error
10^5
10^6
10^7
Needle throw times