



**NANYANG  
TECHNOLOGICAL  
UNIVERSITY**  

---

**SINGAPORE**

## **MH4518 Simulation Techniques in Finance**

### **Group Project Report**

**2023-24 Semester 1**

### ***USD Drop-Back Certificate S&P 500<sup>®</sup> Index***

#### **Group Members:**

George Rahul (U2023835C)

Chopra Dhruv (U2023974A)

Moll-Elsborg Anne (N2303937K)

# Table of Contents

<b>1. Introduction</b>	<b>1</b>
1.1 Product Description	
1.2 Payoff Function	
<b>2. Preliminaries</b>	<b>1</b>
2.1 Data Collection	
2.2 Interest Rate Interpolation	
2.3 Simulation Settings	
<b>3. Black-Scholes Model</b>	<b>2</b>
3.1 Theory and Adjusting for Dividend Yield	
3.2 Comparison of Results	
3.3 Variance Reduction Techniques	
3.4 Sensitivity Analysis	
<b>4. Implied Volatility Model</b>	<b>4</b>
4.1 Theory and the VIX Index	
4.2 Comparison of Results	
4.3 Variance Reduction Techniques	[Moved to Appendix 13, 14]
4.4 Sensitivity Analysis	[Moved to Appendix 15, 16]
<b>5. Cox-Ingersoll-Ross Model</b>	<b>5</b>
5.1 Theory	
5.2 Parameter Tuning	
5.3 Comparison of Results	
<b>6. Heston Model</b>	<b>6</b>
6.1 Theory	
6.2 Parameter Tuning	
6.3 Comparison of Results	
<b>7. Conclusion</b>	<b>7</b>
<b>8. Individual Contribution</b>	<b>8</b>
<b>9. References</b>	<b>8</b>
<b>10. Appendix</b>	<b>8</b>

# 1. Introduction

## 1.1 Product Description

"USD Drop-Back Certificate" is tied to the S&P 500® Index, with a term of 3 years. It is issued by Credit Suisse AG, London Branch, and allows participation in the performance of the S&P 500® Index. The certificate also has a fixed interest component of 9.85% p.a. daily accrued and paid at maturity. It also has a mechanism of trigger events, where additional investments are made if the index drops below certain percentages from its initial level.

## 1.2 Payoff Function

The initial investment of \$1,000 is split into 2 parts, \$550 is invested in the reference index at the initial fixing date and the remaining \$450 accrues simple interest at the rate of 9.85% per annum. However, there are 3 trigger events that occur when the price of the reference index falls below 90, 85 or 80% of its initial level. At each trigger event, \$150 from the interest portion is invested into the index and generates returns from that point in time. **[NOTE: Although our backtesting window does not start from the initial fixing date, none of the trigger events have occurred.]** Combining all of these we can formulate the payoff in Python (it uses a simple routine for determining the price, if any, when a trigger event occurred),

```
def calculate_payoff(S_path, Y_0 = 1000, rate = 0.0985, years = 3):
    """
    Computes the payoff of a simulated path.
    """
    S_0 = 3790.38 # From factsheet
    S_t1 = find_trigger_price(S_path, trigger = 3411.3420) # 0.90 * S_0
    S_t2 = find_trigger_price(S_path, trigger = 3221.8230) # 0.85 * S_0
    S_t3 = find_trigger_price(S_path, trigger = 3032.3040) # 0.80 * S_0
    S_T = S_path[-1]

    # 1. Fixed returns independent of trigger events
    Y_T = (0.55*Y_0) * (S_T/S_0)

    # 2. Adjusted returns after trigger events
    if S_t3 is not None:
        multiplier = (S_T/S_t1 + S_T/S_t2 + S_T/S_t3)
    elif S_t2 is not None:
        multiplier = (S_T/S_t1 + S_T/S_t2)
    elif S_t1 is not None:
        multiplier = (S_T/S_t1)
    else:
        multiplier = 0.0

    Y_T += (0.15*Y_0) * multiplier
```

```
# 3. Daily accrued interest of 9.85% p.a.
if S_t3 is not None:
    principal = 0.0
elif S_t2 is not None:
    principal = (0.15*Y_0)
elif S_t1 is not None:
    principal = (0.30*Y_0)
else:
    principal = (0.45*Y_0)

Y_T += principal * (1 + rate * years)

# Sanity Check
assert Y_T >= 0.0
return Y_T
```

Fig 1: Payoff function realised in Python (left: index returns; right: interest accrued). `find_trigger_price()` in Appendix 1.

## 2. Preliminaries (did not include this in presentation)

### 2.1 Data Collection

The prices of the product and the underlying asset were scraped from the Credit Suisse website. The daily interest rates were downloaded from the US Treasury website. The quarterly dividend rates as well as the VIX data were obtained from the Yahoo Finance API. Links to all these resources are in the Appendix. The data cleaning scripts are also provided.

### 2.2 Interest Rate Interpolation

We interpolated the discrete set of daily interests from the US Treasury website using the *Nelson Siegel Svensson* model to obtain an interest rate curve (details in Appendix 4 - 9). This was done to minimise the error while discounting the predicted price (at the final fixing data) to the date we are trying to price.

## 2.3 Simulation Settings

Settings that were common among all experiments we ran (except the Heston model) are as follows.

- The product was priced from 9th August to 9th November, 2023.
- The backtest window was 252 days long.
- 20,000 simulations were done for pricing each day.
- We generated a set of 20,000 random normal variables and used the same set for each experiment..

Additionally, the simulations were sped up by using the **NumPy** and **Numba** libraries that leveraged parallelisation and Just-In-Time compilation to allow us to conduct 20,000 simulations in under 4 minutes. In fact, we stopped at 20,000 simulations as beyond that our laptops could not keep that many random normal variables in memory!

## 3. Black-Scholes Model

### 3.1 Theory and Adjusting for Dividend Yields

First we considered the Black-Scholes model from the lectures, which follows the stochastic differential equation  $dSt = \mu St dt + \sigma St dWt$ . Under the risk-neutral valuation,  $\mu$  can be replaced with the interest rate (from the interrelated curve) and  $\sigma$  can be estimated from the historical data from the backtest window.

Now while the S&P 500 index is not an asset it does not pay dividends, but the 500 stocks that make up the index may pay dividends and this affects the level of the index. From the lectures we incorporated this adjustment by replacing the risk-neutral interest rate with  $r - q$ , where  $q$  was the average dividend yield of the stocks in the S&P 500 (released quarterly) obtained from Yahoo Finance.

We implemented the equation we got after using Ito's Lemma in Python (left image) and also the adjustment for dividend yields in the simulations (right image),

```
1 @njit
2 def simulate_GBM(num_sim, S_0, r, sigma, delta_t, T, Z):
3     """
4     r is the daily interest rate
5     sigma is daily volatility
6     T is actually tau = T - t (in years)
7     delta_t is 1 / 252
8     Z is the random variates sampled from N(0, 1)
9     """
10    # Initializations
11    v = r - 0.5*(sigma**2)
12    num_periods = int(T/delta_t)
13    S_matrix = np.zeros(shape=(num_sim, num_periods + 1), dtype=np.float64)
14
15    # We are using the form derived by Ito's lemma
16    for i in range(num_sim):
17        S_matrix[i][0] = S_0
18
19        for j in range(1, num_periods + 1):
20            log_diff = v*delta_t + (sigma*np.sqrt(delta_t) * Z[i][j-1])
21            S_matrix[i][j] = S_matrix[i][j-1]*np.exp(log_diff)
22
23    return S_matrix
```

Fig 2: Implementation of Geometric Brownian Motion.

```
1 # GBM with Dividends without Variance Reduction
2 GBM_Div_expected_values = []
3
4 for date, prices in zip(simulation_dates, backtest_windows):
5     # Convert date to index (so we can lookup prices and interest rate curves)
6     date_index = simulation_dates.get_loc(date)
7
8     # Estimate r
9     delta = len(product_lifetime[date:])
10    tau = delta / 252 # time diff in years
11    curve_fit = curves[date_index]
12    r_yearly = curve_fit(tau) / 100 # yearly (interpolated) interest rate
13
14    # Calculate mu and sigma (we discard mu)
15    _, sigma = get_lognormal_statistics(prices, delta_t) # sigma is daily variance
16
17    # IMPORTANT: Adjust the interest rate based on dividends
18    r_adjusted_yearly = r_yearly - get_dividend_rate(date) / 100
19    r_adjusted_daily = r_adjusted_yearly / 252
20
21    # Simulate prices, calculate payoff and expected value
22    simulated_prices = simulate_GBM(num_sim, prices[-1], r_adjusted_daily, sigma, delta_t, tau, Z=Z_matrix[date_index])
23    payoffs = np.exp(-r_adjusted_yearly*tau)*np.apply_along_axis(funcid=calculate_payoff, axis=1, arr=simulated_prices)
24    expected_value = np.mean(payoffs)
25
26    GBM_Div_expected_values.append(expected_value)
```

Fig 3: Dividends Adjustment during GBM Simulations.

### 3.2 Comparison of Results

We plotted the predicted prices from GBM (both with and without the dividends yield adjustment) and used the Mean Absolute Error (MAE) and Mean Squared Error (MSE) as a metric to measure the accuracy of the predictions. **[NOTE:** We will continue to use the MAE and MSE metrics as well as the Time Series plot for comparing the results of all subsequent experiments].

As you can see both models successfully captured the trends in the prices, the only difference being the constant spread difference has been accounted for by the adjusted model. We shall use the adjusted model as the baseline for comparison for the more complicated models. (images are on the next page)

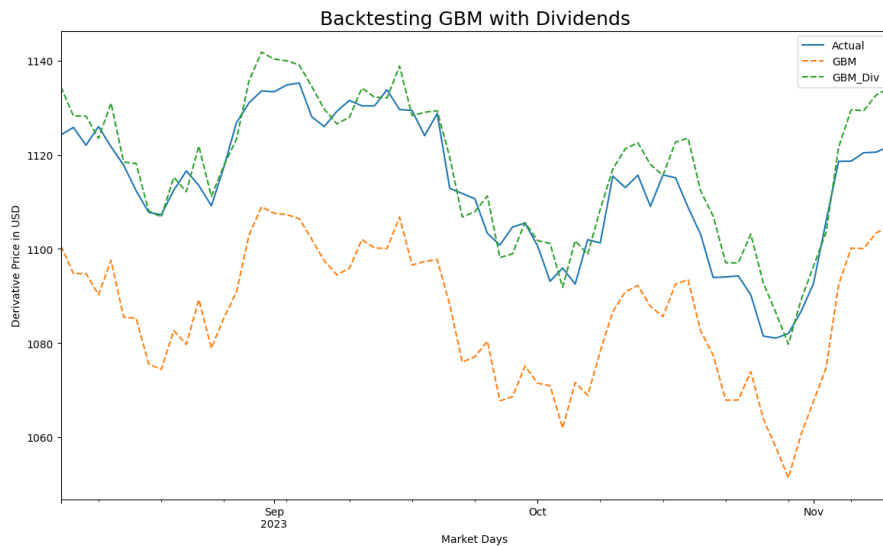


Fig 4: Comparing the results of GBM with and without the dividends adjustment with the actual price.

Mean Absolute Error for standard GBM:	27.1916
Mean Squared Error for standard GBM:	770.7400
Mean Absolute Error for GBM with dividends:	5.2958
Mean Squared Error for GBM with dividends:	41.5403

Fig 5: Comparing the MAE and MSE of the predictions from the GBM models.

### 3.3 Variance Reduction Techniques

We experimented with the Antithetic Variates, Control Variates (using terminal price as the control) and Empirical Martingale Correction techniques discussed in the lectures. **[NOTE:** The variance reduction was applied on the GBM with the dividend yield adjustment.]

The predictions experienced a marginal enhancement, accompanied by a slight decrease in variance. This may be attributed to the substantial number of simulations (20,000), which had already minimised the variance to its lowest extent.



Fig 6: Comparing the results of the variance reduction on the GBM with dividends with the actual price.

Mean Absolute Error for GBM with dividends:	5.2958
Mean Squared Error for GBM with dividends:	41.5403
Mean Absolute Error for GBM_Div with AV:	4.9607
Mean Squared Error for GBM_Div with AV:	38.3676
Mean Absolute Error for GBM_Div with EMS:	5.0107
Mean Squared Error for GBM_Div with EMS:	38.7846
Mean Absolute Error for GBM_Div with CV:	5.2958
Mean Squared Error for GBM_Div with CV:	41.5405

Fig 7: Comparing the MAE and MSE of the predictions from the variance reduction techniques.

### 3.4 Sensitivity Analysis

To assess the model's response to changes in the underlying asset price, the Greeks  $\delta$  and  $\Gamma$  are estimated using the Finite-difference Method for each day in our simulation period. We set  $h = 0.001 * (\text{terminal price})$  and used the GBM with dividend yields to obtain the 3 price paths. The plots of the results are, (images are on the next page)

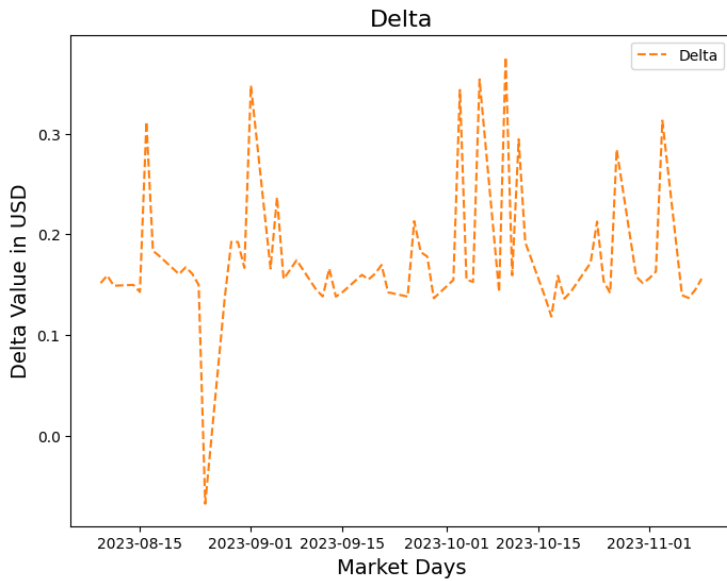


Fig 8: The simulated Deltas of the GBM with dividends.

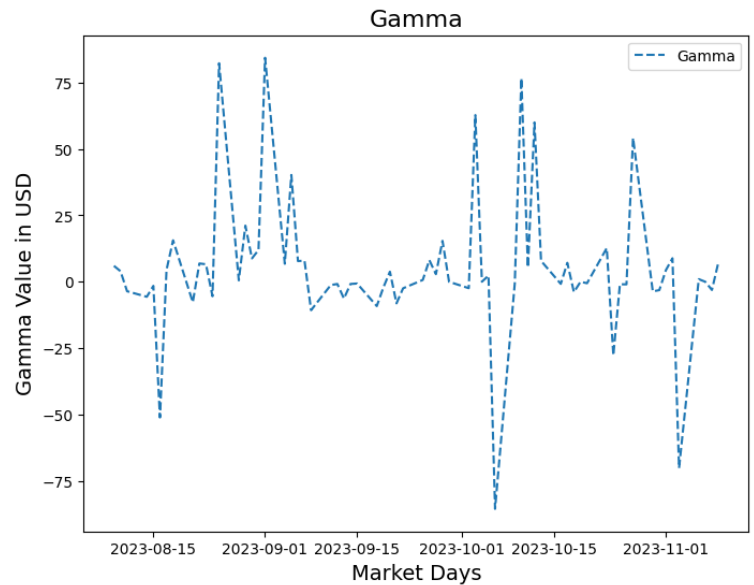


Fig 9: The simulated Gammas of the GBM with dividends.

Surprisingly the delta values are quite small, which indicates the price of the product is quite stable with respect to the underlying asset. The gammas are also on the lower end and sync up to the spikes in the deltas.

## 4. Implied Volatility Model

### 4.1 Theory and the VIX Index

In Black-Scholes we estimated  $\sigma$  using the historical prices, a.k.a., the historical volatility approach. This may not be the best indicator of the future volatility of the market. The implied volatility approach considers a forward-looking estimate of  $\sigma$ , that is calculated from live option prices. But the question arises, how do we gather the prices of the options of the 500 stocks in the S&P 500 and estimate the volatility?

Thankfully, the Chicago Board Options Exchange (CBOE) does this for us with the VIX Index. VIX is the ticker symbol and the popular name for the CBOE's Volatility Index, a popular measure of the stock market's expectation of volatility based on S&P 500 index options.

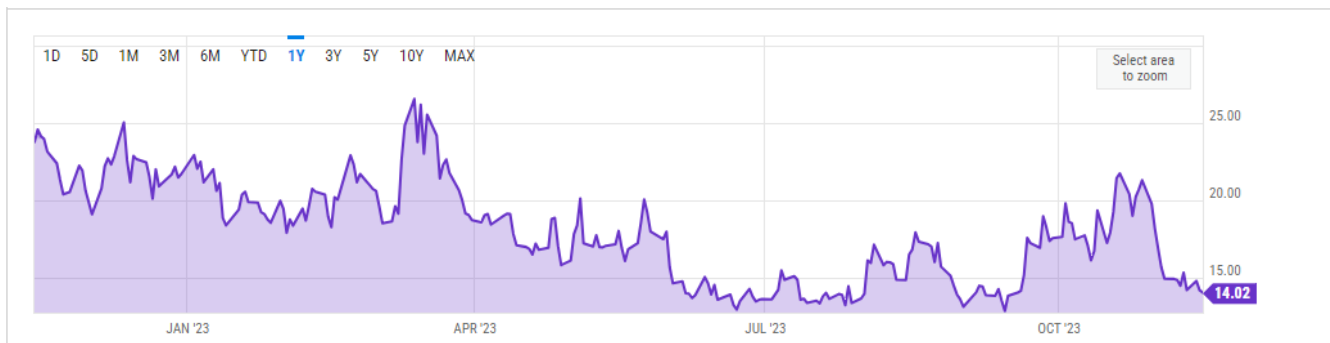


Fig 10: Graph of VIX index that gives us a 30-day forward estimate of the volatility (Implied Volatility)

### 4.2 Comparison of Results

As we can see the implied volatility improved the accuracy and correctly reduced the overpricing at the peaks. Clearly the implied volatility was a better estimator of the actual underlying volatility in the market compared to the historical volatility.

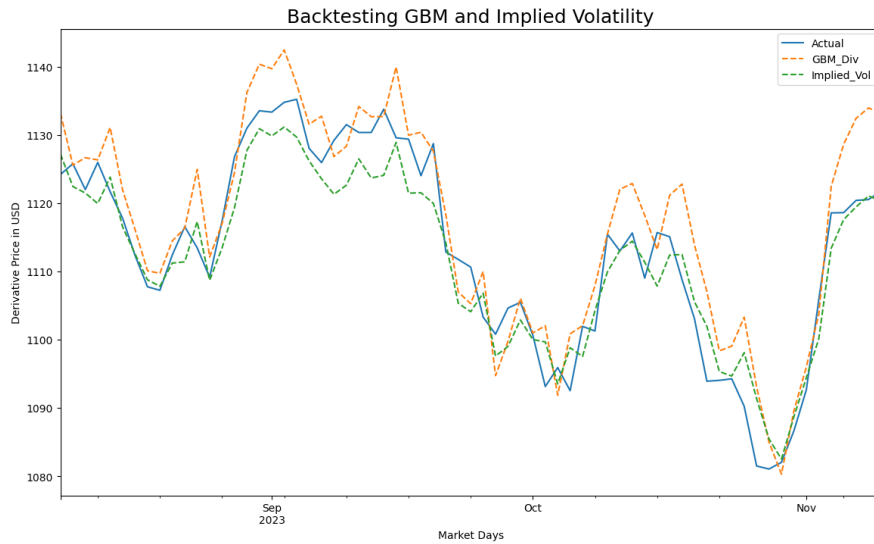


Fig 11: Comparing the results of Implied Volatility model vs GBM with dividends adjustment and the actual price.

Mean Absolute Error for GBM with div:	5.3526
Mean Squared Error for GBM with div:	43.5266
Mean Absolute Error for Implied Vol:	3.7179
Mean Squared Error for Implied Vol:	21.0213

Fig 12: Comparing the MAE and MSE of the predictions from the IV and GBM models.

## 5. Cox, Ingersoll, and Ross Model

### 5.1 Theory

Cox, Ingersoll & Ross(1985) studied the term structure of interest rates. They came up with a model to determine the bond prices, with the interest rate 'r' following a stochastic process determined by the equation:

$$dr = \kappa(\theta - r) dt + \sigma\sqrt{r} dz_1.$$

Where the parameter k is called the 'speed of reversion', theta is the long term interest rate, sigma is the standard deviation of the interest rate, and  $Z_1$  is the standard normal random variable.

The main benefit of modelling the interest rate with the above stochastic equation is that the mean reverting property of the interest rate, which has been observed empirically, can be given a mathematical form. This means that the interest rate r is driven towards the long term mean theta, with a speed of reversion K.

E.g. if the interest rate is below the long term mean, (theta-r) will be positive and then dr or the change in r will *more likely* be positive, driving the interest rate upwards.

We can substitute this stochastic r in each step of the Black-Scholes model, and capture the variations in the derivative prices more accurately. In our example, we have simulated the interest rate r for bonds maturing in 2 and 3 years, and have used their simple linear interpolation to determine the list of interest rates [r] used for running the simulations at each day in our backtesting window.

### 5.2 Parameter Tuning (Although implemented correctly we presented the wrong formula)

For tuning the aforementioned model parameters kappa, theta and sigma, we need to minimise the summation of the following term:

$$\text{Minimize} \sum_t \frac{(r[t] - \theta + \theta \cdot (r[t-1] - \theta))^2}{r[t]}$$

### 5.3 Comparison of Results

As we can observe from the backtesting results, the CIR model performs slightly better than the Black-Scholes model by incorporating some additional information about the interest rate dynamics.

Backtesting CIR and GBM wth Dividends

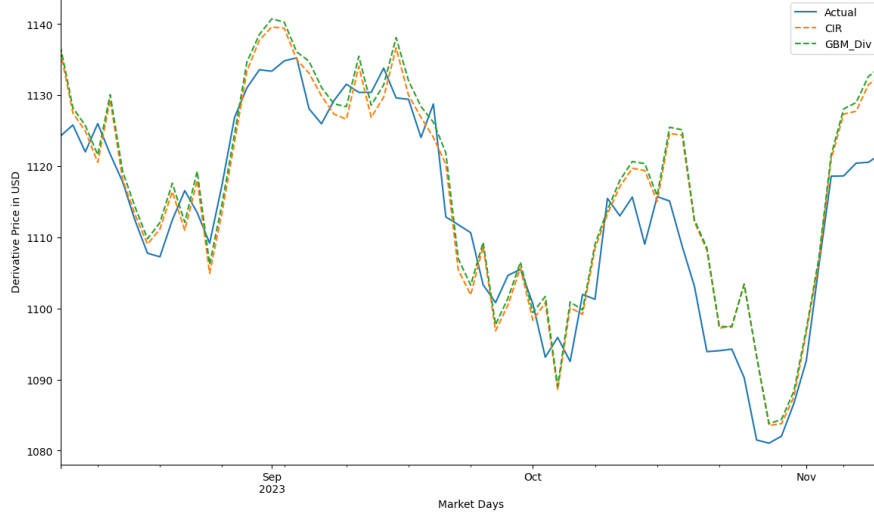


Fig 13: Comparing the results of CIR model vs GBM with dividends adjustment and the actual price.

Mean Absolute Error for GBM_Div:	5.407055586108778
Mean Squared Error for GBM_Div:	43.38799028539089
Mean Absolute Error for CIR:	5.1283271963728065
Mean Squared Error for CIR:	38.95355947390155

Fig 14: Comparing the MAE and MSE of the predictions from the CIR and GBM models.

## 6. Heston Model

### 6.1 Theory

In the Heston model, both the derivative price as well as the variance term are assumed to be stochastic in nature. The model dynamics for the Heston model are given as follows:

$$dS_t = rS_t dt + \sqrt{V_t} S_t dW_{1t}$$

$$dV_t = k(\theta - V_t) dt + \sigma \sqrt{V_t} dW_{2t}$$

Similar to the interest rate in the CIR model, the Variance term in the Heston model has a mean reverting property. This provides the model more flexibility in modelling derivative prices, and often leads to better results than the GBM model which assumes a constant volatility.

### 6.2 Parameter Tuning

The Heston model requires the tuning of five model parameters. These are  $r$ ,  $k$ ,  $\sigma$ ,  $\theta$  &  $\rho$  ( $\rho$  represents the correlation between the variance term and the derivative price). For tuning the above, we utilise the Maximum Likelihood approach and attempt to maximise the log-likelihood of the joint probability density of the  $f(Q_{t+1}, V_{t+1})$ . Dunn et. al. (2014) provide the formula for the log-likelihood function as

$$\begin{aligned} \ell(r, k, \theta, \sigma, \rho) = & \sum_{t=1}^n \left( -\log(2\pi) - \log(\sigma) - \log(V_t) - \frac{1}{2} \log(1 - \rho^2) \right. \\ & - \frac{(Q_{t+1} - 1 - r)^2}{2V_t(1 - \rho^2)} + \frac{\rho(Q_{t+1} - 1 - r)(V_{t+1} - V_t - \theta k + kV_t)}{V_t \sigma(1 - \rho^2)} \\ & \left. - \frac{(V_{t+1} - V_t - \theta k + kV_t)^2}{2\sigma^2 V_t(1 - \rho^2)} \right). \end{aligned}$$

We maximise the above log-likelihood function at each window, and utilise the parameter values returned for simulation.

### 6.3 Comparison of Results

As we can see the Heston model accurately captures the underlying trends of the product price. From the metrics we can see that it is our one of best models for pricing the drop-back certificate. **[NOTE]**: Due to time and resource constraints we



we're only able to price the product from 9th August, 2023 to 20th October, 2023. Hence we cannot say it's the best as it is not a fair comparison to the other models that priced the product for 17 more days (till 9th November 2023)].

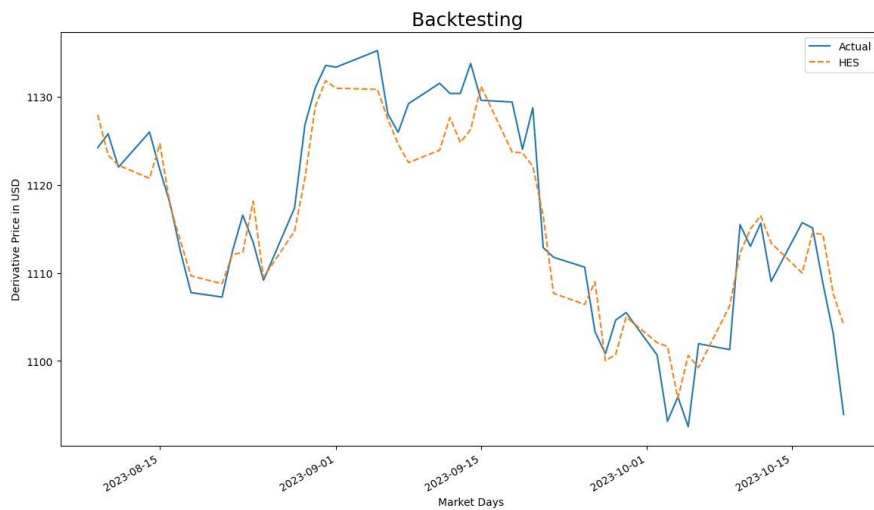


Fig 15: Comparing the results of Heston model vs GBM with dividends adjustment and the actual price.

Mean Absolute Error for HES:	3.4706635338952676
Mean Squared Error for HES:	18.25832116410608

Fig 16: Analysing the MAE and MSE of the predictions from the Heston model.

## 7. Conclusion (Expanded our discussion)

In this project, we have utilised various modelling paradigms to predict the derivative price across rolling windows. Now, we attempt to interpret our results using two ablation experiments:

1. How does our product compare to a similar product with the same initial portfolio distribution (\$550 in S&P 500 & \$450 at 9.85% simple interest p.a.) but *no* triggers. That is, how do the triggers affect the price of the product?
2. What is the difference between investing the denomination in this financial product versus directly investing the denomination in an ETF maintaining the S&P 500 index?

For the first query, we ran simulations using the GBM model on both the real and our hypothetical product with no triggers. The results are presented as follows:

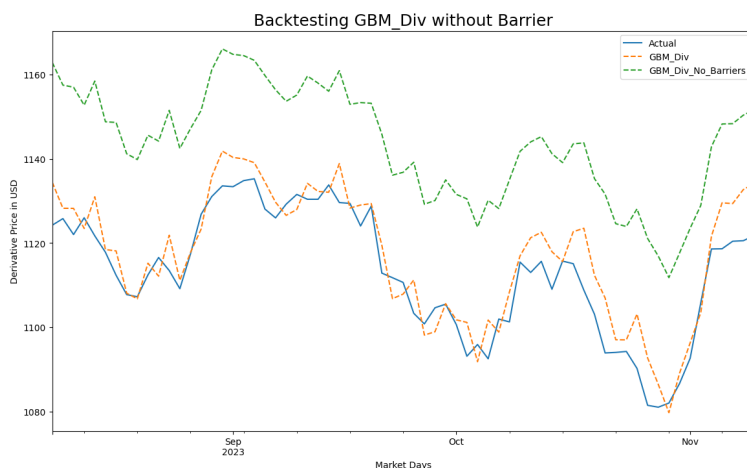


Fig 17: Comparison of results with the GBM model (with dividend adjustment) using the normal payoff and a payoff without any trigger events [See Appendix 2]

From the graph above, we can clearly observe that the hypothetical product without triggers would be priced **higher** than the existing financial product. This can be primarily attributed to the fact that the 9.85% simple interest p.a. is much higher than the historical average return on the S&P 500 index, which is 6.43%. Thus, the value of this product would be higher if

none of the triggers are ever hit, and no proportion of the funds are transferred from the faster growing component to the S&P 500 index.

We have established above that investing in the financial product with the same denomination would yield more on average than investing directly in the S&P 500 index in a bull market. However, what would happen in the scenario in which the price of S&P 500 index falls? Our product would transfer some amount from the faster growing component to the S&P 500 index. The rationale here is that if the S&P 500 index recovers quickly, the transferred amount would also grow quickly, and would lead to better returns than the case in which all the capital was invested in the S&P 500 index. However, if the value of the S&P 500 index does not recover quickly enough (at less than 9.85%) after a crash, the transfer would be worse than the original distribution of the funds. In the worst case, the value of the index falls to zero, and the buyers lose all their investment.

## 8. Individual Contribution

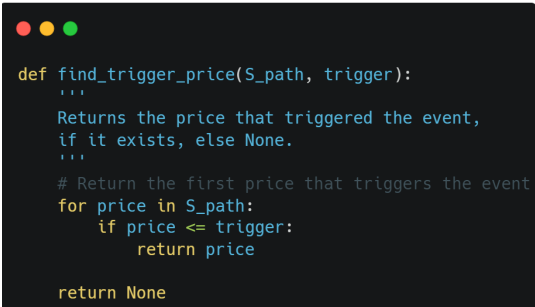
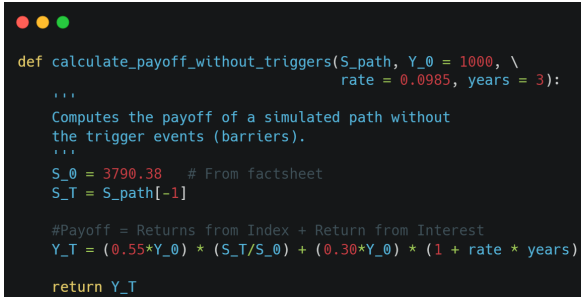
As all of us collaborated on multiple different topics in this project we felt it would be easier to tabulate the overall contribution provided by each team member towards the completion of this group project.

Name	List of Topics Worked On	Overall Contribution to Project
George Rahul	Data Collection and Cleaning, Payoff Formulation, GBM, Implied Volatility, CIR, Heston, Variance Reduction Techniques, Presentation, Report	40%
Chopra Dhruv	Payoff Formulation, GBM, Implied Volatility, CIR, Heston, Variance Reduction Techniques, Presentation, Report	40%
Moll-Elsborg Anne	1. Sensitivity Analysis (few mistakes but done well overall) 2. Implied Volatility (incorrect implementation which had lookahead bias and did not use VIX data, needed to be redone from scratch) 3. Presentation (worked on the slides but did not attend presentation)	20%

## 9. References

1. Brown, Roger H., and Stephen M. Schaefer. "The term structure of real interest rates and the Cox, Ingersoll, and Ross model." *Journal of Financial Economics* 35.1 (1994): 3-42.
2. Dunn, Robin, et al. "Estimating option prices with Heston's stochastic volatility model." (2014).

## 10. Appendix

 <pre>def find_trigger_price(S_path, trigger):     """     Returns the price that triggered the event,     if it exists, else None.     """     # Return the first price that triggers the event     for price in S_path:         if price &lt;= trigger:             return price      return None</pre>	 <pre>def calculate_payoff_without_triggers(S_path, Y_0 = 1000, \                                      rate = 0.0985, years = 3):     """     Computes the payoff of a simulated path without     the trigger events (barriers).     """     S_0 = 3790.38 # From factsheet     S_T = S_path[-1]      #Payoff = Returns from Index + Return from Interest     Y_T = (0.55*Y_0) * (S_T/S_0) + (0.30*Y_0) * (1 + rate * years)      return Y_T</pre>
1. The find_trigger_price() function	2. The Payoff function without triggers (used in the ablation study)

$$r(T) = \beta_0 + \beta_1 \left[ \frac{1 - \exp(-T/\lambda_0)}{T/\lambda_0} \right] + \beta_2 \left[ \frac{1 - \exp(-T/\lambda_0)}{T/\lambda_0} - \exp(-T/\lambda_0) \right] + \beta_3 \left[ \frac{1 - \exp(-T/\lambda_1)}{T/\lambda_1} - \exp(-T/\lambda_1) \right]$$

### 3. The Nelson Siegel Svensson Model for interpolating the daily interest rate

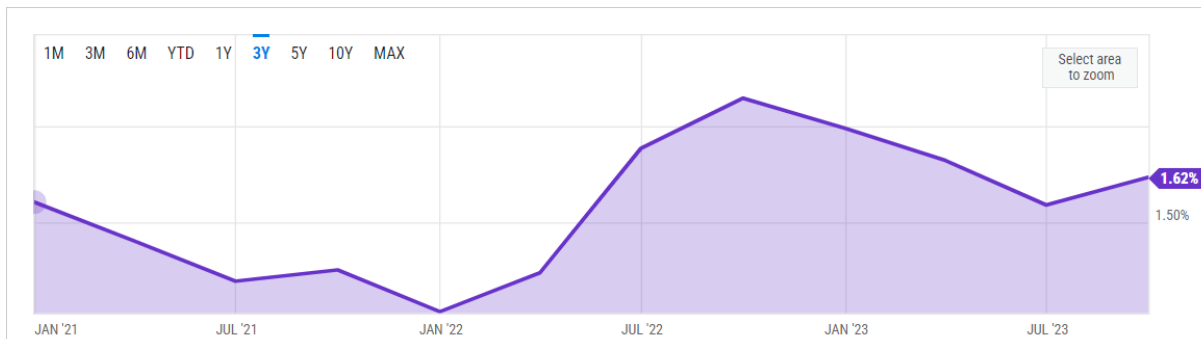
4. Link to the structured product website where the asset and product prices were scraped from:	<a href="https://derivative.credit-suisse.com/ch/ch/en/detail/drop-back-certificate-s-p-500/CH1199361879/119936187">https://derivative.credit-suisse.com/ch/ch/en/detail/drop-back-certificate-s-p-500/CH1199361879/119936187</a>
5. Link to the US Treasury website where the daily treasury rates were downloaded from:	<a href="https://home.treasury.gov/policy-issues/financing-the-government/interest-rate-statistics">https://home.treasury.gov/policy-issues/financing-the-government/interest-rate-statistics</a>
6. Link to the website where the S&P 500 Dividend Yield rates were obtained from:	<a href="https://www.nasdaq.com/market-activity/etf/spy/dividend-history">https://www.nasdaq.com/market-activity/etf/spy/dividend-history</a>
7. Link to the Yahoo Finance Python Library:	<a href="https://pypi.org/project/yfinance/">https://pypi.org/project/yfinance/</a>
8. Link to Numba documentation:	<a href="https://numba.pydata.org/numba-doc/latest/index.html">https://numba.pydata.org/numba-doc/latest/index.html</a>
9. Link to Numpy documentation:	<a href="https://numpy.org/">https://numpy.org/</a>

Initial Level	1. Trigger Level 90%	2. Trigger Level 85%	3. Trigger Level 80%
3,790.38	3,411.3420	3,221.8230	3,032.3040

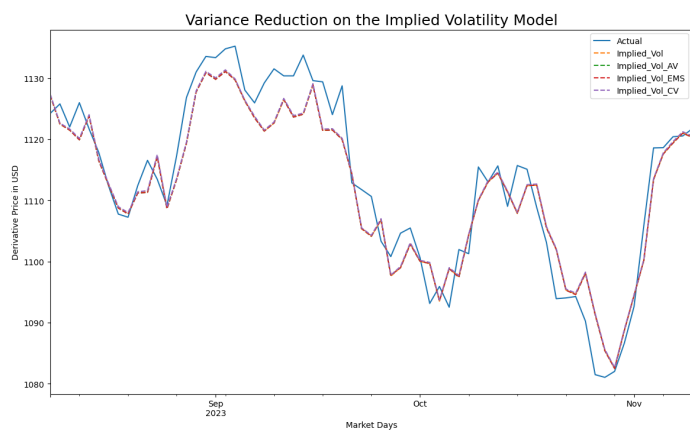
### 10. The Prices of the Trigger Events



11. The SP500 Index. Notice how the barriers have not been hit throughout the lifetime of the product.



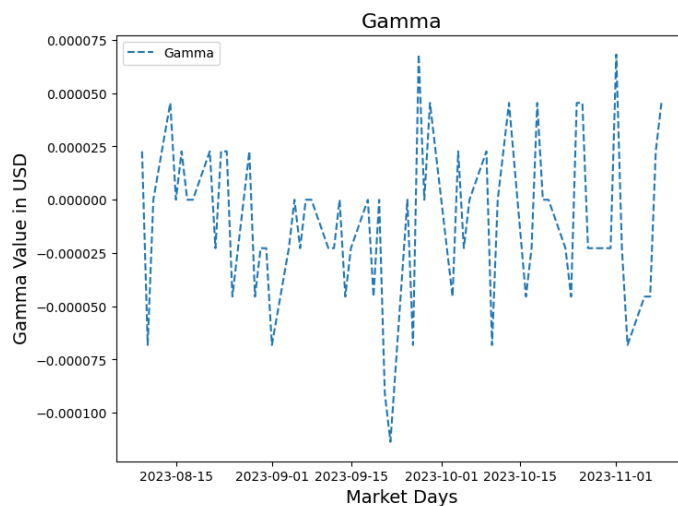
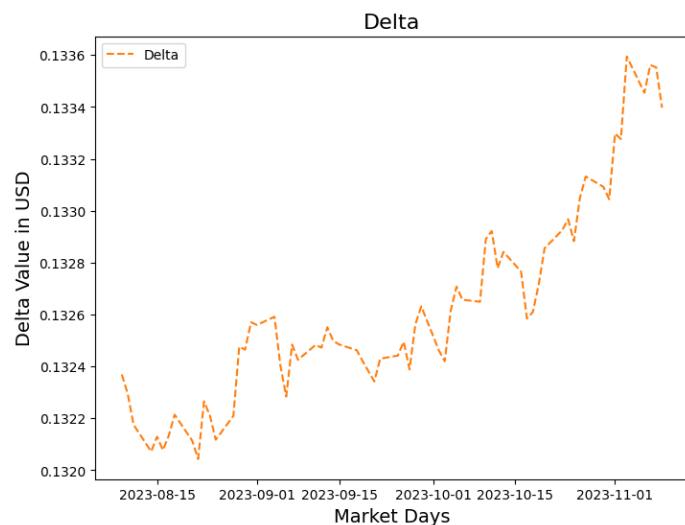
## 12. The SP500 Quarterly Dividend Yields



Mean Absolute Error for Implied Vol:	3.7179
Mean Squared Error for Implied Vol:	21.0213
Mean Absolute Error for Implied Vol with AV:	3.7135
Mean Squared Error for Implied Vol with AV:	20.9723
Mean Absolute Error for Implied Vol with EMS:	3.7137
Mean Squared Error for Implied Vol with EMS:	20.9736
Mean Absolute Error for Implied Vol with CV:	3.6700
Mean Squared Error for Implied Vol with CV:	20.5215

## 13. Results of the Variance Reduction on the Implied Volatility Model

## 14. Metric of the Variance Reduction on the Implied Volatility Model



## 15. The simulated Deltas of the Implied Volatility Model

## 16. The simulated Gammas of the Implied Volatility Model

We simulated the Greeks using the Finite Difference method with  $h = 0.001 \cdot \text{terminal price}$