MASTER IN DATA SCIENCE FOR ECONOMICS, BUSINESS, AND FINANCE -- A.A. 2022/23
Assignment in Economic and Financial Data Science: modulo 1
Gruppo 2 - Studenti: Luca Garbin, Laura Proto, Andrea Mentasti, Giacomo de Gioia

---

Problema 1 -- Premessa teorica.

Un'*opzione europea di tipo call* è uno strumento finanziario derivato che dà diritto al suo possessore (*buyer*) di acquistare un certo asset, detto *sottostante*, a un certo prefissato prezzo, detto *strike*, al tempo T (scadenza del contratto). Trattandosi di diritto di acquistare, il possessore non ha alcun obbligo, pertanto il diritto, cioè l'opzione, verrà esercitata solo in condizione di convenienza.
Il *payoff* di un derivato non è altro che il suo valore di mercato alla scadenza T, e corrisponde al bilancio delle prestazioni e controprestazioni previste dal contratto a quell'epoca, solitamente rappresentato graficamente come funzione del sottostante.
In particolare il payoff di una call con posizione lunga (gergo tecnico per indicare il buyer dell'opzione) è, per evidenti condizioni di convenienza:

$$\pi_T = max\{S_T - K, 0\}$$

S_T : sottostante valutato all'epoca T,
K : strike.

Una *strategia butterfly* è un derivato risultante dalla composizione di altri derivati, precisamente nel seguente modo:

- J posizioni lunghe di una call con strike K_1;
- H posizioni lunghe di una call con strike K_2;
- $J + H$ posizioni corte (venditore dell'opzione) con strike $\overline{K}$, dove quest'ultimo è un valore medio ponderato di K_1 e K_2 così definito:

$$\overline{K} = \omega K_1 + (1 - \omega) K_2 , \qquad con\ \omega = J/(J + H)$$

Il suo payoff si può ottenere come somma algebrica di payoff di opzioni call con posizione lunga:

$$\pi_T(butterfly) = \underbrace{Jmax\{S_T - K_1, 0\}}_{long\ call} \underbrace{- \underbrace{(J + H)max\{S_T - \overline{K}, 0\}}_{long\ call}}_{short\ call\ (come\ def.\ sopra)} + \underbrace{Hmax\{S_T - K_2, 0\}}_{long\ call}$$

Problema 1 -- Codice R commentato e risultati.

Funzioni implementate in R:

- nome del file "Gruppo2_assignment_funzioni.R"

```r
#------------#
# Problema 1  #
#------------#

long_call_payoff = function(S, K) {
  return(max(S - K, 0))
}

butterfly_payoff = function(S, K1, K2, J, H) {
  omega <- J/(J+H) # peso della prima call
  KK <- omega*K1 + (1-omega)*K2 # strike medio (short call della strategia)
  return (J*long_call_payoff(S, K1) - (J+H)*long_call_payoff(S, KK)+ H*long_call_payoff(S, K2))
}

```

payoff di una call con posizione lunga

payoff di una butterfly

- nome del file "Gruppo2_assignment_1.R" che richiama le funzioni di cui sopra:

```r
source("Gruppo2_assignment_funzioni.R")

sottostante <- 70
strike <- 75

long_call_payoff(S=sottostante, K=strike)

sottostante <- 80
strike <- 75

long_call_payoff(S=sottostante, K=strike)

```

primo esempio numerico di payoff della call lunga

secondo esempio numerico di payoff della call lunga

- Risultati dei due esempi numerici:

```
> sottostante <- 70
> strike <- 75
>
> long_call_payoff(S=sottostante, K=strike)
[1] 0
>
> sottostante <- 80
> strike <- 75
>
> long_call_payoff(S=sottostante, K=strike)
[1] 5
>
```

Nel primo esempio si osserva che essendo lo strike (75) superiore al valore del sottostante (70), naturalmente l'opzione call lunga non viene esercitata, quindi correttamente payoff = 0; nel secondo esempio invece lo strike (sempre 75) è questa volta inferiore al valore del sottostante (80), pertanto è conveniente esercitare l'opzione che darà un payoff pari alla differenza = 5.

- Codice per il plot del payoff della strategia butterfly:

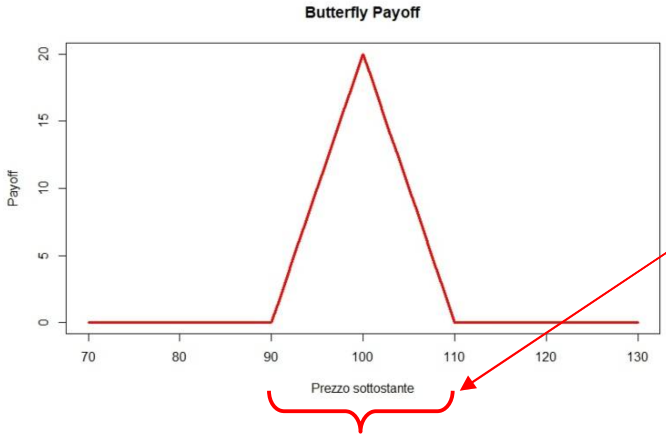
```r
range_min <-70
range_max <- 130
range <- seq(range_min, range_max, by=1)
sottostante <- range #vettore di valori del prezzo dell'asset sottostante
strike_1 <- 90 #strike price della prima call option
strike_2 <- 110 #strike price della seconda call option
long_call_1 <- 2 #posizioni lunghe nella prima call option (strike K1)
long_call_2 <- 2 #posizioni lunghe nella seconda call option (strike K2)

payoff <- range # inizializzazione
for (s in range){
  payoff[s-range_min+1] <- butterfly_payoff(S=s, K1=strike_1, K2=strike_2, J=long_call_1, H=long_call_2)
}

plot(x=sottostante, y=payoff, type = "l", xlab = "Prezzo sottostante", ylab = "Payoff",
     col = "red", lwd = 3, main = "Butterfly Payoff")

```

- Un esempio di plot:



Range di payoff positivo, con , in questo caso, $\overline{K}$ coincidente col punto medio tra i due strike K_1 e K_2. Chi persegue una tale strategia si aspetta di osservare a scadenza un prezzo del sottostante in un intorno di $\overline{K}$ contenuto in $[K_1, K_2]$.

Problema 2 -- Premessa teorica.

A differenza della call vista sopra, l'opzione europea di tipo *put* garantisce al possessore il diritto di vendere il sottostante alla scadenza T al prezzo strike K.
Valutiamo questa opzione col metodo dell'albero binomiale uniperiodale. Il metodo consiste nel calcolare il valore atteso scontato al tempo iniziale, sotto specifiche misure di probabilità dette *risk-neutral* o di martingala, di un portafoglio di replica dell'opzione, costituito da due titoli, uno rischioso e l'altro privo di rischio, presi nelle opportune proporzioni. L'esistenza (e l'unicità) di un tale portafoglio è garantito dal sistema di equazioni lineari con matrice dei coefficienti non singolare che occorre impostare per risolvere il problema della ricerca di tale portafoglio.
In sintesi:

$$\underbrace{\begin{bmatrix} B_{t_0}(1+i)^{T-t_0} & S_{t_0}u \\ B_{t_0}(1+i)^{T-t_0} & S_{t_0}d \end{bmatrix}}_{P_T} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} \phi(S_{t_0}u) \\ \phi(S_{t_0}d) \end{bmatrix}$$

con significato dei simboli:

B_{t_0} : titolo non rischioso (o deposito bancario) all'epoca t_0;

S_{t_0} : titolo rischioso all'epoca t_0;

$(1+i)^{T-t_0}$: fattore di capitalizzazione composta tra t_0 e T al tasso i (privo di rischio);

u, d : rispettivamente fattore di scenario rialzista e ribassista, con rispettive probabilità cd *fisiche* p_u e $p_d = 1 - p_u$, e soggetti alla *condizione di non arbitraggio* $d < (1+i)^{T-t_0} < u$;

ϕ : funzione di payoff;

x, y : le incognite del problema, rispettivamente quantità del titolo non rischioso e rischioso nel portafoglio di replica.

Svolgendo si ottiene:

$$P_T^{-1} = \frac{1}{B_{t_0}(1+i)^{T-t_0}\, S_{t_0}\,[d-u]} \begin{bmatrix} S_{t_0}d & -S_{t_0}u \\ -B_{t_0}(1+i)^{T-t_0} & B_{t_0}(1+i)^{T-t_0} \end{bmatrix}$$

da cui:

$$\begin{bmatrix} x \\ y \end{bmatrix} = P_T^{-1} \begin{bmatrix} \phi(S_{t_0}u) \\ \phi(S_{t_0}d) \end{bmatrix}$$

A questo punto il prezzo al tempo t_0 del derivato è pari al valore di mercato in t_0 del ptf replicante:

$$\phi_{t_0} = xB_{t_0} + yS_{t_0} = sostituendo = \frac{1}{(1+i)^{T-t_0}} \left[q_u\, \phi(S_{t_0}u) + q_d\, \phi(S_{t_0}d) \right]$$

che rappresenta la cd *pricing formula*, nella quale:

$q_u = \frac{(1+i)^{T-t_0}-d}{u-d}$, $q_d = 1 - q_u$, rispettivamente probabilità risk-neutral degli scenari u e d.

Nell'esercizio utilizzeremo anche la cd *put-call parity* e faremo una verifica delle cd *condizioni di Merton* (v. nel codice R per dettagli).

Problema 2 -- Codice R commentato e risultati.

- nome del file "Gruppo2_assignment_funzioni.R"

```r
BM_Put <- function(S0, u, d, K, TGrande=1, t0=0, r=0){

  Time2Mat <- TGrande-t0

  P <- matrix(c((1+r)^(Time2Mat),(1+r)^(Time2Mat), S0*u, S0*d),2,2)
  InverseP <- solve(P)

  quantita_asset <- InverseP %*% matrix(c(max(c(K-S0*u,0)), max(c(K-S0*d,0))),2,1)
  y <- quantita_asset[2,1]
  qu <- ((1+r)^Time2Mat-d)/(u-d)

  Putprice <- (1/(1+r)^Time2Mat)*(qu*max(K-S0*u,0)+(1-qu)*max(K-S0*d,0))

  return(list(prezzo_put=Putprice, quantita_risky=y, prob_risk_neutral=c(qu=qu, qd=1-qu)))

}

Callprice <- function(Put, S0, K, fatt_cap) Put + S0 - K / fatt_cap
```

implementazione della pricing formula

implementazione della put-call parity che permette di passare dal pricing di una put a quello di una call, e viceversa, senza necessariamente dover fare assunzioni sull'andamento futuro del sottostante, ma semplicemente utilizzando delle quantità economiche note al tempo t di valutazione (per ogni epoca t) come il valore del sottostante e il tasso privo di rischio:

$$c_t(S_t, K, T) - p_t(S_t, K, T) = S_t - K/(1+i)^{T-t}$$

- nome del file "Gruppo2_assignment_2.R" che richiama le funzioni di cui sopra:

```r
source("Gruppo2_assignment_funzioni.R")

sottostante <- 100
strike <- 110
epoca_iniziale <- 0
scadenza <- 1
tasso_cap_comp <- 0.05
fatt_cap = (1+tasso_cap_comp)^(scadenza-epoca_iniziale)
rialzista <- 1.10*fatt_cap
ribassista <- 0.8*fatt_cap

res <- BM_Put(S0=sottostante, u=rialzista, d=ribassista, K=strike, TGrande=scadenza,
              t0=epoca_iniziale, r=tasso_cap_comp)

res$prezzo_put   # esempio numerico
res$quantita_risky # esempio numerico
res$prob_risk_neutral # esempio numerico

#--------------
Call_price <- Callprice(Put=res$prezzo_put, S0=sottostante, K=strike, fatt_cap=fatt_cap)
Call_price # esempio numerico

sottostante
Call_price
max(sottostante-strike/fatt_cap, 0)

if ((sottostante >= Call_price)&(Call_price>=max(sottostante-strike/fatt_cap, 0))){
  print("Le condizioni di Merton sono verificate")
}
```

esempio numerico di valutazione di una put europea

funzione che a sua volta richiama la funzione sopra

esempio numerico di valutazione di una call europea

verifica dei vincoli di Merton, per una call:

$$S_t \geq c_t(S_t, K, T) \geq max\{S_t - K/(1+i)^{T-t}, 0\}$$

- Risultati degli esempi numerici sopra:

```r
> res$prezzo_put  # esempio numerico
[1] 8.253968
> res$quantita_risky # esempio numerico
[1] -0.8253968
> res$prob_risk_neutral # esempio numerico
       qu        qd
0.6666667 0.3333333
> #--------------
> Call_price <- Callprice(Put=res$prezzo_put, S0=sottostante, K=strike, fatt_cap=fatt_cap)
> Call_price # esempio numerico
[1] 3.492063
> sottostante
[1] 100
> Call_price
[1] 3.492063
> max(sottostante-strike/fatt_cap, 0)
[1] 0
> if ((sottostante >= Call_price)&(Call_price>=max(sottostante-strike/fatt_cap, 0))){
+   print("Le condizioni di Merton sono verificate")
+ }
[1] "Le condizioni di Merton sono verificate"
> 
```

prezzo della put

quantità del titolo rischioso: la quantità è negativa perché segnala un'operazione di presa in prestito del titolo rischioso, con finalità di vendita allo scoperto

probabilità risk-neutral

prezzo della call calcolato mediante la put-call parity: notare che deve essere $c_t - p_t = S_t - K/(1+i)^{T-t}$, e infatti: 3.492063-8.253968 = 100-110/1.05 = -4.7619

da notare che i vincoli di Merton sono verificati, infatti:
$c_t = 3.492063 < S_t = 100$, e
$c_t > max\{S_t - K/(1+i)^{T-t}, 0\}$ che è = 0, essendo
$S_t - K/(1+i)^{T-t}$ negativo, come visto sopra (=-4.7619)

Problema 3 -- Premessa teorica.

Il modello dell'albero binomiale n-periodale per la valutazione di un'opzione non è altro che una generalizzazione di quello uniperiodale. Si fonda sulla suddivisione della durata residua dell'opzione in n periodi in ciascuno dei quali viene applicato, a ciascun nodo con i suoi rami, il modello uniperiodale visto sopra. Si noti che, fissata la durata residua, all'aumentare di n ($n \to +\infty$) gli intervalli di tempo si riducono, cioè l'ampiezza dei periodi $\to 0$, con la conseguenza di passare da un modello a tempo discreto (*Cox, Ross, Rubistein*) a un modello a tempo continuo (*Black and Scholes*). In questo esercizio mostreremo tale convergenza.

Valuteremo un'opzione di tipo put europeo. Per l'implementazione del codice in R abbiamo utilizzato i pacchetti `quantmod` e `fOptions`.

Problema 3 -- Codice R commentato e risultati.

- nome del file "Gruppo2_assignment_3.R"

```r

library("fOptions")
library("quantmod")
getSymbols("GOOG")
plot(GOOG)
plot(GOOG$GOOG.Close)
ClosePrice <- as.numeric(GOOG$GOOG.Close)
S_corrente <- tail(ClosePrice,1L)
Strike <- 1.05*S_corrente


logRendimenti<- diff(log(ClosePrice))
Sample2ndMom <- mean(logRendimenti^2)
Delta <- 1/360
sigma <- sqrt(1/Delta*Sample2ndMom)


i <- 0
r <- log(1+i)

TimeToMat <- 60/360
N_period <- 3

Put <- CRRBinomialTreeOption(TypeFlag ="pe", S=S_corrente, X=Strike,
                             Time=TimeToMat, r=r, b=r, sigma=sigma, n=N_period)
Put@price

#class(Put)
#slotNames(Put)
#str(Put)

```

scarichiamo la serie storica dei prezzi di un'azione (Google in questo esempio)

facciamo un plot dei prezzi di chiusura (v. sotto)

selezioniamo i prezzi di chiusura
e fissiamo l'ultimo prezzo della serie storica

fissiamo lo strike price, qui la condizione è "in the money"

procediamo alla stima campionaria dei log-rendimenti su base annua:

$$\hat{\sigma} = \sqrt{\frac{\frac{1}{N}\sum_{j=1}^{N}\left(\ln\left(\frac{quotazione_{t_j}}{quotazione_{t_{j-1}}}\right)\right)^2}{\Delta t}}$$

convertiamo il tasso d'int. composto in corrispondente composto continuo

fissiamo la scadenza dell'opzione a 60 gg e numero di periodi = 3

ipotesi dividend yield (o convenience yield) = 0

- grafico dei prezzi di chiusura di Google e risultato della valutazione:


```
> N_period <- 3
> Put <- CRRBinomialTreeOption(TypeFlag ="pe", S=S_corrente, X=Strike,
+                   Time=TimeToMat, r=r, b=r, sigma=sigma, n=N_period)
> Put@price
[1] 9.492612
>
```

- segue confronto tra il modello n-periodale di Cox, Ross, Rubistein e quello di Black and Scholes:

— nome del file "Gruppo2_assignment_funzioni.R":

```r

Put_Binomial_eval <- function (N_period){
  getSymbols("GOOG")
  ClosePrice <- as.numeric(GOOG$GOOG.Close)
  S_corrente <- tail(ClosePrice,1L)
  Strike <- 1.05*S_corrente
  logRendimenti<- diff(log(ClosePrice))
  Sample2ndMom <- mean(logRendimenti^2)
  Delta <- 1/360
  sigma <- sqrt(1/Delta*Sample2ndMom)
  i <- 0
  r <- log(1+i)
  TimeToMat <- 60/360
  #N_period <- 3
  Put <- CRRBinomialTreeOption(TypeFlag ="pe", S=S_corrente, X=Strike,
                               Time=TimeToMat, r=r, b=r, sigma=sigma, n=N_period)
  return(Put@price)
}

```

— nome del file "Gruppo2_assignment_3.R", ultima parte dello script:

```r

BS_eval <- GBSOption(TypeFlag ="p", S=S_corrente, X=Strike, Time=TimeToMat,
                     r=r, b=r, sigma=sigma)

#slotNames(BS_eval)
#str(BS_eval)

BS_eval@price

```

pacchetto di fOptions che implementa
il cd generalized Black-Scholes option

— nome del file "Gruppo2_assignment_3_plot.R":

```r
source("Gruppo2_assignment_funzioni.R")
source("Gruppo2_assignment_3.R")

N_max <- 30
Put_Bin <- rep(0, N_max)

for (N in c(1:N_max)){

  Put_Bin[N] <- Put_Binomial_eval(N_period=N)

}
Put_Bin
BS_eval <- rep(BS_eval@price, N_max)
BS_eval

plot(x=c(1:N_max), y=Put_Bin, type="l", col="red", lwd=3,
     xlab="Numero di periodi (fissata la durata residua)", ylab="Valutazione di una put (europea)",
     main="Convergenza del metodo Binomial Tree")
lines (BS_eval, type = "l", lty="dashed", lwd =3,col="blue")
legend(15,10.2,c("Binomial Tree", "Black and Scholes"), col=c("red","blue"),
       lty=c(1,3), title.font=2)


```

— confronto grafico: