學習歷程

1. 透過 Monte Carlo method

a. 對 Hull White Model 模擬 Short Rate

公式：

Hull White Model： $dr_t = (\theta(t) - ar_t)dt + \sigma dW_t$

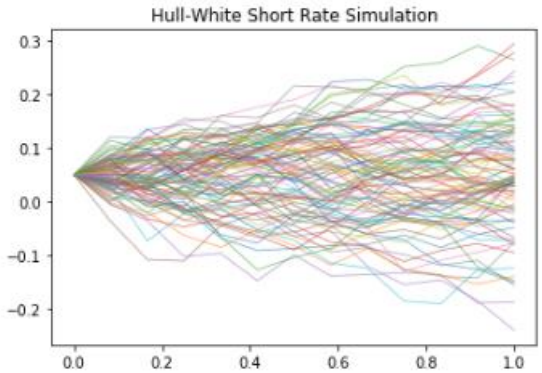
參考並修改 Hull White Term Structure Simulations with QuantLib Python 網站：

http://gouthamanbalaraman.com/blog/hull-white-simulation-quantlib-python.html

```
def generate_paths(num_paths, timestep):
    arr = np.zeros((num_paths, timestep+1))
    for i in range(num_paths):
        sample_path = seq.next()
        path = sample_path.value()
        time = [path.time(j) for j in range(len(path))]
        value = [path[j] for j in range(len(path))]
        arr[i, :] = np.array(value)
    return np.array(time), arr
```

```
time, paths = generate_paths(num_paths, timestep)
for i in range(num_paths):
    plt.plot(time, paths[i, :], lw=0.8, alpha=0.6)
plt.title("Hull-White Short Rate Simulation")
plt.show()
```

利用以上函式畫出Short Rate Simulation：



b. 將 Short Rate 帶入 Geometric Brownian Motion，r 換成 r(t) 模擬

股價

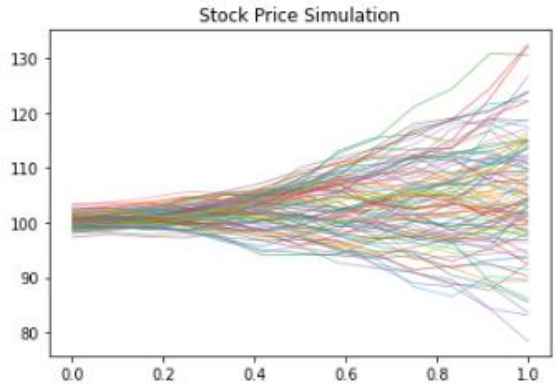
幾何布朗運動參考並修改自老師所提供之範例程式：

```
def genBrownPath(T, mu, sigma, S0, dt):
    S = []
    n = round(T/dt)
    W = [0] + np.random.standard_normal(size = 1)
    W = (W + np.random.standard_normal(size = 1))*np.sqrt(dt)
    for i in range(len(time)):
        S.append(S0*np.exp((float(mu[i])-0.5*sigma**2)*float(time[i]) + sigma*W))
    plt.plot(time, S, lw=0.8, alpha=0.6)
    plt.title("Stock Price Simulation")
    return S
```

將上小題所算出之 r(t) 代入函式：

```
Paths = []
for i in range(num_paths):
    Paths.append(genBrownPath(T, paths[i, :], sigma, S0, dt))
```

得到 Stock Price Simulation：



c. 自訂選擇權履約價，對每一條 path 計算出到期日時的 PayOff

```
p = []
for i in range(num_paths-1):
    Final_price = Paths[i][T]
    p.append(int(Final_price - Strike))
print(p)
```

```
[7, 11, 19, 11, 13, 8, 23, 15, 6, 14, -10, 14, 9, 39, -17, 6, 2, 17, 22, 0, 1, 10, 18, 20, 15, 11, 34, -3, 13, 6, 12, 17, 7, 1,
-13, 4, 15, -10, 3, 12, 28, 7, 20, 6, 3, -2, 6, 19, 18, 25, 1, 9, 7, -4, 29, 1, 16, 22, 21, 22, 16, -8, 2, 10, 11, 12, 9, 16,
2, 9, 15, 7, 33, -2, -2, 10, -2, 8, 12, -2, 9, 5, -10, 9, 19, 13, 3, 24, 7, 16, 8, 19, 19, 35, 14, 24, 13, 7, 12]
```

計算出每一條 path 的 payoff （final price - strike）

2. 計算出 Call Price & Put Price

Call：

```
Pay_off = []
for i in range(num_paths-1):
    Final_price = Paths[i][T]
    if Final_price - Strike > 0:
        X = Final_price - Strike
        Pay_off.append(X + np.exp(-1.0*r*i)*X)
print(np.sum(Pay_off)/num_paths)
```

```
19.49259869862335
```

Put :

```
Pay_off = []
for i in range(num_paths-1):
    Final_price = Paths[i][T]
    if Strike - Final_price > 0:
        X = Strike - Final_price
        Pay_off.append(X + np.exp(-1.0*r*i)*X)
print(np.sum(Pay_off)/num_paths)
```

```
1.3627744276577303
```