Course - CVL867
Assignment 2 Solutions
Instructor: Anoop Krishnan

Utkarsh Pratiush
2022SRZ8573

February 15, 2023

Code:

0.1 Description of code:

a) utils/rng.py — functions to generate random number

b) utils/random_walk.py — Implements random walk in 1d and higher dimensions. Functions for finding when collision will happen and mean square displacement of the random walk.

c) utils/plot.py — utility functions to plot

d) main.py — i) Function returning list of average displacement for multiple simulations of different jump lengths. ii) Implements all the questions at one place, inside main function.

0.2 How to run the code:

python main.py 1 : return solution for Q1
python main.py 2 : return solution for Q2
... so on till Q6

1 Implement an algorithm to run a 1D random walk with 1000 jumps. Take all constants equal to 1. Plot x and x2 with respect to the step number.

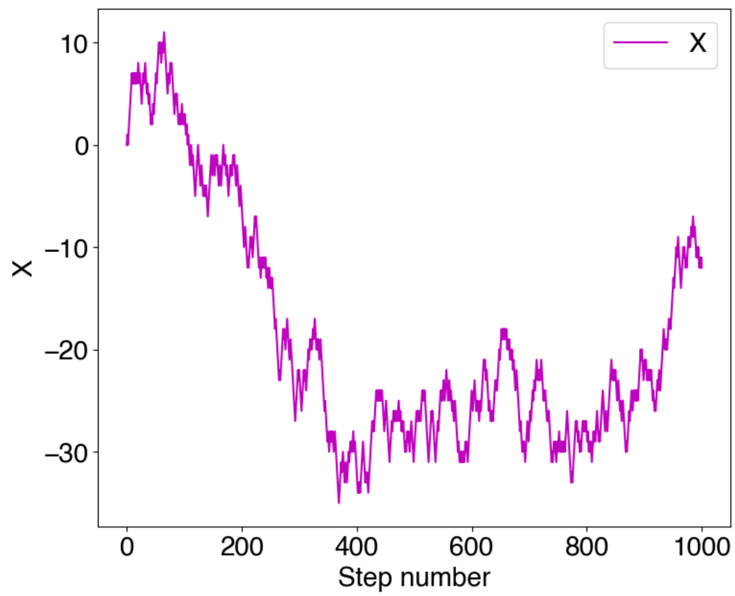

Figure 1: X v/s Step

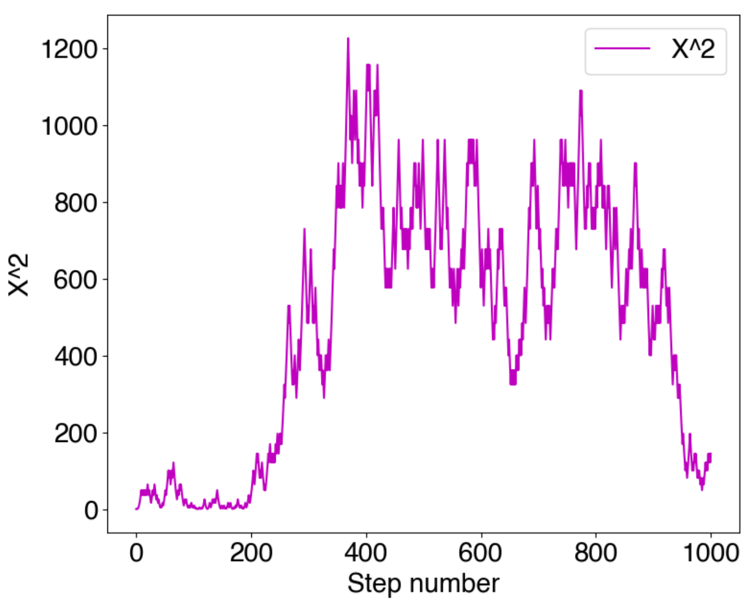

Figure 2: X^2 v/s step

2 By averaging multiple simulations, check that the mean-square displacement tends to a linear function of the number of steps, with a slope of 1.

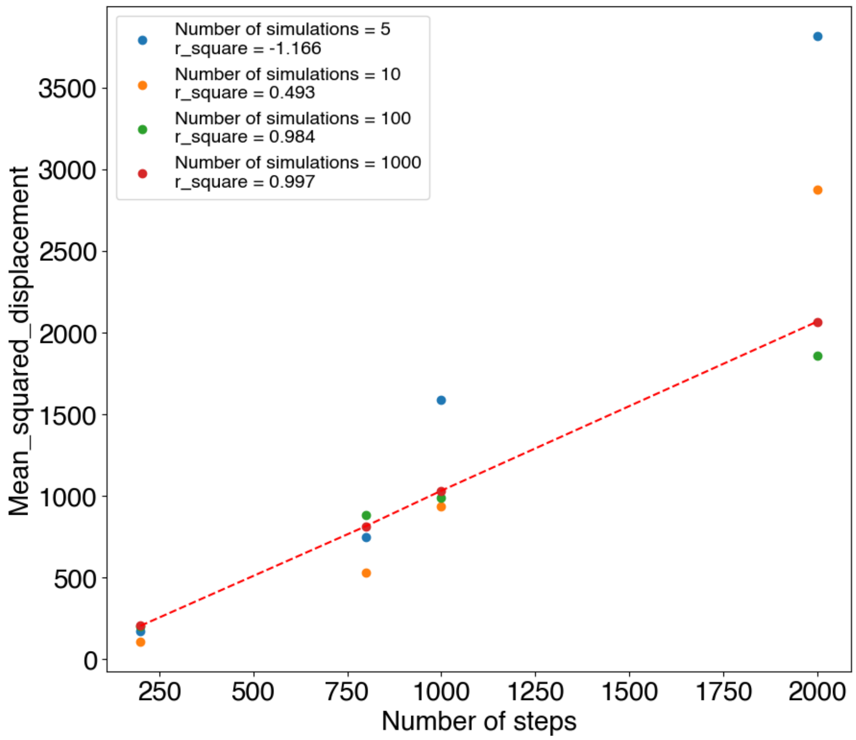

Figure 3: Plot showing Rsquare tending to 1 as number of simulations increasing.

3 Check the effect of a "biased" random walk, that is, when the probability to jump to the right is not 50% (try values between 0 and 100%). How does it affect the shape of the average mean-square displacement with respect to the step number?

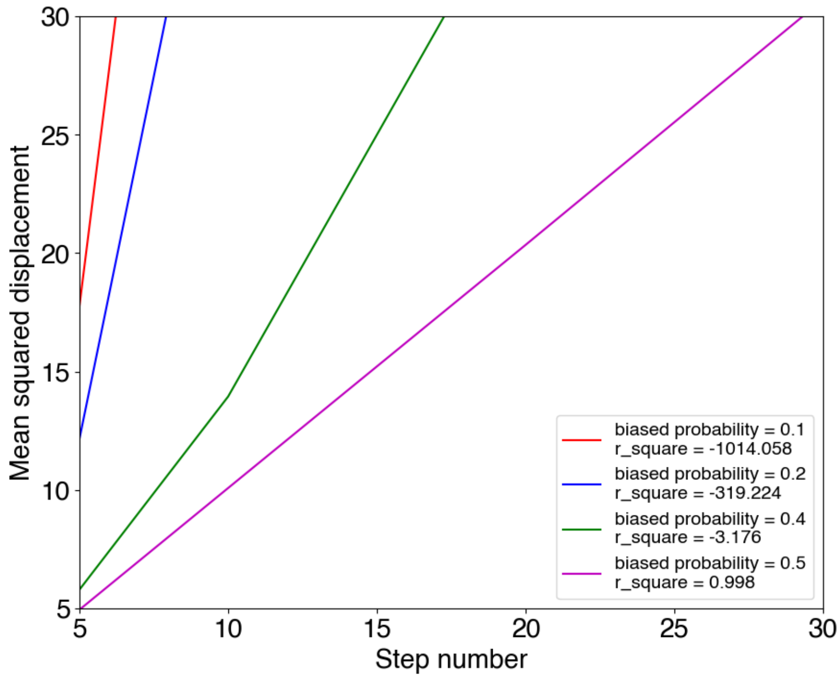

Figure 4: Mean squared displacement v/s step number averaged over multiple simulations. Also see the associated r_sqaure value in plot. Biased probability = 0.5 implies unbiased random walk, thus is linear.

4 Implement a 2D random walk with 1000 jumps. Show an example of particle trajectory path

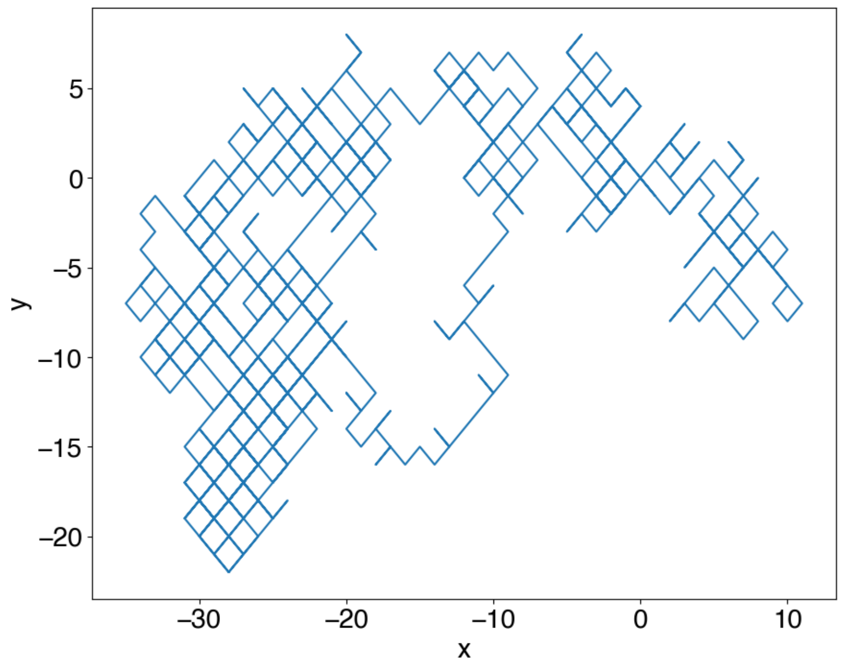

Figure 5: Trajectory of 2D random walk with 1000 jumps. Plot: y v/s x.

5 Assume a 10 by 10 square grid with periodic conditions. One drunk sailor is initially placed in (0,0), and a second one in (5,5). By performing multiple simulations, determine the average number of steps after which the two sailors bump into each other.

Algorithm 1 simulate_collision2d

```
function SIMULATE_COLLISION2D(num_jumps, seed, grid)
    Initialize parameters:
    - sailor1_traj ← [0, 0]
    - sailor2_traj ← [5, 5]
    for step ← 0 to num_jumps − 1 do
        sailor1_traj ← (sailor1_traj + [random(-1, 1), random(-1, 1)]) mod grid
        sailor2_traj ← (sailor2_traj + [random(-1, 1), random(-1, 1)]) mod grid
        if sailor1_traj = sailor2_traj then
            return step + 1
        end if
    end for
    return num_jumps
end function
```

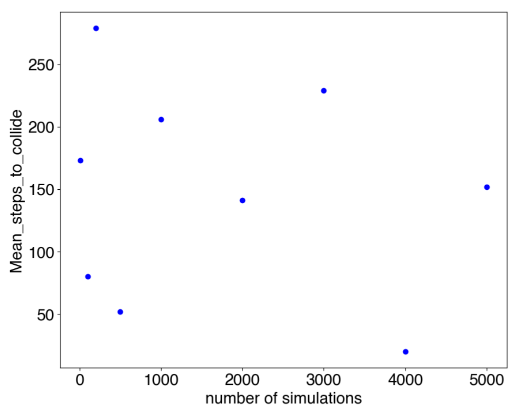

Figure 6: Showing number of mean steps to collide when total jumps are 1000. Value's on Y-axis represent mean across all the simulations.

6 Use Park–Miller random number generator to generate a series of 200 random number with initial seed=71, a=18, m=167. Find the period of the obtained series. Find the minimum value of 'a' for which the period is maximum.

6.1 Period

Period of obtained series: 83 (see red dashed vetrical line in figure 7)

Period of a random number generator is the number of steps after which the random numbers start to repeat. I solved it by checking if the new random number generated is present in the set of random number that have been generated till now.

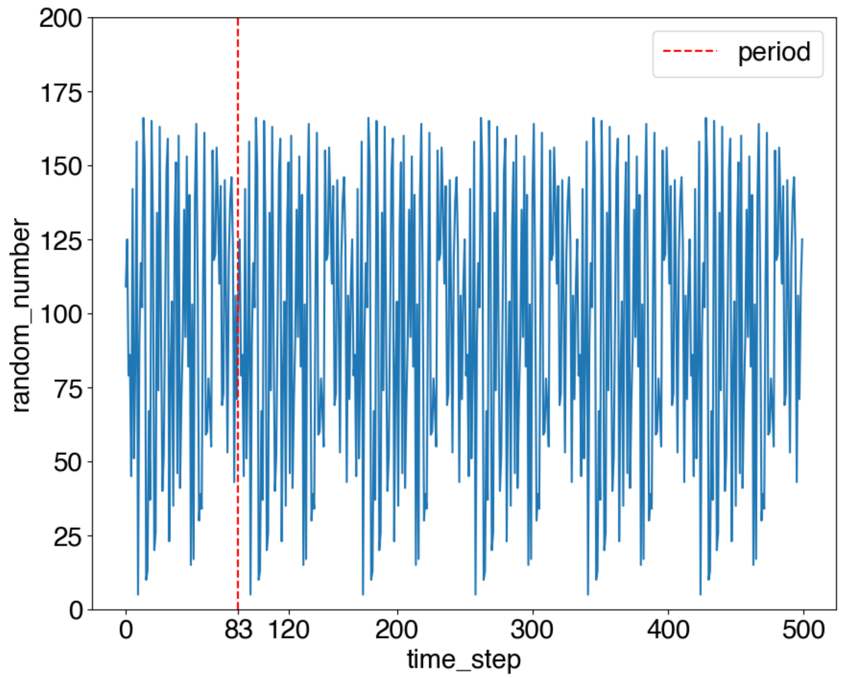

Figure 7: Identifying the period of random number generator

6.2 Minimum "a" for max period

Minimum value of "a" for which period is maximum: 71 (see red dashed vertical line in figure 8)

I repeated the Q6 part "1" with different values of "a" and then plotted it to find the solution.

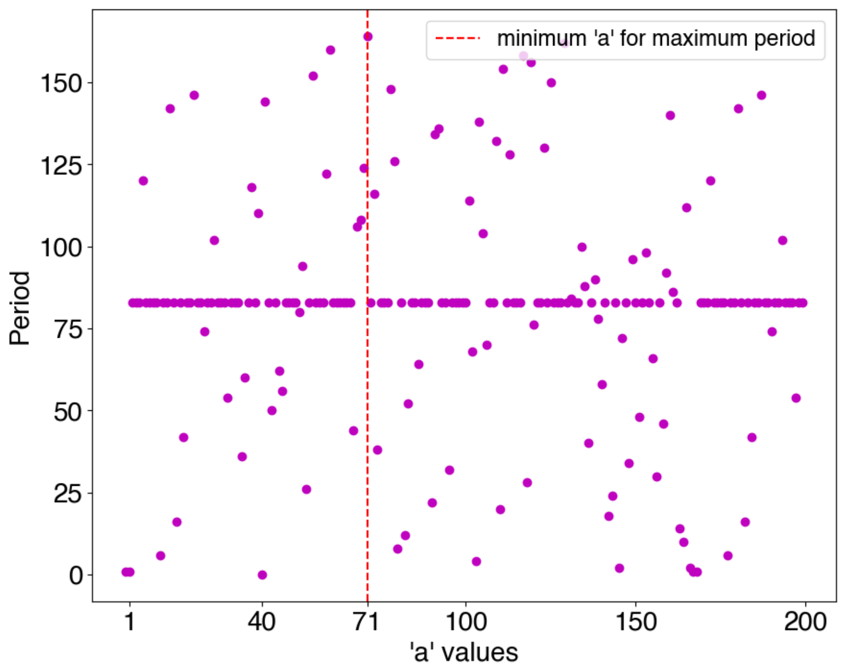

Figure 8: Finding out minimum "a" for which period is maximum