Corrigé

Université de Nice
Département de Mathématiques
NOM :
PRENOM :

L3MASS, année 2017-2018
Calcul stochastique et Finance
Date :
Salle :

Calcul Stochastique et applications à la finance
Feuille-réponses du TP 2
Calcul du prix d'un Call et d'un Put

L'objet de cette séance est de calculer dans un modèle de Cox-Ross-Rubinstein à n étapes le prix d'une option Call et celui d'une option Put de pay-off respectifs $\varphi(S) = (S-K)^+$ et $\psi(S) = (K-S)^+$. Dans un premier temps nous considèrons les Call et les Put à *la monnaie*, c'est-à-dire que nous supposons leur prix d'exercice K égal à S_0.

Rappelons que dans le modèle de Cox-Ross-Rubinstein, les prix de l'actif sous-jacent sont modélisés par une marche aléatoire S_t définie par $S_0 =$`S0` et $S_{t+\delta t} = S_t U_t$, avec $U_t \in \{$`up`$,$`down`$\}$. On posera

$n = 100$ $T = 1$ $\delta t = T/n$ `up` $= e^{+\sigma\sqrt{\delta t}}$ `down` $= e^{-\sigma\sqrt{\delta t}}$ $\sigma = 0.3$ $S_0 = 120$.

1. Définir, comme dans le TP1, la fonction $S(i,j)$ pour $i \in \{0,\dots,n\}$ et $j \in \{0,\dots,i\}$. On prendra $n = 100$, $\sigma = 0.3$ et $S_0 = 140$. Indiquer quelle valeur vous trouvez pour S à l'instant $t = 10\delta t$ s'il n'y a eu que 4 `down`. Expliquer.

$t = 10\delta t$ correspond à $i = 10$; s'il n'y a eu que 4 down c'est qu'il y a eu 6 up et donc $j = 6$. On trouve $S(10,6) = 127,42039$ si $S_0 = 120$ ($148,65$ si $S_0 = 140$)

2. Cette valeur de S dépend-elle du choix de n ? Expliquer.

Cette valeur dépend évidemment de la valeur de n, puisque up et down en dépendent via $\sqrt{\delta t} = \sqrt{\frac{T}{n}}$. Par exemple pour $n = 1000$, $S(10,6) = 122,2\dots$

3. Pour $n = 100$ indiquer quelle valeur vous trouvez pour S à l'instant $t = 0.5$ s'il n'y a eu que 4 `down`.

$t = 0,5 = \frac{T}{2} = \frac{n}{2}\frac{T}{n} = 50\delta t$, donc ici $i = 50$ et $S(50,46) = 423,05058$ si $S_0 = 120$ ($493,56$ si $S_0 = 140$)

4. On rappelle que le prix d'une option ayant pour date d'exercice T et pour payoff $\varphi(S_T)$ se calcule par récurrence rétrograde. Donc la première idée serait de définir, comme pour S une fonction $C(i,j)$. Nous verrons en fin de séance pourquoi ce n'est pas la bonne idée. Pour le Call, nous allons le coder dans Scilab comme une matrice triangulaire notée CC telle que `CC(1+i,1+j)`$=C(i,j)$ de taille $(n+1)\times(n+1)$.

On précise que $r = 0.10$

```
CC=zeros(n+1,n+1) ;
for j=0:n CC(1+n,1+j)=phi(S ( n,( j)) ;
end ;
for i=n-1:-1:0
      for j=0:i CC(1+i,1+j)=(p*CC(1+i+1,1+j+1)+(1-p)*CC(1+i+1,1+j))/R ;
      end ;
end ;
```

```
function pphi= phi(S)
  pphi = max(S-K,0)
endfunction;
```

Choisir K pour un Call *à la monnaie*, introduire les constantes $R = \exp(r\delta t)$ pour $r = 0.1$ et $p = (R-d)/(u-d)$ puis exécuter ce code pour calculer les valeurs `CC(i,j)`. Quelle est la prime du Call (sa valeur à l'instant de souscription du contrat) ? Vérifier, sur quelques valeurs, que le payoff du Call est bien celui que vous attendiez.

Prime $= C(0,0) =$ `CC(1+0,1+0)` $= CC(1,1) = 20,04527$
C'est pour $t = T$ (et donc $i = 50$) qu'on sait que $C_T = (S_T - K)^+$
Comme $K = 80$ on aurait `CC(1+100,j)` si $j \le 50$; `CC(101,:)` est bien nul jusqu'à 5 puis augmente

5. Quel est le prix du Call à la monnaie à la date $t = 4\delta t$ si le cours de l'actif sous jacent n'a fait qu'augmenter depuis la date de souscription du contrat ? Même question si le cours a baissé une fois exactement ?

$t = 4\delta t$ donc $i = 4$. Puis $j = i = 4$ puisque le cours a toujours augmenté
$C(4,4) = CC(1+4,1+4) = 31,030335$
Si une seule baisse $j = i-1 = 3$
$C(4,3) = CC(1+4,1+3) = 24,864474$

6. En vous inspirant du code précédent, que vous dupliquerez puis modifierez, calculer cette fois le prix $PP(1+i,1+j)$ d'un Put à la monnaie. Expliquer les lignes que vous avez changé dans votre code du Call, CC

La seule différence entre Call et Put est que le payoff $C_T = (S_T - K)^+$ devient $P_T = (K - S_T)^+$. On remplace donc simplement

max(S(n,j) − K, 0) par

max(K − S(n,j), 0).

7. Vérifier, en faisant quelques expériences, que le prix du Put est une fonction croissante de K. Pouvez-vous l'expliquez ? Qu'en est-il du Call ?

Si K = S0 on trouve P(0,0) = PP(1,1) = 8,6257597.

On place la définition de PP dans une boucle for K = S0/2 : 2*S0 (!!!) end;

On observe que PP(1,1) croît de 0,03... à 97,58, ce qui se comprend puisque plus K est grand plus le payoff $(K-S_T)^+$ est grand.

Pour le Call c'est le contraire puisque plus K est grand plus $(S_T-K)^+$ est petit.

Ainsi, pour K = S0/2 à K = 2*S0 le Call décroit de 65,7... à 0,42...

8. Vérifier, en faisant quelques expériences, que le prix du Call est une fonction croissante de σ. Pouvez-vous l'expliquez ? Qu'en est-il du Put ?

On revient à K = S0. On place les définitions de CC et PP dans une boucle for sigma = 0,05 : 0,05 : 0,6; up = exp(sigma*sqrt(deltat)); down = 1/up; (!!!); end;

On voit que le Call CC(1,1) croît de (presque) 2,72... à 51,36...

et le Put P(1,1) croît de 1,64... à 14,87.

Ceci s'explique par le fait que Call et Put sont des contrats d'assurance et que sigma mesure le risque assuré

9. Revenons à l'idée de calculer une fonction $C(i,j)$, notée ci-dessous `Callrec`, (et non une matrice `CC`) qui sera donc telle que, pour $i = n$, $C(n,j) = \max(S(n,j) - K, 0)$, puis pour $0 \leq i \leq n-1$ $C(i,j) = (pC(i+1,j+1) + (1-p)C(i+1,j))/R$.

```
function c=Callrec(i,j);
    if i==n then
        c=max (S(n,j)-K,0)
    else
        c=p*Callrec(i+1,j+1)+(1-p)*Callrec(i+1,j);
    end;
endfunction;
disp(Callrec(0,0),"la prime vaut",n,"Pour n=")
```

ATTENTION : avant d'exécuter ce code, choisir $n = 10$, puis **recalculer**, pour cette valeur de n, la fonction S, et les constantes δt, *down*, *up*, R et p. Quelle valeur trouvez-vous pour le Call ? Comparez avec les valeurs données par CC. **Augmenter progressivement** n **de 10 à 20**. Qu'observez-vous ? Pouvez-vous l'expliquer ?(Indication : estimez, en fonction de n combien de fois le programme fait-t-il appel à une valeur de C pour calculer la prime $C(0,0)$)

Pour n = 10 la prime vaut 18,38 (affichage instantané)

Pour n = 15 la prime vaut 18,19 (affichage presque-instantané)

Pour n = 16 ——— 18,78 (petite seconde avant affichage)

Pour n = 18 ——— 18,87 (4 secondes avant affichage)

Pour n = 20 ——— 18,96 (20 secondes avant affichage)

Chaque Callrec(i,j) appelle 2 Callrec(i+1,j+1) et Callrec(i+1,j).

Nb d'appels $2 + 2\cdot 2 + 2\cdot(2\cdot 2) + \cdots + 2\cdot 2^{n-1} = 2\cdot(1+2+4+8+\cdots+2^{n-1}) = 2^{n-1} - 2$.

($i=0$, $i=1$, $i=2$, $i=n-1$)

$1+2+4+8 = 2^3 - 1 = 15$

$2^{n-2} - 1$

croissance exponentielle