National Technical University of Athens
MSc - Data Science and Machine Learning

# Data Driven Models
# Assignment 0

MSc student
Vasileios Depastas
A.M: 03400131
vasileiosdepastas@mail.ntua.gr

March 2023

## 1 Theoretical part

In this exercise, we examine a famous Markov chain, a random walk on the number line. A simple example of such a random walk is the following:

One is supposed to be at position $x_0 = 0$ at time $t = 0$ and every 1 second, this person moves 1 unit to the left with probability $p = 0.5$, or 1 unit to the right with probability $p = 0.5$. Therefore $x_1 = 1$ or $x_1 = -1$. In a similar manner, we find that $x_2 \in \{-2, -1, 0, 1, 2\}$, $x_2 \in \{-3, -2, -1, 0, 1, 2, 3\}$ and so forth. It's evident already that the possible range of values $x_t$ can take increases as $t$ increases.

Each random walk is a stochastic process for which we can denote the position of a person after $n$ steps and at the corresponding time $t_n$ with $X(t_n)$. Then:

$$X(t_n) = X(t_0) + \sum_{i=1}^{n} Y_i,$$

where $Y_i$ are random variables that are independent and identically distributed (i.i.d.) that follow the same distribution as a random variable $Y$. This random variable(s) can take two values, $Y = -1$ or $Y = 1$, with probability $p = P(Y = -1)$ and given that $P(Y = -1) + P(Y = 1) = 1$, it comes that $P(Y = 1) = 1 - p$. In the simplest scenario, which is the case in this assignment, $p = 0.5$ and therefore $p = P(Y = -1) = P(Y = 1) = 0.5$.

We then proceed to calculate the expected value and variance of the random variable $X(t_n)$.

$$\begin{aligned}
E\left[X(t_n)\right] &= E\left[X(t_0) + \sum_{i=1}^{n} Y_i\right] \\
&= E\left[X(t_0)\right] + E\left[\sum_{i=1}^{n} Y_i\right] \\
&= X(t_0) + \sum_{i=1}^{n} E\left[Y_i\right] \\
&= X(t_0) + \sum_{i=1}^{n} E\left[Y\right] \\
&= X(t_0) + n \cdot E[Y] \\
&= X(t_0) + n \cdot \cancelto{0}{\left[0.5 \cdot (-1) + 0.5 \cdot 1\right]} \\
&= X(t_0),
\end{aligned}$$

where we used the fact that $Y_i$ are i.i.d. We also found here that $E\left[Y\right] = 0$. For the given problem, it is given that $X(t_0) = 0$, and hence $E\left[X(t_n)\right] = X(t_0) = 0$.

In a similar manner, we calculate the variance:

$$Var\left[X(t_n)\right] = E\left[X^2(t_n)\right] - E^2\cancelto{0}{\left[X(t_n)\right]} = E\left[X^2(t_n)\right], \text{ where:}$$

$$X^2(t_n) = \left(\cancelto{0}{X(t_0)} + \sum_{i=1}^{n} Y_i\right)^2 = \sum_{i=1}^{n} Y_i \cdot \sum_{i=1}^{n} Y_i = \sum_{i=1}^{n}\sum_{j=1}^{n} Y_i Y_j, \text{ therefore:}$$

$$\begin{aligned} Var\left[X(t_n)\right] &= E\left[\sum_{i=1}^{n}\sum_{j=1}^{n} Y_i Y_j\right] \\ &= \sum_{i=1}^{n}\sum_{j=1}^{n} E\left[Y_i Y_j\right]. \end{aligned}$$

However, given that $Y_i$ are i.i.d, then for $i \neq j$, it follows that

$$E\left[Y_i \cdot Y_j\right] = \cancelto{0}{E\left[Y_i\right]} \cdot \cancelto{0}{E\left[Y_j\right]} = 0.$$

Based on that result, the previous sum becomes the following:

$$\begin{aligned} Var\left[X(t_n)\right] &= \sum_{i=1}^{n}\sum_{j=1}^{n} E\left[Y_i Y_j\right] \\ &= \sum_{i=1}^{n} E\left[Y_i^2\right]. \end{aligned}$$

Now, we have that $Y = \pm 1$, which means that $Y^2 = 1$ and $E\left[Y^2\right] = E\left[1\right] = 1$. Therefore:

$$\begin{aligned} Var\left[X(t_n)\right] &= \sum_{i=1}^{n} E\left[Y_i^2\right] \\ &= \sum_{i=1}^{n} 1 \\ &= n. \end{aligned}$$

Summarizing the previous results:

$$E\left[X(t_n)\right] = 0,$$

$$Var\left[X(t_n)\right] = n.$$

For a wide-sense stationary process, the variance needs to be constant with respect to time. Here, this is not true, given that the process is dependent on the step number $n$ which defines in turn the time $t$. Therefore, the process cannot be ergodic, which is a subset of a wide-sense stationary process. That means that $X(t_n)$ and $X(t_{n+k})$ are not independent as $k \to \infty$.

## 2 Simulation

We now proceed to validating the previously stated theoretical results with the help of a simulation performed in Python.

We generate a sample of 10,000 random walk instances using $n = 1,000$ steps for each, and for each one of those walks/stochastic processes we retain all the state values for every step (from step 0 to step $n = 1,000$), hence we retain all values $X(t_i)$ for $i \in 0, 1, ..., n$ for each random walk.

Firstly, we plot the 3 first random walk values over n for demonstration purposes. As we can see they can vary a lot.

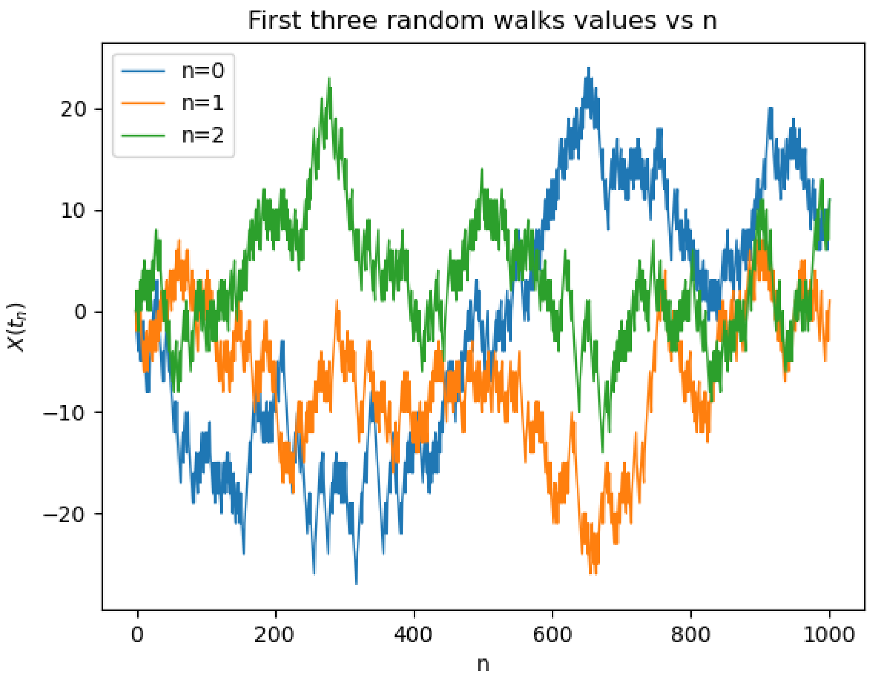

Figure 1: random walks $X(t_i)$, $i \in 0, 1, ..., n$ for the three first random walks of the sample.

Then, we calculate the mean value and variance out of all the instances' values for a particular $n$. We expect the mean value to be 0 regardless of $n$, and variance to present a linear relationship to $n$.

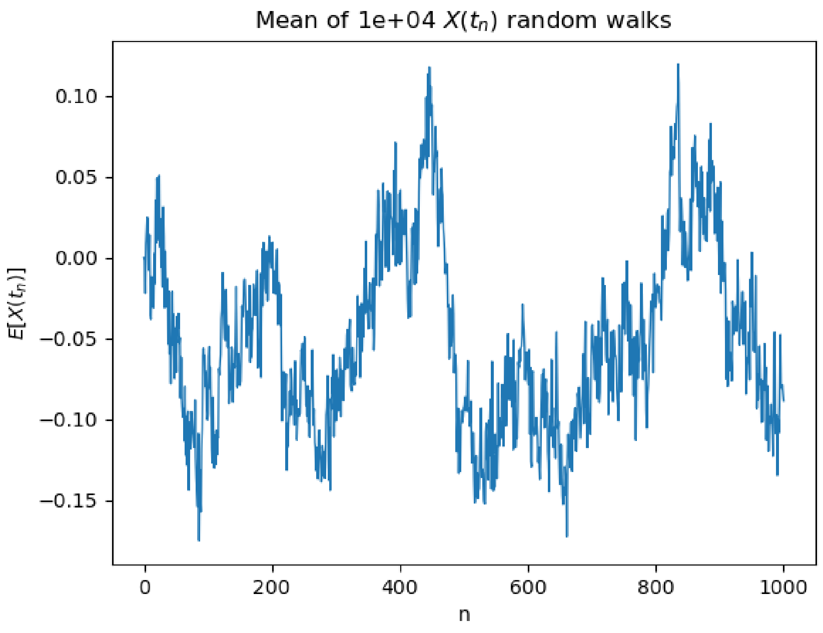

Figure 2: Mean of random walks sample for each $n$

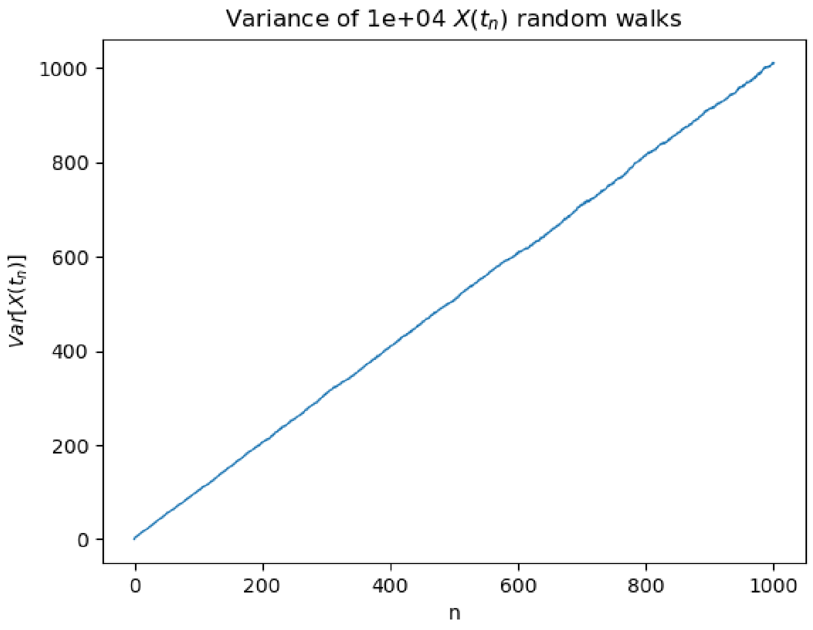

Figure 3: Variance of random walks sample for each $n$

As we can see in the previous figures, the variance for a sample of size 10,000 goes resembles very closely the relationship $Var\left[X(t_n)\right] = n$, and at the same time the mean $E\left[X(t_n)\right]$ tends to go close to 0 as expected, however, it would be more clearly visible with a larger sample.

Below follows the code that generated the simulation figures:

```python
import random

import numpy as np
import matplotlib.pyplot as plt

# Constants
X_t0 = 0
N = 1000
SAMPLE_SIZE = 10000


def coin_flip() -> float:
    prob = random.uniform(0, 1)  # between 0 and 1 uniformly
    return prob


def X_tn_range(n: int) -> np.array:
    coin_flips = [coin_flip() for i in range(n)]  # perform n coin flips
    Y_values = [1 if prob <= 0.5 else -1 for prob in coin_flips]

    current_sum = X_t0
    X_cum_values = np.array(current_sum)  # first element is X_t0
    for y in Y_values:
        current_sum += y
        X_cum_values = np.append(X_cum_values, current_sum + y)

    return X_cum_values


def sample_of_X_tn(n: int, sample_size: int) -> np.array:
    all_series = X_tn_range(n)  # first random walk instance
    for i in range(sample_size-1):
        X_tn_series = X_tn_range(n)
        all_series = np.vstack((all_series, X_tn_series))
    return all_series


def get_sample_mean(A: np.array) -> np.array:
    return A.mean(axis=0)  # mean for each timestep across the different sample instances


def get_sample_var(A: np.array) -> np.array:
    return A.var(axis=0)  # variance
```

```
def plot_random_walks(x: int, n_range: int, A: np.array) -> None:
    plt.figure()

    for i in range(x):
        X_tn_series = A[i]
        plt.plot(n_range, X_tn_series, label=f'n={i}', linewidth=1)

    plt.title('First three random walks values vs n')
    plt.xlabel('n')
    plt.ylabel('$X(t_n)$')
    plt.legend()
    plt.savefig('First-three-random-walks.png')


def plot_single_line(n_range: int, A: np.array, title: str, ylabel: str, filename: str) ->
    plt.figure()
    plt.plot(n_range, A, linewidth=1)
    plt.title(f'{title}')
    plt.xlabel('n')
    plt.ylabel(ylabel)
    plt.savefig(f'{filename}.png')


if __name__ == '__main__':
    n_range = np.arange(0, N+1)  # all 0 to 1000 number in a numpy array

    sample: np.array = sample_of_X_tn(N, SAMPLE_SIZE)
    sample_mean: np.array = get_sample_mean(sample)
    sample_var: np.array = get_sample_var(sample)

    plot_random_walks(3, n_range, sample)
    plot_single_line(n_range,
                     sample_mean,
                     title=f'Mean of {int(SAMPLE_SIZE):.0e} $X(t_n)$ random walks',
                     ylabel='$E[X(t_n)]$',
                     filename='Mean')
    plot_single_line(n_range,
                     sample_var,
                     title=f'Variance of {int(SAMPLE_SIZE):.0e} $X(t_n)$ random walks',
                     ylabel='$Var[X(t_n)]$',
                     filename='Variance')
```