Case 2: Black-Scholes formula for Unit-Linked

Asha de Meij - i6254733

November 7, 2024

Consider the unit-linked contract with a maturity guarantee discussed in Week 1. Assume the following parameters:

- $r = 1\%$,
- $\sigma = 15\%$,
- The final guaranteed amount at $T = 10$ equal to 100

Question 1. Derive the closed-form expression for the price at $t = 0$ for the unit-linked contract with a 0% guarantee. Hint: Use the formula for the payoff $F(T, Z) = \max\{e^Z, 100\}$ for the unit-linked contract at time T and evaluate the integral from slide 18 for this $F(T, Z)$. Calculate the integrals in a similar way as the Black-Scholes formula in Joshi, Section 6.8.

From the hint, we know that we should consider:

$$F(T, Z) = \max\{e^Z, 100\} \tag{1}$$

Furthermore, the formula on slide 18 gives us:

$$C(t, e^Z) = e^{-r(T-t)} \int_{-\infty}^{\infty} F(T, Z)\, \phi(t, z; T, Z)\, dZ \tag{2}$$

where

$$\phi(t, z; T, Z) = \frac{\exp\left\{-\frac{1\left(Z-2-\left(--\frac{1}{2}\sigma^2\right)(T-t)\right)^2}{2\sigma^2(T-t)}\right\}}{\sqrt{2+\sigma^2(T-t)}} \tag{3}$$

and $\phi(t, z; T, Z)$ is the pdf of $Z \sim N(z + (r - \frac{\sigma^2}{2})(T - t), \sigma^2(T - t))$.

By substituting eq. (1) into (2), we get:

$$C(t, e^Z) = e^{-r(T-t)} \int_{-\infty}^{\infty} \max\{e^Z, 100\}\, \phi(t, z; T, Z)\, dZ$$

To evaluate this integral, we need to consider the case where $e^Z > 100$ and when $e^Z < 100$. So we need to split up the integral into 2 "regions"; we get the following:

$$C(t, e^Z) = e^{-r(T-t)} \left(\int_{-\infty}^{\ln 100} 100\, \phi(t, z; T, Z)\, dZ + \int_{\ln 100}^{\infty} e^Z\, \phi(t, z; T, Z)\, dZ \right)$$

We first evaluate the 2nd integral:

$$\int_{\ln 100}^{\infty} e^Z \, \phi(t, z; T, Z) \, dZ = \int_{\ln 100}^{\infty} e^Z \, \frac{\exp\left(-\frac{\left(Z - z - \left(r - \frac{\sigma^2}{2}\right)(T-t)\right)^2}{2\sigma^2(T-t)} \right)}{\sqrt{2\pi\sigma^2(T-t)}} \, dZ$$

Let $m = z + (r - \frac{\sigma^2}{2})(T-t)$ and $v = \sigma^2(T-t)$, now we can simplify the integral:

$$= \int_{\ln 100}^{\infty} e^z \frac{1}{\sqrt{2\pi v}} \exp\left(-\frac{(Z-m)^2}{2v} \right) dZ = \int_{\ln 100}^{\infty} \frac{1}{\sqrt{2\pi v}} \exp\left(Z - \frac{(Z-m)^2}{2v} \right) dZ$$

$$= \int_{\ln 100}^{\infty} \frac{1}{\sqrt{2\pi v}} \exp\left(\frac{(Z-m)^2 - 2Zv}{2v} \right) dZ$$

By expanding $(Z-m)^2 - 2Zv$ we get:

$$\begin{aligned} (Z-m)^2 - 2Zv &= Z^2 - 2Zm + m^2 - 2Zv \\ &= Z^2 - 2Z(m-v) + m^2 \\ &= (Z - (m+v))^2 - 2mv - v^2 \end{aligned}$$

We can plug this back into the integral and get:

$$\begin{aligned} &\int_{\ln 100}^{\infty} \frac{1}{\sqrt{2\pi v}} \exp\left(-\frac{(Z-m)^2 - 2Zv}{2v} \right) dZ \\ &= \int_{\ln 100}^{\infty} \frac{1}{\sqrt{2\pi v}} \exp\left(-\frac{1}{2} \frac{(Z - (m+v))^2 - 2mv - v^2}{v} \right) dZ \\ &= e^{m + \frac{1}{2}v} \int_{\ln 100}^{\infty} \frac{1}{\sqrt{2\pi v}} \exp\left(-\frac{(Z - (m+v))^2}{2v} \right) dZ \\ &= e^{m + \frac{1}{2}v} \left(1 - N\left(\frac{\ln 100 - m - v}{\sqrt{v}} \right) \right) \\ &= e^{m + \frac{1}{2}v} N\left(\frac{m + v - \ln 100}{\sqrt{v}} \right) \\ &= e^{z + \left(r - \frac{1}{2}\sigma^2\right)(T-t) + \frac{1}{2}\sigma^2(T-t)} N\left(\frac{z + \left(r - \frac{1}{2}\sigma^2\right)(T-t) + \sigma^2(T-t) - \ln 100}{\sigma\sqrt{T-t}} \right) \end{aligned}$$

Since $z = \ln S_t$ and $t = 0$:

$$\int_{\ln 100}^{\infty} \phi(t, z; T, Z) \, dZ = S_t e^{rT} N\left(\frac{\ln \frac{S_t}{100} + \left(r + \frac{1}{2}\sigma^2\right) T}{\sigma\sqrt{T}} \right)$$

Now we evaluate the first integral $\int_{100}^{\infty} 100\phi(t,z;T,z)dZ$:

$$\int_{100}^{\infty} 100\phi(t,z;T,z)dZ = \int_{\ln 100}^{\infty} 100\,\frac{\exp\left(-\frac{\left(Z-z-\left(r-\frac{\sigma^2}{2}\right)(T-t)\right)^2}{2\sigma^2(T-t)}\right)}{\sqrt{2\pi\sigma^2(T-t)}}\,dZ$$

Again, let $m = z + (r - \frac{\sigma^2}{2})(T-t)$ and $v = \sigma^2(T-t)$, now we can simplify the integral:

$$= \int_{\ln 100}^{\infty} 100\,\frac{\exp\left(-\frac{(Z-m)^2}{2v}\right)}{\sqrt{2\pi v}}\,dZ = 100\left(\Phi\left(\frac{\ln(100)-m}{\sqrt{v}}\right) - \Phi(-\infty)\right)$$

where Φ is the CDF of the standard normal distribution

$$= 100N\left(\frac{\ln(100) - z - \left(r-\frac{1}{2}\sigma^2\right)(T-t)}{\sigma\sqrt{T-t}}\right) = 100N\left(\frac{\ln\left(\frac{100}{S_t}\right) - \left(r-\frac{1}{2}\sigma^2\right)T}{\sigma\sqrt{T}}\right)$$

since $t = 0$ and $Z = \ln S_t$

Now we can rewrite $C(t, e^z)$ as:

$$\begin{aligned} C(t,e^z) &= e^{-r(T-t)}\left(\int_{-\infty}^{\ln 100}\phi(t,z;T,z)dZ + \int_{\ln 100}^{\infty} e^z\phi(t,z;T,z)dZ\right) \\ &= 100e^{-r(T-t)}N(d_1) + S_tN(d_2), \end{aligned}$$

where

$$d_1 = \frac{\ln\left(\frac{100}{S_t}\right) - \left(r-\frac{1}{2}\sigma^2\right)(T-t)}{\sigma\sqrt{T-t}}, \quad d_2 = \frac{\ln\left(\frac{100}{S_t}\right) + \left(r+\frac{1}{2}\sigma^2\right)(T-t)}{\sigma\sqrt{T-t}}$$

with $r = 0.01$, $\sigma = 0.15$, $T = 10$, $t = 0$, and $S_0 = 100$, we get:

$$d_1 = 0.263 \quad \text{and} \quad d_2 = 0.448 \Rightarrow N(d_1) = 0.5105 \quad \text{and} \quad N(d_2) = 0.6729$$

If we plug these values into the formula we derived, we get:

$$\begin{aligned} \text{price at } t_0 &= 100 \cdot e^{-0.01\cdot(10-0)} \cdot 0.5105 + 100 \cdot 0.6729 \\ &= 46.192 + 67.29 \\ &= 113.482 \end{aligned}$$

```python
d1 = (np.log(100 / S0) - (r - 0.5 * sigma ** 2) * (T - t)) / (sigma * np.sqrt(T - t))
d2 = (np.log(100 / S0) + (r + 0.5 * sigma ** 2) * (T - t)) / (sigma * np.sqrt(T - t))

Nd1 = norm.cdf(d1)
Nd2 = norm.cdf(d2)

C = S0 * Nd2 + 100 * np.exp(-r * (T - t)) * Nd1

print(f"Option Price: {C}")
```

```
Option Price: 113.48497644043974
```

Figure 1: Option Price computation using Black-Scholes formula in Python

Question 2. Compare the result of this formula with the binomial tree calculation of Case 1. Plot the convergence graph of the binomial tree plus the analytical price as an extra horizontal line in your graph.

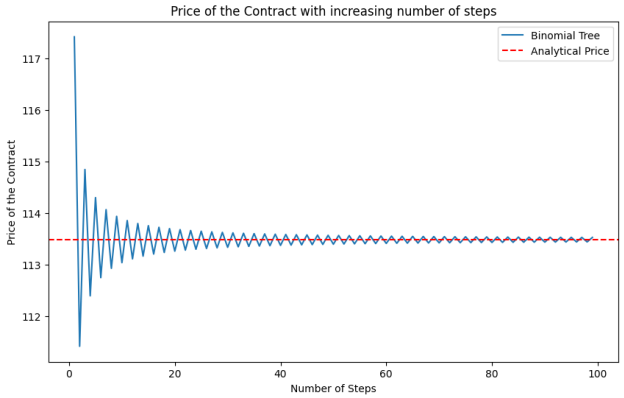

Figure 2: Convergence graph of the binomial tree and the analytical price

Figure 1 shows the convergence of the option price computed by the binomial tree as the number of steps increases. In this graph the red horizontal line represents the analytical price calculated using the Black-Scholes model. As the number of steps in the binomial tree increases, the computed price approaches the analytical price computed using the Black-Scholes formula, confirming that both methods yield the same option price in the limit.

Question 3. Calculate the "delta-hedge" at time t by computing the derivative $C_S(t, S)$ of your closed-form price-formula of the unit-linked contract. Compare the closed-form answer to the replicating portfolio for several binomial "forks" in the binomial tree.

First, I made a function that computes the "delta-hedge" of the closed-form:

```
def closed_form_deltaHedge(S, t):
    d1 = (np.log(S/100) + (r + 0.5*sigma**2)*(T - t))/(sigma*np.sqrt(T - t))
    return norm.cdf(d1)

# closed-form delta Hedge at t=0
closed_form_delta_0 = closed_form_deltaHedge(S0, 0)
print(f"Closed-form delta Hedge at t=0: {closed_form_delta_0:.4f}")
```

```
Closed-form delta at t=0: 0.6729
```

Figure 3: Computation of delta-hedge using the closed-form price-formula

Next, I adapted my **get_optionPrice** method from Case 1, which is now called **adapted-BinomialTree**. This new function returns the stock price, option price, and delta hedge for each node in the tree. The image below displays the new compuation thats was added to the method:

```
  # Delta hedge matrix
  deltaHedge = np.zeros((n, n))
  for i in range(n):
      for j in range(i+1):
          deltaHedge[i,j] = (contractPrice[i+1,j] - contractPrice[i+1,j+1]) /␣
↪(stockValues[i+1,j] - stockValues[i+1,j+1])
```

Figure 4: Delta Hedge Computation

Using the method **plot_binomial_tree**, I then plotted the delta hedge values of the binomial tree across 4 time steps. The code and resulting graph are provided below:

```
def plot_binomial_tree(stock_prices, deltas, steps):
    plt.figure(figsize=(12, 8))
    for i in range(steps + 1):
        for j in range(i + 1):
            # Plot stock prices
            plt.plot(i, stock_prices[i, j], 'bo')
            # Lines between nodes
            if i < steps:
                plt.plot([i, i + 1], [stock_prices[i, j], stock_prices[i + 1,␣
↪j]], 'b-')
                plt.plot([i, i + 1], [stock_prices[i, j], stock_prices[i + 1, j␣
↪+ 1]], 'b-')

            # Annotate delta values at each node
            delta_val = deltas[i, j]
            plt.annotate(f"{delta_val:.2f}", (i, stock_prices[i, j]),␣
↪textcoords="offset points", xytext=(0, 5), ha='center')

    plt.title("Binomial Tree for Stock Prices and Deltas")
    plt.xlabel("Time Step")
    plt.ylabel("Stock Price")
    plt.grid(True)
    plt.show()
```

Figure 5: plot_binomial_tree method

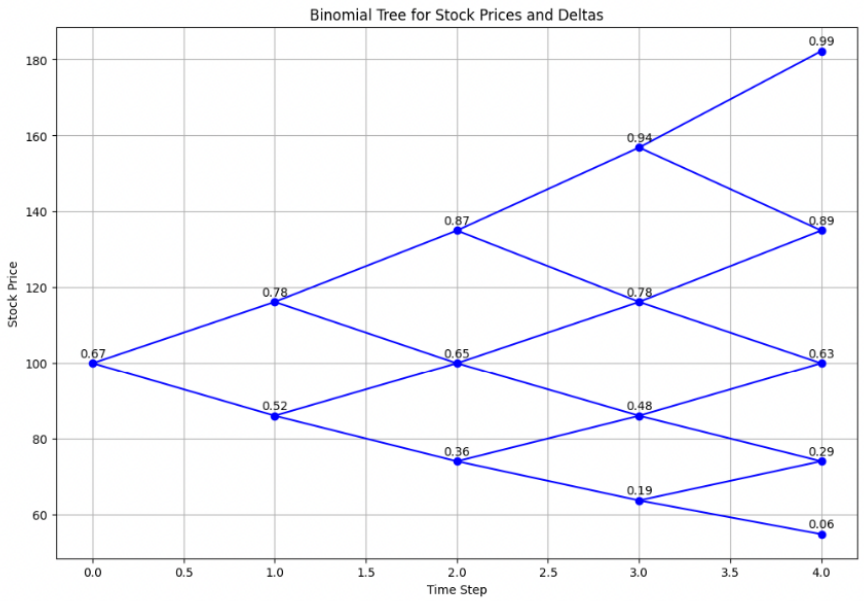

Figure 6: Binomial Tree representation with delta hedges

In this plot, I've displayed some of the delta hedge values from a binomial tree model with n=10 n=10 steps to show that the delta-hedging method works effectively. In the graph we can see that the delta hedge value (0.67) at t=0 appears similar to the closed-form delta value(0.6729).

However, to investigate further, I printed the delta hedge values at t=0 while increasing the number of steps in the binomial tree. This analysis enabled me to observe that the binomial model's delta hedge converges towards the closed-form delta hedge value as the number of steps increases, indicating that the binomial approximation improves with more time steps.

```
n_values = [10, 25, 50, 100, 400, 800, 1000]

for n in n_values:
    stockValues, contractPrice, deltaHedge = adaptedBinomialTree(r, sigma, T,␣
 ↪guarantee, n)
    print(f"Steps: {n}, Delta: {deltaHedge[0, 0]:.4f}")
```

```
Steps: 10, Delta: 0.6689
Steps: 25, Delta: 0.6715
Steps: 50, Delta: 0.6721
Steps: 100, Delta: 0.6725
Steps: 400, Delta: 0.6728
Steps: 800, Delta: 0.6729
Steps: 1000, Delta: 0.6729
```

Figure 7: Delta Hedge values for different number fo time steps