# Reporte 1:
# Movimiento Browniano

1642654 Jorge Alejandro Torres Quintanilla

13 de febrero de 2022

## 1. Objetivo

Esta práctica tiene como objetivo el estudiar de manera sistemática el movimiento aleatorio de una partícula en un espacio que va de 1 a 5 dimensiones. Se varía también la cantidad de pasos que dura la caminata, y para cada caminata en cada dimensión, el experimento se repite 30 veces. Los resultados se observan un una sola gráfica con diagramas caja-bigote. Asimismo, se evalúa el tiempo promedio en que se realiza el cómputo de una réplica del experimento.

## 2. Desarrollo

En mi repositorio de GitHub se puede observar la evolución del código desarrollado. Tomando como base el código proporcionado por la Dra. Elisa Schaeffer [1], se hace una modificación para iterar entre la cantidad de pasos que se realizan en cada dimensión. A continuación el código utilizado para la práctica.

```
from random import random, randint
from math import sqrt
import matplotlib.pyplot as plt
import numpy as np
from time import time

runs = 30 #replicas
caminatas = [100, 1000, 10000] #pasos
results = [] #almacena las dimensiones

for i in range(3): #itera la cantidad de pasos
    dur = caminatas[i]
    for dim in range(1, 6): #de una a cinco dimensiones
        mayores = []
        for rep in range(runs):#corre el experimento 30 veces en cada dimension
            before = time()*1000
            pos = [0] * dim
            mayor = 0
            for paso in range(dur):
                eje = randint(0, dim - 1)
                if pos[eje] > -100 and pos[eje] < 100:
                    if random() < 0.5:
                        pos[eje] += 1
                    else:
                        pos[eje] -= 1
                else:
                    if pos[eje] == -100:
                        pos[eje] += 1
                    if pos[eje] == 100:
                        pos[eje] -= 1
```

```
                mayor = max(mayor, sqrt(sum([p**2 for p in pos])))
                after = time()*1000
            mayores.append(mayor)
        results.append(mayores)
tiempo = after - before
print(tiempo)

#separar los resultados en tres grupos de caminatas
walks_1 = results[0:5] #caminatas de 100 pasos
walks_2 = results[5:10] #caminatas de 1000 pasos
walks_3 = results[10:15] #caminatas de 10000 pasos

#empezar a graficar

ticks = ['1', '2', '3', '4', '5']

#funcion definir los colores de cajas y bigotes
def set_box_color(bp, color):
    plt.setp(bp['boxes'], color=color)
    plt.setp(bp['whiskers'], color=color)
    plt.setp(bp['caps'], color=color)
    plt.setp(bp['medians'], color='orange')

plt.figure()

bpl = plt.boxplot(walks_1, positions=np.array(range(len(walks_1)))*5.0-1.0, \
                  sym='', widths=0.8)
bpc = plt.boxplot(walks_2, positions=np.array(range(len(walks_2)))*5.0, \
                  sym='', widths=0.8)
bpr = plt.boxplot(walks_3, positions=np.array(range(len(walks_3)))*5.0+1.0, \
                  sym='', widths=0.8)
set_box_color(bpl, '#D7191C')
set_box_color(bpc, '#2CB62C')
set_box_color(bpr, '#2C7BB6')

#leyenda
plt.plot([], c='#D7191C', label='100 pasos')
plt.plot([], c='#2CB62C', label='1000 pasos')
plt.plot([], c='#2C7BB6', label='10000 pasos')
plt.legend()

plt.xticks(range(0, len(ticks)*5, 5), ticks)
plt.xlim(-2, len(ticks)*5)
plt.title('Distancia Euclideana')
plt.xlabel('Dimensiones')
plt.ylabel('Distancia Maxima')
plt.tight_layout()
plt.savefig('DistanciaEucl.png')
plt.show()
```

Como se observa, primero se realizan 30 ciclos de 100 pasos en una a cinco dimensiones, después con 1,000 pasos y por último con 10,000 pasos. En cada ciclo se calcula la distancia máxima euclideana tomada desde el punto de origen utilizando la ecuación (1) y se agrega a la lista de resultados para ser graficados posteriormente.

$$d_{max} = \sqrt{\sum_{n=1}^{m} x_n^2} \tag{1}$$

Donde $m$ es la cantidad de ejes.

Adicionalmente, se utiliza el comando `time()` para medir el tiempo que toma realizar los 30 ciclos en cada dimensión, a lo cual se realiza una normalización para conocer el promedio que tarda en ejecutarse un solo ciclo.

# 3. Resultados

## 3.1. Distancia Máxima Euclideana

En la figura 1 se puede observar la agrupación estadística de las distancias máximas recorridas por una partícula con movimiento aleatorio al realizar caminatas de 100 (rojo), 1,000 (verde) y 10,000 (azul) pasos, para espacios cartesianos con 1 a 5 ejes. Cabe resaltar que estos ejes tienen un rango de -100 a 100 unidades, y que la partícula retrocede un paso al llegar al borde.

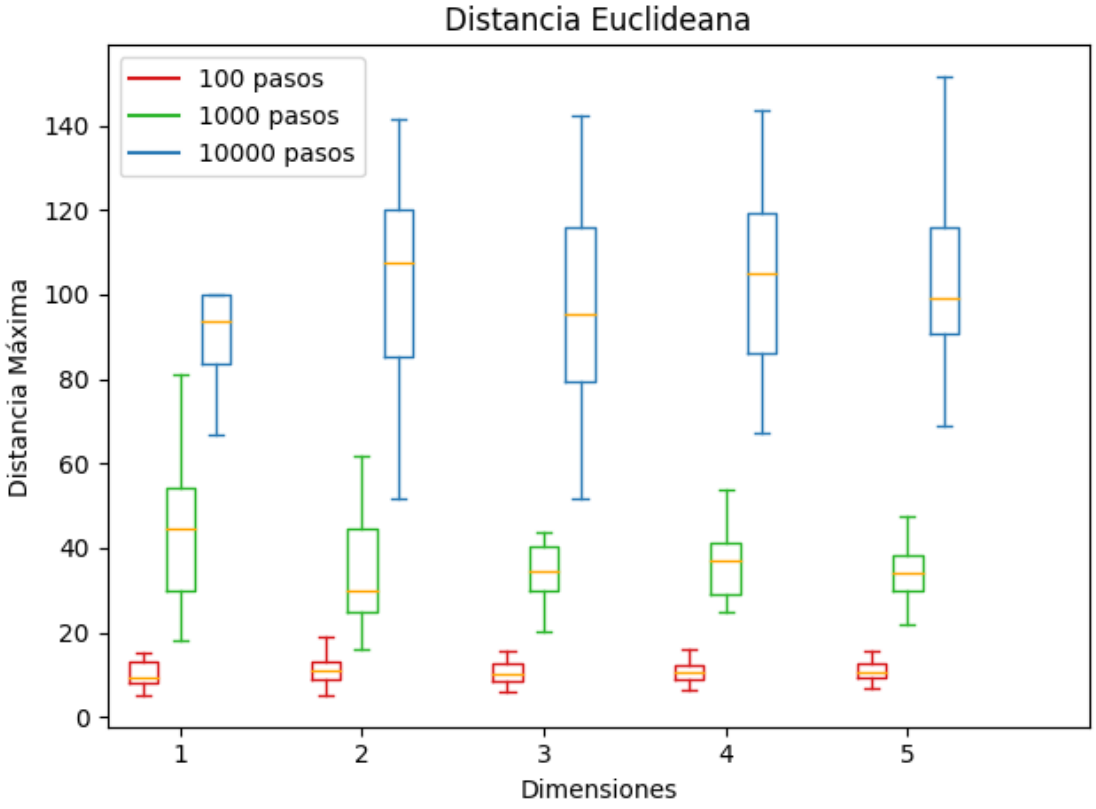


Figura 1: Diagrama caja-bigote donde se observan las estadísticas de distancia máxima euclideana para los grupos de 100, 1,000 y 10,000 pasos en dimensiones de 1 a 5.

Cuadro 1: Tiempo promedio que tarda una repetición de 30 ciclos en realizarse.

| **Repetición** | **Tiempo (ms)** |
|---|---|
| 1 | 31.2421875 |
| 2 | 31.278564453125 |
| 3 | 37.772216796875 |
| 4 | 28.126953125 |
| 5 | 31.248291015625 |
| 6 | 31.731201171875 |
| 7 | 31.338134765625 |
| 8 | 31.368896484375 |
| 9 | 31.35498046875 |
| 10 | 33.336669921875 |
| **Promedio** | 31.8798085703125 |

### 3.2. Tiempo de Ejecución

Al realizar un estudio automatizado del tiempo que tarda una repetición en ejecutarse y calcular un promedio al correr el programa varias veces, se puede crear una idea aproximada de lo eficiente que es el código para la tarea en cuestión. Los resultados de tal estudio se muestran en el cuadro 1 y se puede observar que mi computadora tarda $\approx 32$ ms en completar una repetición de 30 ciclos del experimento, por lo que se estima que tardaría $\approx 1.06$ ms en completar un ciclo.

## 4. Conclusión

Al observar los diagramas de la figura 1 se puede apreciar cómo el rango de distancia máxima es cada vez más amplio al aumentar la cantidad de pasos, lo cual se debe a que tiene más posibilidades de terminar en puntos diferentes del espacio. Por otro lado, el rango de distancias no aumenta o disminuye significativamente al variar la cantidad de ejes en que se puede mover la partícula.

Tomando en cuenta que el tiempo promedio de ejecución de una repetición es de 32 ms y que se realizan 15 repeticiones (5 grupos de dimensiones por 3 grupos de largo de caminata), se tiene que el programa debería tardar $\approx 480$ ms en ejecutarse. La realidad es que tarda más debido a que debe realizar y guardar la gráfica en un archivo PNG. Además, existe la posibilidad de paralelizar operaciones al reservar núcleos del CPU para ejecutar el programa, lo cual no se ha implementado en esta ocasión.

## Referencias

[1] Dra. Elisa Schaeffer. Brownianmotion. *Repositorio, GitHub*, 2019. URL https://github.com/satuelisa/Simulation/tree/master/BrownianMotion.