Home Assignment 2 - Quantitative Finance

Preliminary Solutions

Aleksandar Gradev (12321078)

18.12.2025

Task definition

In the German market various index certificates are traded. From a legal perspective a certificate is a bond which guarantees a certain stream of payments to its holders. In the standard case (straight bond) 100% of the notional value is paid at the maturity of the bond and there are fixed annual coupon payments. For an index certificate the payment at maturity and/or the coupon payments depend on the development of the DAX.

This assignment is based on the following framework:

- All instruments are traded in a frictionless market with a continuously compounded riskless money market account with an annualized rate of 3%.
- The starting value of the DAX is 23,500.
- The DAX follows a Geometric Brownian Motion with a drift rate of 8.7% under the physical measure and a volatility of 14%
- Each certificate has a maturity of five years and pays annual coupons (at the end of years 1, 2, 3, 4, and 5).

I will start by initializing the known parameters from the task definition:

```
# Init parameters
S_0    <- 23500  # DAX starting price
drift  <- 0.087  # Drift under P
sigma  <- 0.14   # Volatility
r_free <- 0.03   # Risk free rate

K      <- 25000  # Strike price
M      <- 60     # Number of time steps
T      <- 5      # Maturity in years
N      <- 5000   # Number of runs with generated numbers
delta  <- T/M    # Length of time step
```

Underlying's price

First, I will start by generating the random numbers for the Monte Carlo simulation. Since we have to work with antithetic numbers, I will first generate $5000 \cdot 60$ normal random numbers. Those numbers will be stored in a 60 x 5000 matrix, where the rows represent the time steps and the columns represent each draw.

```r
# Create a MxN empty matrix to filled with normal random numbers
Z <- matrix(rnorm(M*N), nrow = M, ncol = N)

# Build a small helper function to see the dimensions of a matrix in a nice way
dimension <- function(X) {
  cat("The matrix has", dim(X)[1], "rows and", dim(X)[2], "columns\n")
}

dimension(Z)
```

```
## The matrix has 60 rows and 5000 columns
```

Now, to generate the antithetic numbers, we just have to compute $-Z_i$ for each Z_i random number. I will do this by directly multiplying the matrix $\mathbf{Z}$ from the left with a diagonal square matrix of size 60 filled with -1 on the diagonal. Then I will bind this $-\mathbf{Z}$ matrix to the $\mathbf{Z}$ matrix.

```r
# Compute the antithetic numbers
anti_Z <- diag(-1, nrow = M) %*% Z

# Bind the antithetic numbers
Z <- cbind(Z, anti_Z)

# Check results
dimension(Z)
```

```
## The matrix has 60 rows and 10000 columns
```

```r
Z[1:3,1:3]
```

```
##            [,1]       [,2]       [,3]
## [1,] -0.3186741  0.2563870 -0.2760655
## [2,]  0.3921667 -0.2304095  0.3514772
## [3,]  0.3567374  0.9329366 -1.3620721
```

```r
Z[1:3, (1+N):(3+N)]
```

```
##            [,1]       [,2]       [,3]
## [1,]  0.3186741 -0.2563870  0.2760655
## [2,] -0.3921667  0.2304095 -0.3514772
## [3,] -0.3567374 -0.9329366  1.3620721
```

Now I will initialize an empty 61 x 10 000 matrix to store the DAX price. We want to have 61 (M+1) rows so that we can store the price at t_0 in the first row. Consequently, we want to have 10 000 (2xN) columns as we have to account for prices computed with the antithetic numbers.

```r
# Init prices matrix
S <- matrix(NA, nrow = M+1, ncol = 2*N)
dimension(S)
```

```
## The matrix has 61 rows and 10000 columns
```

```r
# Fill the first row with the t_0 price of the DAX
S[1,] <- S_0
```

The matrix is ready. I will first calculate the drift and the volatility shocks under the physical measure using the following formulas:

$$\text{Drift} = (r_f - \frac{1}{2}\sigma^2) \cdot \Delta t$$

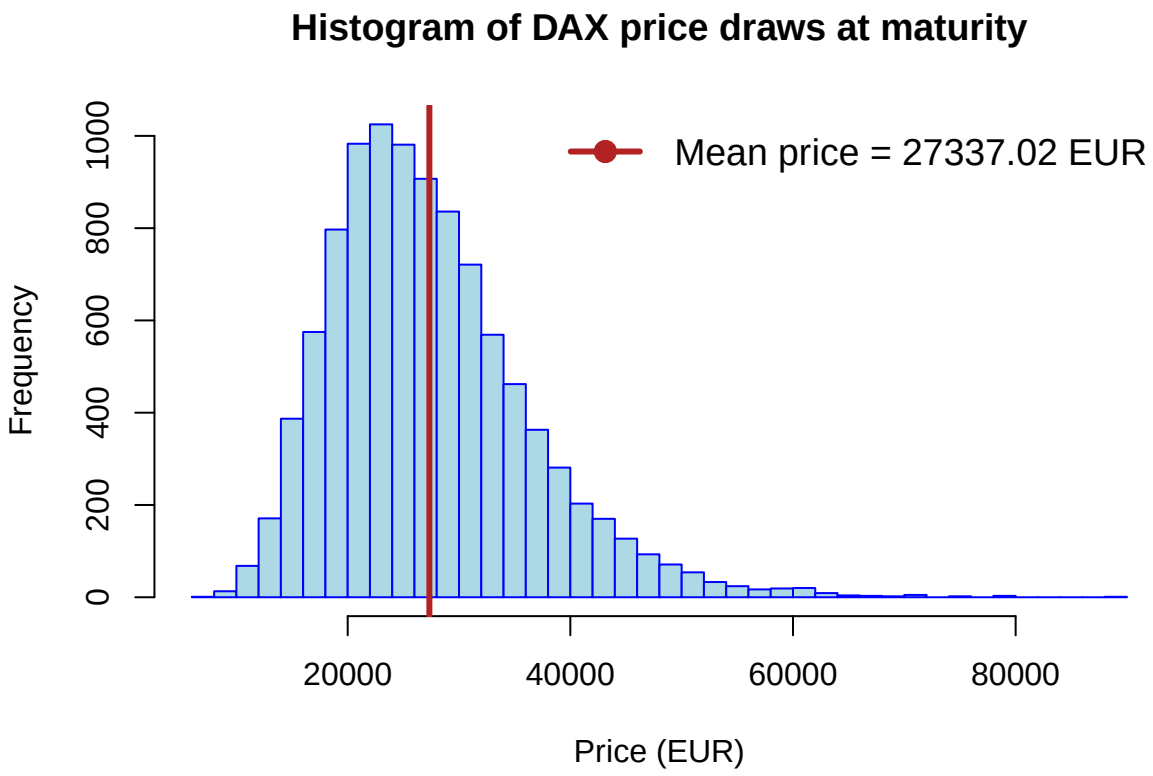
$$\text{Vol Shock} = \sigma * \sqrt{\Delta t}$$

Then, to obtain the price at time t, we just have to use the formula:

$$S_{t+\Delta t} = S_t \cdot \exp\left(\underbrace{(r - \frac{1}{2}\sigma^2)\Delta t}_{\text{Drift}} + \underbrace{\sigma\sqrt{\Delta t}}_{\text{Vol Shock}} \cdot Z \right)$$

I will exploit the vectorization of R and compute all draws of the price at t_i, i.e. in the loop I will compute one whole row at a time. This optimizes significantly the computing time.

```r
# Calculate drift and volatility shock
drift <- (r_free-0.5*sigma^2)*delta
vol_shock <- sigma*sqrt(delta)

# Calculate the prices USING VECTORIZATION
for (t in 2:(M+1)) {
  S[t, ] <- S[t-1, ] * exp(Z[t-1, ]*vol_shock + drift)
}

# Look at the distribution of DAX prices at t_60
{
hist(S[61,],
     breaks = 30,
     main   = "Histogram of DAX price draws at maturity",
     xlab   = "Price (EUR)",
     col    = 'lightblue',
     border = 'blue')

mean_S_maturity <- mean(S[61,])

abline(v   = mean(S[61,]),
       col = "firebrick",
       lwd = 3)

legend("topright",
       legend = paste("Mean price =", round(mean_S_maturity, 2), "EUR"),
       col    = "firebrick",
       lwd    = 3,
       pch    = 19,
       bty    = "n",
       cex    = 1.2)
}
```



European Call Option

Now, when we know the distribution of the DAX price at maturity, we can see the option value for each draw. We can apply the function for the European call payoff and obtain the distribution of call values at maturity. Then we can compute the mean and discount it to its present value.

```
# Build a helper function for the European call payoff
european_call_payoff <- function(X, K) {
  C <- max(X - K, 0)
  return(C)
}

# Create a vector with the draws of the final call value given the payoff func
C_T <- sapply(S[61, ], function(X) european_call_payoff(X, K))
C_0 <- exp(-r_free*T)*mean(C_T)

cat("The present value of the European Call option written on the DAX is", C_0,
    "EUR")
```

```
## The present value of the European Call option written on the DAX is 3949.663 EUR
```

Bonus Certificate

The Bonus certificate is a financial instrument with a maturity of five years that pays an annual coupon of 10% if the DAX remains above a specified lower boundary, which in our case is 28 000 EUR. If the DAX falls below this boundary on a coupon date, the coupon payment for that year is zero. Regardless of the index's performance, the certificate guarantees a terminal payment of 100% of the notional value at maturity.

More formally, we can describe the Bonus certificate in the following way:

Let S_i denote the price of the underlying at time i, where $i \in [0, M]$ and B_i denote the bonus certificate value at time i. Let N denote the notional price of the certificate, BB denote the bonus boundary of the certificate and T denote the terminal date.

Then the coupon payment C_t, where $t \in [1, T]$, is defined as:

$$C_t = \begin{cases} 0.10 \cdot N & \text{if } S_{\frac{M}{T} \cdot t} > BB \\ 0 & \text{otherwise} \end{cases}$$

We know that the present value of the coupon payment is

$$C_{t_0} = e^{-r.t} \cdot C_t$$

Then by the no-arbitrage principle we can say that the present price of this certificate has to be equal to the present value of all its future cash flows payments, i.e.

$$C_0 = \underbrace{\sum_{t=1}^{T} C_{t,0}}_{\text{Coupon payments}} + \underbrace{N \cdot e^{-r \cdot T}}_{\text{Notional payment}}$$

For our case, let the notional value N be 100 EUR. Since we want to use the vectorization of R, we can build a matrix $\mathbf{C}$ with each column being a draw of the Monte Carlo simulation, where the rows represent the payments of the certificate. Moreover, we can build a vector $\mathbf{D}$ that should store the discounting coefficients. Then if we multiply $\mathbf{D}'$ with $\mathbf{C}$ we will obtain a vector with the sum of the present value of all payments.

$$\mathbf{D}' \cdot \mathbf{C} = \begin{pmatrix} e^{-r_f} & e^{-2r_f} & e^{-3r_f} & e^{-4r_f} & e^{-5r_f} \end{pmatrix} \cdot \begin{pmatrix} C_{1,1} & C_{1,2} & \dots & C_{1,10\,000} \\ C_{2,1} & C_{2,2} & \dots & C_{2,10\,000} \\ C_{3,1} & C_{3,2} & \dots & C_{3,10\,000} \\ C_{4,1} & C_{4,2} & \dots & C_{4,10\,000} \\ N + C_{5,1} & N + C_{5,2} & \dots & N + C_{5,10\,000} \end{pmatrix} = \begin{pmatrix} C_{1_0} & C_{2_0} & \dots & C_{10\,000_0} \end{pmatrix}$$

```r
# Init parameters of the certificate
S_coupon_dates <- S[c(12,24,36,48,60)+1, ]
BB <- 28000
Notional <- 100

# Init empty matrix that will store the coupon payments
coupon_payment_matrix <- matrix(NA, nrow = dim(S_coupon_dates)[1],
                                ncol = dim(S_coupon_dates)[2])

# For each time step (and path) check whether a coupon is payed
for (i in 1:length(S_coupon_dates)) {
  coupon_payment_matrix[i] <- ifelse(S_coupon_dates[i] > BB, 0.1*Notional, 0)
}
```

```r
# Add the terminal payment of the notional
coupon_payment_matrix[dim(S_coupon_dates)[1], ] <- (
  coupon_payment_matrix[dim(S_coupon_dates)[1], ] + Notional)

# Init discounting vector (storing the discounting coeffs)
discounting_vector <- c()
for (i in 1:T) {discounting_vector <- c(discounting_vector, exp(-r_free*i))}

# Calculate the sum of present value of all future cash flows for each draw
certificate_prices <- discounting_vector %*% coupon_payment_matrix

# Compute the mean certificate price
mean_certfificate_price <- mean(certificate_prices)
cat("The present value of the Bonus Certificate written on the DAX is",
    mean_certfificate_price, "EUR")
```

```
## The present value of the Bonus Certificate written on the DAX is 99.33553 EUR
```