Options dans le cadre Black-Scholes

TP-2: Pricing Vanna-Volga

Version: 22 mar 2022

The purpose of this problem set is to explore the Vanna-Volga pricing model. In this problem set, you will use the following functions:

GBSPrice: Price of a vanilla option:

$$P = f(\text{PutCall}, S, K, T, r, b, \sigma)$$

where:

PutCall 'c' for a call, 'p' for a put

b cost of carry: ridk free rate r less dividend yield d

r risk-free rate

```
GBSPrice <- function(PutCall, S, K, T, r, b, sigma) {
  d1 <- (log(S/K) + (b+sigma^2/2)*T)/(sigma*sqrt(T))
  d2 <- d1 - sigma*sqrt(T)

  if(PutCall == 'c')
    px <- S*exp((b-r)*T)*pnorm(d1) - K*exp(-r*T)*pnorm(d2)
  else
    px <- K*exp(-r*T)*pnorm(-d2) - S*exp((b-r)*T)*pnorm(-d1)

px
}
```

GBSVega: Vega ($\frac{\partial P}{\partial \sigma}$) of a Vanilla option:

```
GBSVega <- function(PutCall, S, K, T, r, b, sigma) {
  d1 <- (log(S/K) + (b+sigma^2/2)*T)/(sigma*sqrt(T))
  S*exp((b-r)*T) * dnorm(d1)
}
```

Volatility Interpolation

Given the implied volatility at three strikes, we will use the Vanna-Volga pricing method to interpolate the volatility curve. Assume $r = 0, b = 0, T = 1, \text{Spot} = 100$.

```
# Benchmark data: (strike, volatility)
VolData <- list(c(80, .32), c(100, .30), c(120, .315))
```

Let's first define an array of pricing functions for the benchmark instruments:

```
C1 <- function(vol=sigma, spot=Spot) GBSPrice(PutCall='c', S=spot, K=VolData[[1]][1], T=T, r=r, b=b, si

C2 <- function(vol=sigma, spot=Spot) GBSPrice(PutCall='c', S=spot, K=VolData[[2]][1], T=T, r=r, b=b, si

C3 <- function(vol=sigma, spot=Spot) GBSPrice(PutCall='c', S=spot, K=VolData[[3]][1], T=T, r=r, b=b, si

C <- c(C1, C2, C3)
```

1. Write a utility functions to compute the risk indicators, all by finite difference:

```
Vega <- function(f, vol, spot=Spot) {
  d_vol <- 10e-5
  return( (f(vol+d_vol*vol, Spot)-f(vol-d_vol*vol, Spot))/(2*d_vol) )
}

Vanna <- function(f, vol, spot=Spot) {
  d_vol <- 10e-5
  d_spot <- 10e-5
  return( (f(vol+d_vol*vol, Spot+d_spot*Spot)+
           f(vol-d_vol*vol, Spot-d_spot*Spot)-
           f(vol+d_vol*vol, Spot-d_spot*Spot)-
           f(vol-d_vol*vol, Spot+d_spot*Spot))/(4*d_vol*d_spot) )
}

Volga <- function(f, vol, spot=Spot) {
  d_vol <- 10e-5
  return( (f(vol+d_vol*vol, Spot) - 2*f(vol, Spot) + f(vol-d_vol*vol, Spot))/d_vol**2 )
  }
```

Then, the calculation of vega for the three benchmark options may be performed by:

```
r<-0
b<-0
T<-1
Spot <- 100
B.vega <- sapply(1:3, function(i) Vega(C[[i]], VolData[[i]][2]))
```

2. Compute vectors of vega, vanna, volga for the three hedge instruments

```
r<-0
b<-0
T<-1
Spot <- 100
B.vega.benchmark <- sapply(1:3, function(i) Vega(C[[i]], VolData[[i]][2]))
B.vanna.benchmark <- sapply(1:3, function(i) Vanna(C[[i]], VolData[[i]][2]))
B.volga.benchmark <- sapply(1:3, function(i) Volga(C[[i]], VolData[[i]][2]))
B.vega.benchmark
```

```
## [1]  8.840076 11.834380 11.499491
```

```r
B.vanna.benchmark
```

```
## [1] -14.84369   5.91719  26.87955
```

```r
B.volga.benchmark
```

```
## [1]  4.0722703 -0.2662745  3.5671601
```

3. Choose a new strike for which we want to compute the implied volatility. Let's choose $K = 110$.
4. Compute the risk indicators for a call option struck at that strike.

```r
r<-0
b<-0
T<-1
Spot <- 100
K <- 110

#interpolation quadratic
strike.square <- c(VolData[[1]][1]^2, VolData[[2]][1]^2, VolData[[3]][1]^2)
strike <- c(VolData[[1]][1], VolData[[2]][1], VolData[[3]][1])
vol.data <- c(VolData[[1]][2], VolData[[2]][2], VolData[[3]][2])
output =  lm(vol.data~strike+strike.square)

vol.interpolation <- function(k= K){
  return(0.75-8.875e-03*k+4.375e-05*k^2)
}

VolData.ATM <- vol.interpolation(K)
##############
f <- function(vol=VolData.ATM, spot=Spot){
  GBSPrice(PutCall='c', S=spot, K=K, T=T, r=r, b=b, sigma=vol)
}
B.vega <- Vega(f, VolData.ATM, Spot)
B.vanna <- Vanna(f, VolData.ATM, Spot)
B.volga <- Volga(f, VolData.ATM, Spot)
B.vega
```

```
## [1] 11.93362
```

```r
B.vanna
```

```
## [1] 18.34531
```

```r
B.volga
```

```
## [1] 0.9056663
```

```r
b <- c(B.vega, B.vanna, B.volga)
```

5. Compute the Vanna-Volga price adjustment and the corresponding implied volatility.

```r
A <- matrix(data = c(B.vega.benchmark, B.vanna.benchmark, B.volga.benchmark), nrow =3)
X <- solve(A, b)
print("Matrice A =")
```

```
## [1] "Matrice A ="
```

```r
print(A)
```

```
##           [,1]      [,2]       [,3]
## [1,]  8.840076 -14.84369  4.0722703
## [2,] 11.834380   5.91719 -0.2662745
## [3,] 11.499491  26.87955  3.5671601
```

```r
print("Risk indicators (b) = ")
```

```
## [1] "Risk indicators (b) = "
```

```r
print(b)
```

```
## [1] 11.9336183 18.3453055  0.9056663
```

```r
print("Weights =")
```

```
## [1] "Weights ="
```

```r
print(X)
```

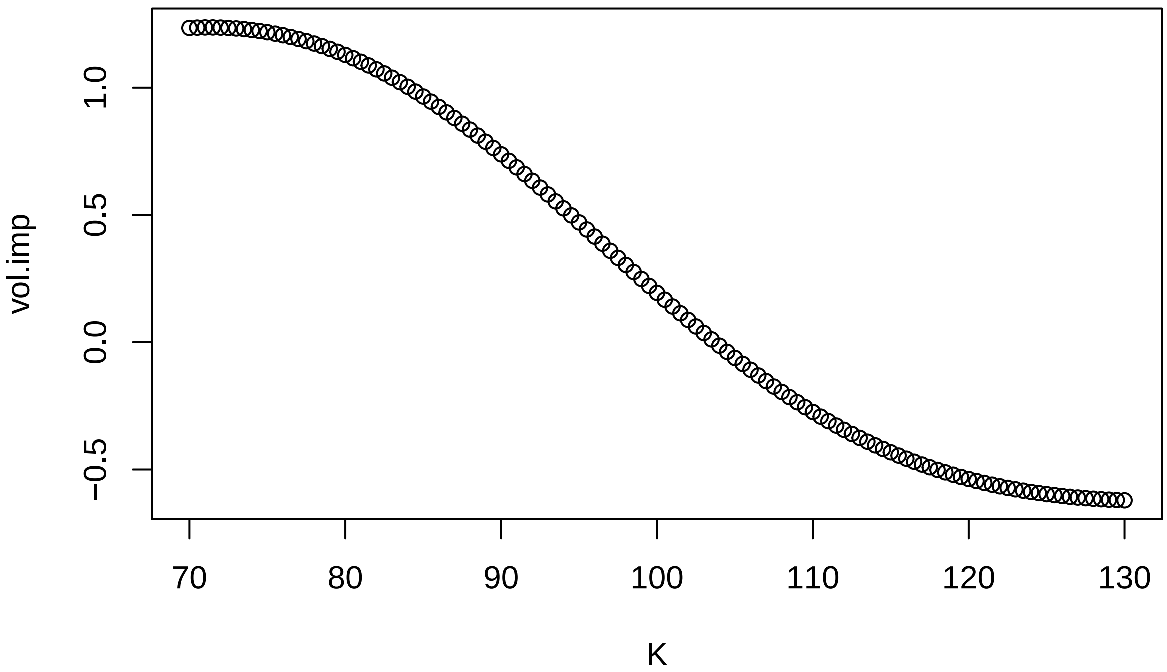
```
## [1]  1.7000563 -0.3995027 -2.2162341
```

```r
#C.BS <- C
#C.M <-

#somme <- 0
#for (i in 2:3) {
#  somme <- somme + X[i]*(C.M[i]-C.BS[i])
#}
#O.M <- O.BS + somme

vol.K <- VolData[[1]][2]*X[1] + VolData[[2]][2]*X[2] + VolData[[3]][2]*X[3]
vol.K
```

```
## [1] -0.2739465
```

6. Wrap the above logic in a function in order to interpolate/extrapolate the vol curve from $K = 70$ to $K = 130$

```r
r<-0
b<-0
T<-1
Spot <- 100
K <- seq (70, 130 ,0.5)
vol.imp <- c()


f <- function(vol, spot=Spot, k){
  GBSPrice(PutCall='c', S=spot, K=k, T=T, r=r, b=b, sigma=vol)
}

Vega <- function(f, vol, spot=Spot, k) {
  d_vol <- 10e-5
  return( (f(vol+d_vol*vol, Spot, k)-f(vol-d_vol*vol, Spot, k))/(2*d_vol) )
}

Vanna <- function(f, vol, spot=Spot, k) {
  d_vol <- 10e-5
  d_spot <- 10e-5
  return( (f(vol+d_vol*vol, Spot+d_spot*Spot, k)+
           f(vol-d_vol*vol, Spot-d_spot*Spot, k)-
           f(vol+d_vol*vol, Spot-d_spot*Spot, k)-
           f(vol-d_vol*vol, Spot+d_spot*Spot, k))/(4*d_vol*d_spot) )
}

Volga <- function(f, vol, spot=Spot, k) {
  d_vol <- 10e-5
  return( (f(vol+d_vol*vol, Spot, k) - 2*f(vol, Spot, k) + f(vol-d_vol*vol, Spot, k))/d_vol**2 )
  }

vol.f <- function(f, vol = VolData.ATM, spot=Spot, k) {
  b = c(1:3)
  b[1] = Vega(f, vol, spot=Spot, k)
  b[2] = Vanna(f, vol, Spot, k)
  b[3] = Volga(f, vol, Spot, k)
  A = matrix(data = c(B.vega.benchmark, B.vanna.benchmark, B.volga.benchmark), nrow =3)
  X = solve(A, b)
  return(VolData[[1]][2]*X[1] + VolData[[2]][2]*X[2] + VolData[[3]][2]*X[3])
}


for (i in 1:121) {
  vol = vol.interpolation(K[i])
  vol.imp = append(vol.imp, vol.f(f, vol = vol,spot = Spot, k = K[i]))
}

plot(K, vol.imp)
```



Pricing a digital call

Recall that a digital call with strike K pays one euro if $S_T \geq K$, and nothing otherwise.

Using the same logic as in the previous question, price a digital call, maturity $T = 1$, struck at $K = 105$.