Proyecto
Valuación de derivados

Cebada Santana, Cesar

1 Problema 1

Este código implementa una simulación de un Movimiento Browniano Geométrico y calcula el precio de una opción europea utilizando el método de Monte Carlo. A continuación, se describe lo que se hizo en el código:

1. Se importan las bibliotecas necesarias, como matplotlib.pyplot, numpy, math y pandas, para realizar operaciones matemáticas, visualizar gráficos y trabajar con datos estructurados.

2. Se define la función calculoDeOpcion con los siguientes parámetros de entrada:
Número de simulaciones (M).
Precio inicial del activo subyacente ($precio\_inicial$).
Volatilidad del subyacente (volatilidad).
Cambios en el tiempo (dt).
Plazo de vencimiento (T)
Tipo de opción digital (”c” para una opción call, ”p” para una opción put) (o).
Precio de ejercicio de la opción (strike).
Tasa libre de riesgo ($tasa\_libre\_riesgo$).

3.Se inicializan las listas $payoffs\_opcion\_europea$ y $paths$ para almacenar los payoffs de la opción y las trayectorias simuladas respectivamente.

4. Se realiza un bucle for para generar M simulaciones del Movimiento Browniano Geométrico. En cada simulación, se actualiza el precio del subyacente utilizando la fórmula del movimiento browniano y se almacenan los precios en la lista precios.

5. Dependiendo del tipo de opción (o), se calcula el payoff de la opción en el último precio de cada trayectoria simulada y se ajusta el payoff con el factor de descuento (1 + tasa_libre_riesgo)**T. Los payoffs se agregan a la lista payoffs_opcion_europea.

6. Se utiliza la biblioteca matplotlib.pyplot para graficar las trayectorias simuladas en el subyacente.

7. Se calcula el precio del derivado como el promedio de los payoffs de la opción y se muestra en la salida.

Estos son los resultados que obtuvimos:

	Opcion tipo call	Opcion tipo put
1000 simulaciones con Monte Carlo	10.504494473786751	5.87482890219224
10000 simulaciones con Monte Carlo	10.626101081490418	5.646991338034845
100000 simulaciones con Monte Carlo	10.472683047405093	5.594273684466119
1000000 simulaciones con Monte Carlo	10.455416594894659	5.582514897528591
Black Scholes	10.450583572185565	5.573526022256971

2 Problema 2

Este código implementa una función llamada arbolBinomial que calcula el valor de una opción europea (tanto call como put) utilizando el método del árbol binomial. A continuación, se describe lo que se hizo en el código:

1. Se importa la biblioteca numpy con la declaración import numpy as np, lo que permite utilizar funciones matemáticas y arreglos multidimensionales eficientes.

2. Se define la función arbolBinomial con los siguientes parámetros de entrada:
N: Número de pasos en el árbol binomial.
precio_inicial: Precio inicial del activo subyacente.

u: Factor de subida en el árbol binomial.
d: Factor de bajada en el árbol binomial.
K: Precio de ejercicio (strike) de la opción.
r: Tasa libre de riesgo de manera anual.
T: Tiempo de maduración en años.
M: Monto que paga la opción al vencimiento.
o: Tipo de opción digital ("c" para una opción call, "p" para una opción put).

3. Se calculan algunas constantes necesarias para el cálculo del árbol binomial, como el paso de tiempo (dt) y la probabilidad neutral al riesgo (q).

4. Se crea una lista llamada subyacente que almacena los valores de los activos subyacentes en cada nodo del árbol binomial.

5. Se calcula el payoff de la opción en los nodos finales del árbol, dependiendo del tipo de opción ("c" o "p"). Para una opción call, se compara el precio del subyacente con el precio de ejercicio (K), mientras que para una opción put se hace la comparación inversa. Si se cumple la condición, el payoff es M, de lo contrario es cero.

6. Se realiza un bucle hacia atrás a través de los nodos del árbol binomial, actualizando los payoffs en cada nodo intermedio utilizando la fórmula del descuento y la combinación de las probabilidades de subida y bajada.

7. Finalmente, se devuelve el valor del payoff en el nodo inicial del árbol, que representa el valor de la opción europea. El resultado que obtuvimos con los datos que se nos dan en el problema es de 21.038398434152327.