Finance Quantitative

Valorisation d'un produit structuré
de type "shark"

Patrick Hénaff

Version: 30 mars 2023

```
library(xts)
library(hornpa)
library(lubridate)
library(xtable)
library(quantmod)
library(PerformanceAnalytics)
library(TTR)
library(lubridate)
library(roll)
library(Hmisc)
library(nFactors)
library(kableExtra)
library(broom)
library(fOptions)

get.src.folder <- function() {
  path.expand("../GP/src")
}

get.data.folder <- function() {
  path.expand("../GP/data")
}

source(file.path(get.src.folder(), 'utils.R'))
source(file.path(get.src.folder(), 'FileUtils.R'))
```

Le produit "Shark"

On se propose de valoriser un produit structuré de type "Shark", décrit dans la note d'information "EMTN Shark Eurostoxx50 Janvier 2016".

A maturité (3 ans) l'acheteur recevra le remboursement suivant, fonction de la valeur de l'indice sous-jacent S_t :

- Si $S_t > 125$ le client reçoit 105.
- Si $100 \leq S_t \leq 125$, le client reçoit S_t.
- Si $S_t < 100$, le client reçoit 100.

```
shark.payoff <- function(S) {
  res <- numeric(length(S))
  indx <- S>125
  if(sum(indx)>0) res[indx] <- 105
  indx <- (S>100) & (S <= 125)
  if(sum(indx)>0) res[indx] <- S[indx]
  indx <- (S <= 100) & (S > K.min)
  if(sum(indx)>0) res[indx] <- 100
  indx <- (S <= K.min)
  if(sum(indx)>0) res[indx] <- S[indx]
  res
  }
```

Questions

L'objectif de ce travail est de déterminer la valeur de ce produit structuré.

Valorisation analytique

Déterminer un portefeuille composé de call Européen et d'options digitales qui reproduit le payoff du produit structuré.

Le produit structuré Européen est un portefeuille composé de : - +1 call de strike 100 - -1 call de strike 125 - −20 call binaire de strike 125 - 1 zero-coupon de nominal 100 et maturité 3 ans

```
Digital <- function(TypeFlag, K, sigma, r, d, S0, TTM) {
  d <- (log(S0/K) + (r-d-sigma^2/2)*TTM)/(sigma*sqrt(TTM))
  exp(-r*TTM)*ifelse(TypeFlag=="c", pnorm(d), 1-pnorm(d))
}

shark.price <- function(S.0) {
  X <- numeric(3)
  X[1] <- GBSOption(TypeFlag="c", S=S.0, X=100, Time=TTM, r=r, b=r-div, sigma=sigma)@price
  X[2] <- -GBSOption(TypeFlag="c", S=S.0, X=125, Time=TTM, r=r, b=r-div, sigma=sigma)@price
  X[3] <- -20 * Digital("c", 125, sigma, r, div, S.0, TTM)
  sum(X)
}
```

Valoriser ce portefeuille avec les données de marché ci-dessous.

```
S.0 <- 100
r <- 0.02
div <- 0.03
TTM <- 3
sigma <- 0.2
K.min <- 0
```

```
P <- shark.price(S.0)
ZC <- 100*exp(-r*TTM)
```

```
## [1] "option:  3.14 ZC:  94.18 Total:  97.31"
```

Valorisation par arbre binomial

— Ecrire une fonction qui calcule le payoff du produit structuré à maturité.
— Utilser cette fonction pour valoriser le produit structuré dans un arbre binomial, avec les mêmes données de marché que plus haut.

Définissons un modèle binomial capable de valoriser un produit dérivé quelconque, fonction d'un seul sous-jacent :

```
ArbreBinomial <- function(payoff, Ex.type="EU", S.0, TTM, T.Ex=0, r, div, sigma, n) {
  # time step
  dt = TTM / n

  u = exp(sigma * sqrt(dt))
  d = 1 / u
  p = (exp((r - div) * dt) - d) / (u - d)
  # discount factor
  Df = exp(-r * dt)

  ST = S.0 * u ^ (0:n) * d ^ (n:0)
  option.value = payoff(ST)

  for (j in seq(from = n - 1, to = 0, by = -1)) {
    option.value = (p * option.value[2:(j + 2)] +
                            (1 - p) * option.value[1:(j + 1)]) * Df

    if((Ex.type == "US") & (j*dt > T.Ex)) {
      ST = S.0 * u ^ (0:j) * d ^ (j:0)
      ex.value <- payoff(ST)
      option.value <- pmax(option.value, ex.value)
    }
  }
  option.value[1]
}
```

A titre de vérification, vous pouvez calculer le prix d'un call Européen à l'aide de l'arbre et avec une formule analytique.

```
call.payoff <- function(S)     pmax(S-Strike, 0)
digital.payoff <- function(S) (S>Strike)*1

for(Strike in c(100, 125)) {
P.1 <- GBSOption('c', S.0, Strike,
                              TTM, r, b=r-div,sigma=sigma)@price
P.2 <- ArbreBinomial(call.payoff, "EU", S.0, TTM, 1,r, div, sigma, n=100)

print(paste('Prix du call: BS:', round(P.1,2),
            'ArbreBinomial', round(P.2,2)))
}
```

```
## [1] "Prix du call: BS: 11.41 ArbreBinomial 11.38"
## [1] "Prix du call: BS: 4.83 ArbreBinomial 4.84"
```

```
Strike <- 125
P.1 <- 20 * Digital("c", Strike, sigma, r, div, S.0, TTM)
P.2 <- 20*ArbreBinomial(digital.payoff, "EU", S.0, TTM, NA, r, div, sigma, n=100)
print(paste('Prix du call Digital: BS:', round(P.1,2),
            'ArbreBinomial', round(P.2,2)))
```

```
## [1] "Prix du call Digital: BS: 3.45 ArbreBinomial 3.17"
```

Valoriser la structure "Shark" dans l'arbre binomial.

```
TTM <- 3
K.min <- 0
P.Shark.binomial <- ArbreBinomial(shark.payoff, "EU", S.0, TTM, NA, r, div, sigma, n=200)
ZC <- 100*exp(-r*TTM)

print('Shark Euro par méthode binomiale')
```

```
## [1] "Shark Euro par méthode binomiale"
```

```
print(paste('Option:', round(P.Shark.binomial-ZC,2), 'ZC:', round(ZC,2),
            'Prix total:', round(P.Shark.binomial,2)))
```

```
## [1] "Option: 3.07 ZC: 94.18 Prix total: 97.25"
```

Shark Américain

On se propose de modifier le produit structuré pour qu'il puisse être exercé à tout moment à partir de la fin de la deuxième année. On souhaite que la valeur actualisée du produit reste le même que dans le cas initial. Pour cela, on envisage d'ajouter une clause à la formule de valeur d'exercise. Si $S_t < K_{\min}$, alors la valeur d'exercise est S_t, c'est à dire que le capital n'est plus garanti.

Déterminer la valeur de $K_{\min}$, en utilisant un arbre binomial ou la méthode "LS Monte Carlo".

Arbre binomial

On determine par approximations successives la valeur de $K_{\min}$, qui doit être à environ 73 pour conserver la valeur de la structure.

```
TTM <- 3
K.min <- 73
T.Ex <- 2
P.Shark.binomial <- ArbreBinomial(shark.payoff, "US", S.0, TTM, T.Ex,r, div, sigma, n=100)

print('Shark Américain par méthode binomiale')
```

```
## [1] "Shark Américain par méthode binomiale"
```

```
print(paste('Prix total:', round(P.Shark.binomial,2)))
```

```
## [1] "Prix total: 97.66"
```

On utilise l'algorithme de Longstaff et Schwartz pour la valorisation d'options Américaines par simulation. La fonction ci-dessous teste deux modèles de regression : une base de polynomes d'ordre 2, et une regression locale sur une base de splines, qui se révèle très supérieur. Cette expérimentation souligne l'importance d'un bon modèle de régression dans la méthode LSM.

```
library(stats)
American.LSM <- function(payoff, S0 = 100, sigma = 0.2,
                         nb.paths = 1000,
                         nb.steps = 100,
                         r=0, div=0, TTM = 1, T.Ex=NA,
                         fit=c("poly", "spline")) {

  fit <- match.arg(fit)
  # one time step
  dT <- TTM/nb.steps
  # discount factor for one time step
  df <- exp(-r*dT)

  # simulated paths for underlying asset
  S <- t(GBMPathSimulator(nb.steps, nb.paths, S0, r-div, sigma, TTM,
                          center='Mean'))

  # matrix of optimal cash flow per path
  CF <- matrix(0, nrow=nb.paths, ncol=(nb.steps+1))

  # continuation value at last time step is exercice value
  CF[,(nb.steps+1)] <- payoff(S[,(nb.steps+1)])

  t.last <- ifelse(is.na(T.Ex), 1, round(T.Ex/dT))
  for (t in nb.steps:t.last) {
    # vector of discount factors for future time steps
    df.vector <- matrix(exp(-r*dT*seq(1,(nb.steps+1-t))), ncol=1)
    # discounted continuation value
    disc.cont.value <- CF[,(t+1):(nb.steps+1)] %*% df.vector
    # indep variables:
    # constant, X, X^2, X^3
    exercice.value <- payoff(S[,t])
    # restrict regression to in-the-money paths at time t
    indx <- exercice.value>0
    if(any(indx)) {
      x <- exercice.value[indx]
      y <- disc.cont.value[indx]
      if((t == nb.steps) | (fit == "poly")) {
      X <- cbind(x,x^2,x^3)
      reg <- lm(y ~ X)
      fitted <- reg$fitted.values
      } else {
        res <- smooth.spline(x, y)
        fitted <- predict(res, x)$y
      }

    # determine paths where it's optimal to exercice at time t
    exercice.now <- exercice.value[indx] > fitted
```

```
    # update optimal cash flow
    update.cf <- vector(length=nb.paths)
    update.cf[indx] <- exercice.now
    CF[update.cf,t] <- exercice.value[update.cf]
    CF[update.cf,(t+1):(nb.steps+1)] <- 0
    }
    }

  # discounted optimal cash flow along each path
  disc.vector <- matrix(exp(-r*seq(0,TTM,length.out=(nb.steps+1))),
                        nrow=(nb.steps+1), ncol=1)
  price <- mean(CF %*% disc.vector)
  list(price=price, CF=CF, S=S)
  }
```

Cette fonction utilise un simulateur de chemins log-normaux :

```
GBMPathSimulator <-
  function(nbObs, nbSim, S0, mu, sigma, TTM, center = 'Mean') {
    # 1 - generate nbObs * nbSim Z ~ N(0,1) numbers
    #       make sure E(Z(t)) = 0 for all t
    #       a) by centering the data
    #       b) with antithetic variates,

    delta.t <- TTM / nbObs

    if (center == 'Mean') {
      Z <- rnorm(nbObs * nbSim,mean = 0,sd = 1)
      dim(Z) <- c(nbObs, nbSim)
      Z <- Z - mean(Z)
    } else {
      Z <- rnorm(nbObs * nbSim / 2,mean = 0,sd = 1)
      dim(Z) <- c(nbObs, nbSim / 2)
      Z <- cbind(Z,-Z)
    }

    # 2 - generate paths S(i,t)

    path = (mu - sigma * sigma / 2) * delta.t + sigma * sqrt(delta.t) * Z
    S <- S0 * rbind(rep(1, nbSim), exp(apply(path, 2, cumsum)))
    S
  }
```

A titre de vérification, valorisons un call Américain avec l'algorithme de Longstaff-Schwartz :

```
res <- American.LSM(call.payoff, S0 = S.0, sigma = sigma,
                            nb.paths = 50000,
                            nb.steps = 100, r = r, div = div,
                            TTM= TTM, )

p = ArbreBinomial(call.payoff, "US",  S=S.0, TTM, T.Ex=0, r=r, div=div,
                  sigma=sigma, n=100)
```

Prix d'un call Américain : LSM : 5.03, Binomial : 4.98.

Valorisation du produit “Shark”

Le produit “Shark” est finalement valorisé par :

```
res.poly <- American.LSM(shark.payoff, S0 = 100, sigma = sigma,
                    nb.paths = 10000,
                    nb.steps = 100, r = r, div = div,
                    TTM = TTM, T.Ex=2, fit="poly")
res.sp <- American.LSM(shark.payoff, S0 = 100, sigma = sigma,
                    nb.paths = 10000,
                    nb.steps = 100, r = r, div = div,
                    TTM = TTM, T.Ex=2, fit="spline")
```

Valorisation du “Shark Américain” :

- avec une regression polynomiale : 94.12
- avec une regression locale : 96.54
- avec un arbre binomial : 97.66