UH Introduction to mathematical finance II, Exercise-7 (23.03.2016)

In all the exercises we consider random variables defined on a probability space $(\Omega, \mathcal{F})$ equipped with a probability measure $\mathbb{P}$ and a filtration $\mathbb{F} = (\mathcal{F}_t : t \in \mathbb{N})$, where $\mathcal{F}_s \subseteq \mathcal{F}_t$ for $s \leq t$.

1. Let τ be a stopping time in the filtration $\mathbb{F} = (\mathcal{F}_t)$. Show that $(\tau + 1)$ is a $\mathbb{F}$-predictable time, which means $\{\omega : \tau(\omega) \leq t\} \in \mathcal{F}_{t-1}\ \forall t$. Is $(\tau - 1)$ always $\mathbb{F}$-stopping time ?

2. Show that the process $(M_t : t = 0, 1, \ldots, T)$ is a $(P, \mathbb{F})$-martingale if and only if $E_P(M_\tau\ ) = E(M_0)$ for all $\mathbb{F}$-stopping times τ.

   Hints: $\Longrightarrow$ consider the stopped process $M_{t\wedge\tau}$. $\Longleftarrow$, for $s \leq t$ and $A \in \mathcal{F}_s$, consider the random time $\tau(\omega) = s\mathbf{1}_A(\omega) + s\mathbf{1}_{A^c}(\omega)$, show first that it is a $\mathbb{F}$-stopping time.

3. Let $(F_t : t \leq T)$ be the face value of an american option, and let U_t its value. Let also V_t be the value at time t of the european option F_T. Prove that $V_t \leq U_t\ \forall t = 0, 1, \ldots, T$.

4. We have shown that for an american call option of the form $F_t = (S_t - K)^+$, $k = 0, 1, \ldots, T$, the optimal exercise price is $\tau = T$ and the value U_t of the american option at time t coincides with the value of V_t of the european option $(S_T - K)^+$. Show that this is not true for the put option. In particular the call-put partity does not hold for american options.

5. We consider the Cox Ross Rubinstein binomial tree model, with assets $(B_t, S_t : t \in \mathbb{N})$, where $B_0 = S_0 = 1$

   $$B_t = B_{t-1}(1 + r), \quad S_t = S_{t-1}(1 + R_t), \quad t \geq 1$$

   where $R_t(\omega)$ are random variables with

   $$P(R_t = u|\mathcal{F}_{t-1})(\omega) = 1 - P(R_t = d|\mathcal{F}_{t-1})(\omega) \in (0, 1)$$

   and $-1 < d < r < u$ are deterministic. The filtration is $\mathbb{F} = (\mathcal{F}_t : t \in \mathbb{N})$ where $\mathcal{F}_t = \sigma(S_u : u \leq t)$. Recall that this market is arbitrage free and complete, and under the risk-neutral martingale measure Q, the random variables R_t are Q-independent and identically distributed with

   $$Q(R_t = u) = 1 - Q(R_t = d) = q = \frac{r - d}{u - d} \in (0, 1)\ .$$

In such Cox-Ross-Rubinstein binary tree model let the time horizon be $T = 2$, initial value $S_0 = 150$, $u = 2/5$, $d = -1/5$, $r = 1/10$ and the strike price $K = 140$.

(a) Compute on the binary tree the prices and of the american option with pays to the holder the face value

$$F_t = \max_{0 \le u \le t} (K - S_u)^+$$

at a $\mathbb{F}$-stopping time $\tau \le T$ chosen by the option holder.

(b) Compute also the hedging strategy for the option seller

(c) Compute the smallest and biggest optimal exercise times for the option holder.

You can use also use the octave/matlab computer software and modify the example codes provided below for the american put and call options.

6. We start to get familiar with the octave/matlab scientific computing language and with the financial package. Octave is free gnu-software for scientific computing. It is a clone of the commercial matlab software. In case you have access to matlab you can also use matlab.

   Download and install octave (the current version is 4.0.0) on your device following the instruction link `https://www.gnu.org/software/octave/download.html`

   There is also a free android version for tablets/phones downloadable from the play store.

   `https://play.google.com/store/apps/details?id=com.octave&hl=en`

   `https://play.google.com/store/apps/details?id=com.octave.financial&hl=en`

   For example in ubuntu you can install octave with these commands:

```
sudo apt-add-repository ppa:octave/stable

sudo apt-get update

sudo apt-get install octave
```

Download and install the octave financial package `http://octave.sourceforge.net/financial/` `http://octave.sourceforge.net/financial/overview.html`

For example in the ubuntu linux distribution:

```
sudo apt-get install liboctave-dev

sudo apt-get install octave-info

sudo apt-get install octave-financial

octave --force-gui

>> pkg install -forge financial

>> pkg load financial

>> K= 60; S0 = 50; r = 0.05, sigma= 0.2, Dt = 0.01; N = 100;

>> americancall = binprice(K,S0,r,sigma,Dt,T,1);

>> americanput= binprice(K,S0,r,sigma,Dt,T,1);

>> europeanput=americancall+K-S0;
```

The output returned is the asset price and American option value at each node of the binary tree.

The octave and matlab function binprice() computes American call and put option prices using the binomial tree CRR model.

```
[AssetPrice, OptionValue] =

binprice (Price, Strike, Rate, Time, Increment, Volatility, OptType)

>> help binprice

>> doc binprice
```

Since in the CRR model the european and american call option have the same price, you can use the same binprice formula together with

the call-put parity for european options to get the price of the european put.

Run the example

```
>>  option_example
```

downloadable from https://wiki.helsinki.fi/pages/viewpage.action?pageId=164343144&preview=/164343144/194680277/option_example.m

```
%%% octave 4.0.0  code  by Dario Gasbarra dario.gasbarra@helsinki.fi  21.3.2016
% uncomment to install the package if not installed
%pkg install -forge financial
 pkg load financial
 K= 60; S0 = 50; d=-0.1, r = 0.01, u=0.2, T = 10;
 [S,americancall, fv_americancall] = binpriceFacevalue(K,S0,d,r,u,T,0);
 [S,americanput, fv_americanput]= binpriceFacevalue(K,S0,d,r,u,T,1);
 europeanput=americancall+K-S0;
 crrgraph(americancall,fv_americancall);
 title('American call option prices and face value in the CRR binary tree model')
 figure
 crrgraph(americanput,fv_americanput);
 title('American put option prices and face value in the CRR binary tree model')
```

using the function crrgraph() downloadable from https://wiki.helsinki.fi/pages/viewpage.action?pageId=164343144&preview=/164343144/194680274/crrgraph.m

```
 octave 4.0.0 code  by Dario Gasbarra  dario.gasbarra@helsinki.fi    21.3.2016
function  crrgraph(option,face)
T=size(option,1);
N=T*(T+1)/2;
B=zeros(T,T);
XY=zeros(N,2);
last=0;
V=zeros(N,1);
F=zeros(N,1);
for  (i=1:T)
  idx=(last+1):(last+i);
  B(i,1:i)=idx;
  XY(idx,2)=i;
  XY(idx,1)=(1:i);
  V(idx)=option(1:i,i);
  F(idx)=face(1:i,i);
  last=last+i;
 endfor;
A=zeros(N,N);
for(i=1:(T-1))
for(j=1:i)
A( B(i,j),   [B(i+1,j) B(i+1,j+1)])=1 ;
endfor;
endfor;
A=A+A';

%XY(1,:)=XY(1,:)+T/2;
%XY(2,:)=T-XY(2,:);
gplot(A,XY,'-*');
axis("off")

dy=0.3
for(i=1:N)
text( XY(i,1),XY(i,2),num2str(V(i)));
text( XY(i,1),XY(i,2)+dy,num2str(F(i)));
endfor;
endfunction;
```

and the function binpriceFacevalue(), downloadable from https://wiki.helsinki.fi/pages/viewpage.action?pageId=164343144&preview=/164343144/194680273/binpriceFacevalue.m

```
%%% octave 4.0.0 code   by Dario Gasbarra dario.gasbarra@helsinki.fi  21.3.2016
%%% it gives also the face values of the american option

## This program is free software; you can redistribute it and/or modify it under
## the terms of the GNU Lesser General Public License as published by the Free
## Software Foundation; either version 3 of the License, or (at your option) any
## later version.
##
## This program is distributed in the hope that it will be useful, but WITHOUT
## ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or
## FITNESS FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License
## for more details.
##
## You should have received a copy of the GNU Lesser General Public License
## along with this program; if not, see <http://www.gnu.org/licenses/>.

## Computes the American call and put option prices using the
## Cox-Ross-Rubinstein binomial tree.
## @seealso {binprice, blkprice, blsprice}

function [AssetPrice, OptionValue, FaceValue] = binpriceFacevalue(Price, Strike,d,r,u,N,OptType )
  %%% OptType=0 for american call option,
  %%  OptType=1 for american put option
  %%% the function returns at each node of the CRR binary tree
  %%% the price of the asset AssetPrice
  %%% the price of the american option
  %%% and the face value of the option

  if ((d<= -1)||( r<=d)||(r>=u)||(u<d))
   fprintf('the model does not satisfy -1<d<r<u ')
  end

  if (nargin!=7)
    print_usage ();
  elseif (! isbool (OptType) && ! isequal (OptType, 0) && ...
          ! isequal (OptType, 1))
    error ("binprice: OPTTYPE must be logical, 0 (call) or 1 (put)");
  endif

  AssetPrice = OptionValue = FaceValue=zeros (N);

  ## Martingale measure
  a=(1+r);
  p = (r- d) / (u - d);
  q = 1 - p;

  ## Build tree
  AssetPrice(1, 1) = Price;
  for n = 2:N
    AssetPrice(1:(n - 1), n) = (1+u) * AssetPrice(1:(n - 1), n - 1);
    AssetPrice(n, n) = (1+d) * AssetPrice(n - 1, n - 1);
  endfor

  ## Initial condition
  opt = 2 * OptType - 1;
  FaceValue(:,N)=OptionValue(:, N) = max (opt * (AssetPrice(:, N) - Strike), 0);

  ## Time-stepping loop
  for n = (N - 1):-1:1
    HoldValue = (q * OptionValue(2:(n + 1), n + 1) + ...
                 p * OptionValue(1:n,       n + 1)) / a;
    FaceValue(1:n, n)=opt * (AssetPrice(1:n, n) - Strike);
    %OptionValue(1:n, n) = max (HoldValue, opt * (AssetPrice(1:n, n) - Strike));
   OptionValue(1:n, n) = max (HoldValue,FaceValue(1:n, n));
 endfor

endfunction

## Tests
%K= 60; S0 = 50; d=-0.1, r = 0.01, u=0.2, T = 10;
% [S,americancall, fv_americancall] = binpriceFacevalue(K,S0,d,r,u,T,0);
% [S,americanput, fv_americanput]= binpriceFacevalue(K,S0,r,sigma,Dt,T,1);
```