Université de Nice
Département de Mathématiques
NOM :
PRENOM :

L3MASS et L3Maths, année 2018-2019
Calcul stochastique et Finance

Feuille de réponses du TP 3
Prix d'une option comme espérance conditionnelle de son payoff

L'objet de cette séance est de calculer dans un modèle de Cox-Ross-Rubinstein à n étapes le prix d'une option, Call ou Put, comme l'espérance de son payoff actualisé, **à toute date** t. Comme précédemment, on modélise les prix par une marche aléatoire S_t définie par $S_0 = S0$ et $S_{t+\delta t} = S_t U_t$, avec $U_t \in \{\texttt{up}, \texttt{down}\}$, en introduisant une fonction $(i,j) \mapsto S(i,j)$ représentant toujours la valeur de l'actif S_t à l'instant $t = i\delta t$ s'il y a eu j up depuis l'instant $t = 0$.

Pour calculer la prime d'un Call sur l'actif S_t on va utiliser la formule $C_0 = \mathbb{E}(\phi(S(n,J))/R^n$, et plus généralement la formule $C_t = \mathbb{E}(\phi(S(n,J_i))|\mathcal{F}_t)/R^n$ où J_i est une variable aléatoire qui suit une loi binômiale : $\mathbb{P}(\{J_i = j\}) = \begin{pmatrix} i \\ j \end{pmatrix} p^j (1-p)^{i-j}$.

1. On pose $n = 100$; $S_0 = 140$; $\sigma = 0.3$; $r = 0.15$ et $K = S_0$. Définir la fonction $S(i,j)$ et les deux matrices CC et PP comme aux précédents TP. Quelles primes du Call et du Put trouvez-vous ?

Prime du Call = $CC(1,1) = 27,121423$
Prime du Put = $PP(1,1) = 7,6205394$

2. Expliquer, en vous aidant de l'aide de Scilab, ce que retourne `binomial(p,n)'` puis expliquez pourquoi la commande `max(S(n,0:n)-K,0)*binomial(p,n)'` calcule l'espérance du payoff du Call $\mathbb{E}(\phi(S(n,J))$.

- binomial(p,n)' est le vecteur colonne (= transposée) des valeurs de la loi binomiale $\mathcal{B}(n,p)$ dont la composante $b_{i+1} = C_n^i p^i (1-p)^{n-i}$. NB : il faut que $n \geq 1$
- $\max(S(n,0:n)-K,0) * \text{binomial}(p,n)' = \varphi(S(n,\cdot)) \cdot b = \sum \varphi(S_{n,i}) \cdot C_n^i p^i (1-p)^{n-i} = \mathbb{E}(\varphi($

3. En déduire la valeur de la prime du Call à la monnaie (n'oubliez pas l'actualisation !) puis celle de la prime du Put à la monnaie. Expliquer vos calculs. Comparer avec les valeur obtenues à la première question.

$C_0 = C(0,0) = \mathbb{E}(\varphi(S_n))\, e^{-rT} = \max(S(n,0:n)-K,0) * \text{binomial}(p,n)' / R\hat{}n$
On retrouve bien 27,121423
NB : si on calcule la différence, Scilab trouve 7,10D-15 = $7,1 \cdot 10^{-15}$ qui est très petit mais non nuls : les erreurs d'arrondis sont distinctes

4. On a, pour tout $t \in 0, \delta t ... T$, la relation suivante dite *relation de parité Call-Put* entre les prix des Call et des Put $C_t - P_t = S_t - Ke^{-r(T-t)}$. Vérifier par un calcul que cette relation est vraie à l'instant final $t = T$.

$* \quad (S_T - K)^+ - (K - S_T)^+ = \begin{cases} S_T - K - 0 \text{ si } S_T \geq K \\ 0 - (K - S_T) \text{ si } S_T \leq K \end{cases} = S_T - K$ dans tous les cas. Donc

si $t = T$, $C_t - P_t = C_T - P_T = (S_T - K)^+ - (K - S_T)^+ = S_T - K = S_T - Ke^{-r0} = S_T - Ke^{-r(}$

5. Vérifier que cette relation est satisfaite à l'instant initial $t = 0$ avec les valeurs obtenues à la question 3 pour les primes des Call et Put.

A-t-on $C_0 - P_0 = S_0 - Ke^{-rT}$
$CC(1,1) - PP(1,1) - (S(0,0) - K/R\hat{}n) = 2,132D-14 = 2,132 \cdot 10^{-14}$
qui est bien très inférieure à la précision de calcul assurée par Scilab
On a donc bien la parité Call-Put pour $t = 0$

6. Le code suivant permet de calculer le prix d'un Call $C(i,j)$ à un instant $t = i\delta t \in \{0, \delta t, 2\delta t, ...\}$ quelconque (et non plus seulement en $t = 0$).

```
function cal=C(i,j)
If i==n then cal=max(S(n,j)-K,0)    // payoff φ(S_T)=(S_T-K)^+ = C_T lorsque t=iδt=nδt=T
else cal=(max(S(n,j:j+n-i)-K,0))*binomial(p,n-i)')/R^(n-i) // E(φ(S_T)) e^{-r(T-t)} si t<T
endfunction;
```

Expliquez pourquoi ceci donne effectivement le prix du Call à tout instant $t \in \{0, \delta t, 2\delta t, ...\}$.

voir commentaire des deux cas $t=n\delta t=T$ et $t=i\delta t<T$ $(i<n)$.

NB : binomial(p,0) n'est pas défini (on aurait pu le définir par 0 ; pas d'alé.
il est donc nécessaire de traiter séparément les cas $t=T$ et $t<T$.

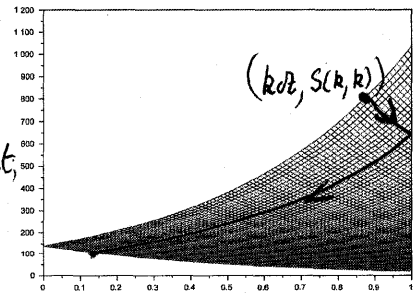
7. Reprendre le code précédent mais cette fois pour un Put à la monnaie en précisant ce qui change. Quelle valeur du Put à la monnaie trouvez-vous après 50 pas de temps si l'actif sous-jacent n'a cessé de baisser ? Quelle valeur au même instant si l'actif sous-jacent n'a baissé que 20 fois ?

```
function put=P(i,j)
if i==n then put=max(K-S(n,j),0)
   else put=(max(K-S(n,j:j+n-i),0)*binomial(p,n-i)')/R^(n-i)
endfunction
```

$i=50$ pas, aucun "up" $P(50,0) = 98,645866$

8. Le code suivant permet de tracer l'arbre CRR. Commenter la figure obtenue.

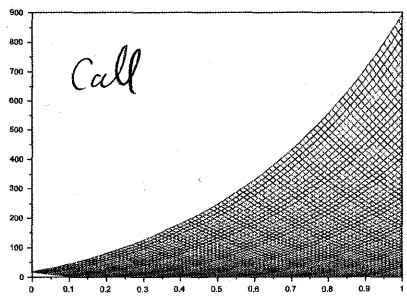
```
Abs=zeros(1+n,1+n);Ord=zeros(1+n,1+n);
for k=0:n
      for m=0:n-k
            Abs(1+m,1+k)=(k+m)*deltat;
            Ord(1+m,1+k)=S(k+m,k);
      end;
      for m=1:k
            Abs(1+n-k+m,1+k)=(n-m)*deltat;
            Ord(1+n-k+m,1+k)=S(n-m,k-m);
      end;
end;
plot(Abs,Ord)
```

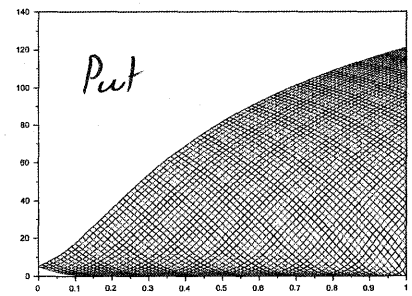


ligne n°k.

plot(Abs, Ord) trace les lignes polygonales dont les abscisses sont données par les lignes de Abs, et les ordonnées par les lignes de Ord.
Abs = matrice (n+1)×(n+1) dont les lignes sont les abscisses
Ord = ——— ordonnées

9. En vous inspirant du tracé de l'arbre CRR donné ci-dessus, tracer l'arbre d'un Call de prix d'exercice $K = 140$. Commenter la figure obtenue. Refaire l'expérience pour un Put.





← ne peut excéder K puisque $K - S_T \leq K$.

Code pour Call : il suffit de remplacer Ord par OrdC et S par C
——— Put : ——— OrdP et ——— P