Modelos Estocásticos - T05: Raíces Complejas

Jorge Aurelio Morales Manrique
C.C. 1010075711
jomorales@unal.edu.co

Universidad Nacional de Colombia
Marzo 10 de 2021

Manual de usuario

En la carpeta adjunta con el nombre "Código Fuente", se encuentra un archivo denominado **main.py** el cuál contiene el código de la simulación realizada con el lenguaje de programación Python. El programa solicita como entrada el valor de β y n los cuales representan el valor sobre el cual se define el intervalo y el número de simulaciones respectivamente. A continuación se lista la serie de pasos necesarios para ejecutar el programa en ambientes Windows, Linux:

1. Descargar e instalar Python 3.x accediendo a la página oficial https://www.python.org/downloads/
2. Descargar y ubicar el archivo con el código fuente en un directorio (el nombre del directorio no afecta)
3. Abrir una consola de comandos. En Windows realizar la combinación **(Windows + R)**, en el campo de texto emergente digitar cmd y posteriormente dar click en aceptar

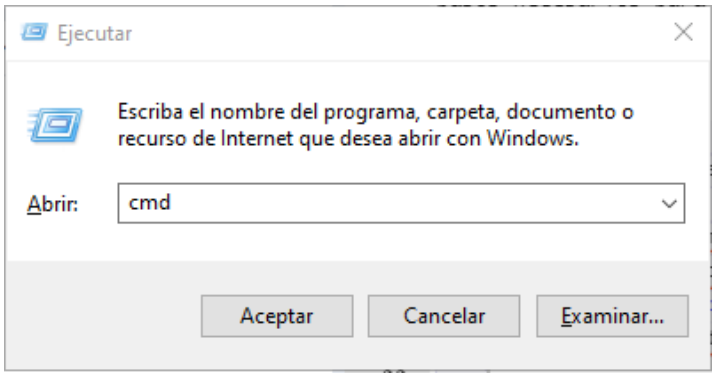

Figura 1: Proceso para abrir una consola de comandos en el sistema operativo Windows.

4. Una vez en la consola se debe cambiar al directorio que contiene el archivo con el código fuente. Tanto en Windows como en Linux se realiza el siguiente comando **(cd path)** donde **path** es la ruta a la cual se quiere cambiar.

5. Posteriormente digitar el siguiente comando

```
python main.py
```

6. Ingresar los valores de β y n.

Una vez se ingresan los datos el programa comenzará la simulación imprimiendo al final el valor obtenido para $P(A)$ para el β especificado.

Manual técnico

La estructura interna del programa se divide en dos componentes principales los cuales se describen a continuación

- **Función "simulate".** Recibe como parámetro el valor de β, genera dos números aleatorios con la función **uniform** de la librería **random**, la cual recibe como parámetros los límites inferior y superior del intervalo.

```
4   def simulate(beta):
5       '''
6       :param beta: valor de ß
7       :return: verdadero si con los valores de b y c
8       la ecuación tiene raíces reales
9       '''
10      b = random.uniform(-beta, beta)
11      c = random.uniform(-beta, beta)
12      if b**2 >= c:
13          return 1
14      return 0
```

Figura 2: Función "simulate".

- **Simulación.** Realiza n simulaciones y retorna la probabilidad obtenida.

```
30      beta = float(input('Ingrese el valor de beta: '))
31      n = int(input('Ingrese el número de simulaciones: '))
32      successes = 0
33      for i in range(n):
34          if simulate(beta=beta) == 1:
35              successes = successes + 1
36      probability = successes / n
37      print(f'P(A) = {probability}')
```

Figura 3: Secuencia de instrucciones para realizar la siulación.