Assignment 1: Galton Board

Miguel Alcañiz Moya

September 26, 2024

1 Statement of the assignment

In this programming assignment, you will have to write a program to simulate a Galton board, also known as a Galton box, quincunx, or bean machine. This device is used to illustrate the central limit theorem, particularly that with a sufficiently large sample, the binomial distribution is approximated by the normal distribution.

The device consists of a vertical board with interleaved rows of pegs. A large number of small balls fall from the top, bouncing left and right as they hit the pegs on their way to the bottom. The bottom of the board collects balls, which will follow a binomial distribution (see Figure 1).

Figure 1: A Galton board

We can simulate the Galton board using a virtual triangular matrix. Balls "drop" from cell $(0, 0)$ down to n levels. At each step of the simulation, a ball in some cell (i, j) moves to the "left" (cell $(i + 1, j)$) with probability 1/2, or to the "right" (cell $(i, j + 1)$) with probability 1/2. After n steps, the ball will be in some cell $(i, n - i)$, because at each of the n steps, we increase either the row or the column by 1, and the sum of row and column must be n.

The probability $p_{i,n}$ that a ball starting at $(0, 0)$ ends at cell $(i, n-i)$ is given by a binomial distribution (why?). If we drop N balls, we expect $N \cdot p_{i,n}$ balls to land in cell $(i, n - i)$. The standard deviation is $\sqrt{N p_{i,n} q_{i,n}}$, where $q_{i,n} = 1 - p_{i,n}$.

Write a program to carry out the experiment and study the match between the experimental data and the predictions of probability theory. Study the effect of large boards (increase the value of n) and a large number of experiments (a larger number N of balls). Also, check the agreement between the binomial distribution and the normal distribution $N(\mu, \sigma^2)$. Notice that as $p_{i,n} \to 0$ with increasing n, the approximation between the binomial $Bin(n, 1/2)$ and the normal distribution $N(n/2, n/4)$ improves. This becomes more visible when we draw a larger number N of samples from the $Bin(n, 1/2)$ distribution.

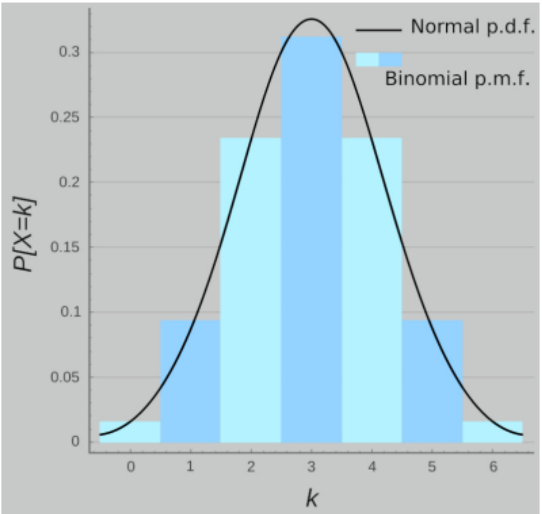


Figure 2: Binomial vs normal distributions

Use graphical plots to illustrate the outcomes of your experiments. Quantifying (and plotting/tabulating) the error between empirical results and theoretical predictions is important, especially in terms of how the error evolves as we vary the relevant parameters, namely n and N. For example, do not simply produce graphs similar to Figure 2; instead, compute the mean quadratic error between the probability density function (PDF) of the normal distribution (the solid black line) and the probability mass function (PMF) of the binomial distribution (heights of the blue rectangles) at the integer points $k = 0, k = 1, \ldots$.

2 Code

In this section I am going to comment the code I made with python. I used python because it has a nice library to plot called *matplotlib.pyplot*.

The code is uploaded in the GitHub: $@github.com : miguelalcaniz02/Galton\_Box\_Alcaniz\_Miguel.git$ In the GitHub project there will be this pdf file, the python code file, the license and the ReadMe with the instructions to execute the file.

The code is commented so I am just going to add it, I think it is quite simple to understand.

```
import math
import random
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import norm
from matplotlib.ticker import FuncFormatter

# Ask the user for the size of the Galton board and the number of steps (iterations)
```

```python
n = int(input("Introduce the size of the Galton Board: "))
N = int(input("Introduce the number of steps: "))

# Initialize the board with n positions, all set to 0
board = [0] * (n+1)

# Simulate the Galton board
for i in range(N):
    position = 0
    for j in range(n):
        position += random.choice([0, 1])  # Randomly add 0 or 1 to each position
    board[position] += 1

# Normalize the board by dividing by the total number of balls
board = [x/ N for x in board]

# For the plot
plt.title(f"Galton Board Simulation for n = {n:.0f} and N = {N:.0f}.")
plt.xlabel('Position')
plt.ylabel('Quantity of balls')

# Plot the results of the binomial as a bar chart
plt.bar(range(n+1), board, label = 'Binomial distribution')

# Generate a range of x values from 0 to n
x = np.linspace(0, n, N)

# Compute the normal distribution's PDF for these x values
y = norm.pdf(x, n/2, math.sqrt(n/4))

# Create the plot of the normal distribution
plt.plot(x, y, color='red', label = 'Normal distribution')

# Define a to_percent function for the y label
def to_percent(y, position):
    # Multiply by 100 and add a percent sign
    return f'{100 * y:.0f}%'
plt.gca().yaxis.set_major_formatter(FuncFormatter(to_percent))

# Calculate the Mean Quatratic mean
zx = np.linspace(0, n, n+1)
zy = norm.pdf(zx, n/2, math.sqrt(n/4))
MQE = np.sum(np.square(zy-board))/2

# Calculate the mean and the variance of the generated binomial distribution
positions = np.arange(n+1)
mean = np.sum(positions * board)
variance = np.sum(board * (positions - mean)**2)

# Add text below the plot
plt.subplots_adjust(bottom=0.50)  # Increase bottom margin to 50% of the figure height
plt.figtext(0.05, 0.35, f"The mean quatratic error is {MQE:.6f}.", fontsize=12)
plt.figtext(0.05, 0.29, f"The mean of the normal distribution is {n/2:.6f}.", fontsize=11)
plt.figtext(0.05, 0.24, f"The mean of the binomial distribution is {mean:.6f}.",
```

```
fontsize=11)
plt.figtext(0.05, 0.19, f"(The difference between both means is {abs(mean-n/2):.6f}.)",
fontsize=11)
plt.figtext(0.05, 0.14, f"The variance of the normal distribution is {n/4:.6f}.",
fontsize=11)
plt.figtext(0.05, 0.09, f"The variance of the binomial distribution is {variance:.6f}.",
fontsize=11)
plt.figtext(0.05, 0.04, f"(The difference between both variances is {abs(n/4-variance):.6f}.)",
fontsize=11)

# Plot the legend and show the plot
plt.legend()
plt.show()
```

3 Examples

In this section I am going to show some examples of the usage of the code. When you execute the code for a given n and N you get a bar plot of the discrete binomial distribution of mean $n/2$ and values from 0 to n with N tries. The plot shows the percentage of the results for every value. There is also drawn a normal distribution that should behave like the bars that represent the binomial.

In the first example the plot correspond to the values $n = 10$ and $N = 500$.

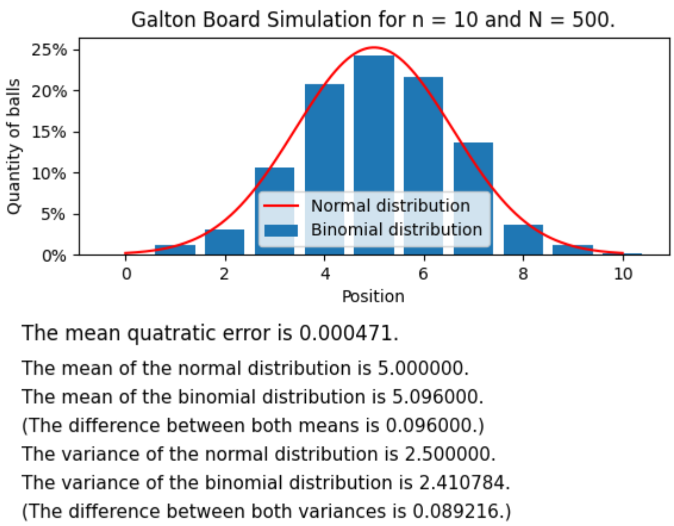
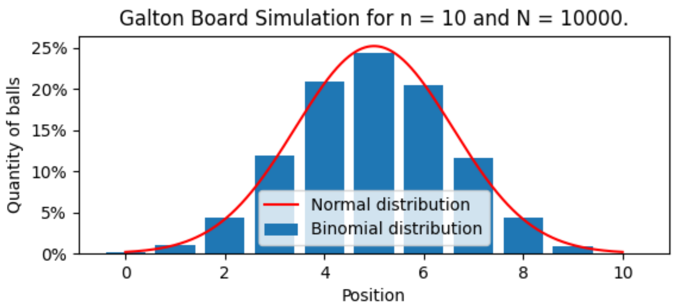


Figure 3: Example 1

We can see as in the example above the distribution created by this random process called binomial distribution follows really well the behaviour of the normal distribution. We can also appreciate that the mean quadratic error is really close to 0. An also that the practical mean and variance are almost equal to the theoretical one.

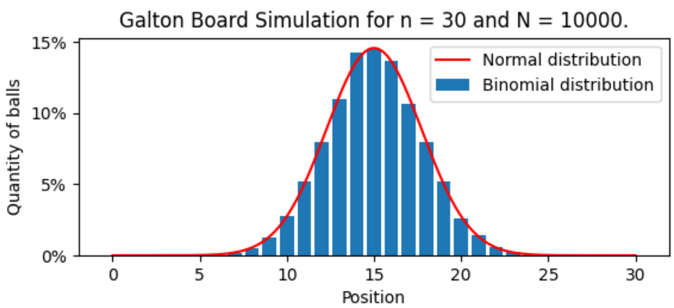


The mean quatratic error is 0.000053.

The mean of the normal distribution is 5.000000.
The mean of the binomial distribution is 4.983100.
(The difference between both means is 0.016900.)
The variance of the normal distribution is 2.500000.
The variance of the binomial distribution is 2.473814.
(The difference between both variances is 0.026186.)

Figure 4: Example 2

In this second example we can see how getting more samples *of the balls falling from the top* makes the binomial more similar to the normal distribution, we can see that because the mean quadratic error gets smaller (as we could expect), as well as the precision on the mean and the variance.



The mean quatratic error is 0.000042.

The mean of the normal distribution is 15.000000.
The mean of the binomial distribution is 14.999100.
(The difference between both means is 0.000900.)
The variance of the normal distribution is 7.500000.
The variance of the binomial distribution is 7.436899.
(The difference between both variances is 0.063101.)

Figure 5: Example 3

In this third example we took the same N and a bigger n than example 2. We can see how taking a bigger n also gets better results, based again on the mean quadratic error, and the similarity to the theoretical mean and variance.