Université de Nice | Année 2004-2005
ESSI | DESS IMAFA

Fiche de TP 7 :
Calcul de prix d'options parisiennes

On appelle *option parisienne* une option Call ou Put qui prend (*in*) ou perd (*out*) sa valeur si le cours de l'actif sous-jacent passe *assez de temps* sous (*down*) ou au-dessus (*up*) d'une *barrière* H. L'instant présent est $t = 0$; l'échéance est T ; le cours présent du sous-jacent est S_0 ; le prix d'exercice est K. "Assez de temps" signifie une durée au-delà de la barrière égale à τ, appelée *durée d'excursion.* Cette durée peut être *cumulative* (on compte tous les jours passés au-delà de la barrière) ou *non-cumulative* (on ne compte que les jours *consécutifs* passés au-delà de la barrière). Nous noterons DICPc(τ) le prix (prime) d'une option Call parisienne cumulative qui ne prend valeur qu'après une excursion cumulée de durée τ sous la barrière H. De façon analogue, nous noterons DOCPnc(τ) le prix (prime) d'une option Call parisienne non-cumulative qui perd sa valeur après toute excursion sous la barrière H de durée τ. De façon analogue, on définirait DIPPc(τ) ou DOPc(τ) ou DIPPnc(τ) ou DOPnc(τ), et analogue on ajoutant le suffixe Pc(τ) ou Pnc(τ) aux noms des options barrières usuelles

1. Que peut-on dire du signe de DIC-DICPc(τ) ? (expliquer)
2. Que peut-on dire de DICPc(τ)+DOCPc(τ) ? (expliquer)
3. Nous nous proposons de calculer une estimation de DICPc(τ), en utilisant un modèle d'arbre binaire, subdivisant $[0, T]$ en n intervalles égaux de longueur $\delta t = T/n$, avec

$$\left\{ \begin{array}{rcl} S_0 & = & 1 \\ S_{t+\delta t} & = & S_t \exp(\pm\sigma\sqrt{\delta t}) \end{array} \right. \tag{1}$$

dans le cas $T = 1$, $\tau = 0.40$, $S_0 = 140$, $\sigma = 0.4$, $K = 140$, $r = 0.025$, $H = 112$. Voici un code calculant la valeur de l'option en chaque noeud (i, j) de l'arbre, pour $n = 14$:

```
restart;
n=14 :T :=1 :delta_t :=evalf(T/n) :tau :=0.40 :
S0 :=140 :sigma :=0.4 :K :=140 :r :=0.025 :R :=exp(r*delta_t) :
up :=exp(sigma*sqrt(delta_t)) :down :=exp(-sigma*sqrt(delta_t)) :
S :=proc(i,j) option remember : if i=0 then S0 :elif j=0 then evalf(S(i-1,0)*down) :
else evalf(S(i-1,j-1)*up) fi : end;
H :=112 :
SousH :=proc(i,j) option remember :if S(i,j)<H then 1 else 0 fi end;
k0 :=ceil(tau/T*n);# nb mini de pas de l'excursion
AssezLongtemps :=proc(k) option remember;
if k>=k0 then 1 else 0 fi end;
p :=(R-down)/(up-down) :
CallP :=proc(i,j,k) option remember : # k compte le nombre de pas sous H
if i=n then max(S(n,j)-K,0)*AssezLongtemps(k)
else (p*CallP(i+1,j+1,k+SousH(i+1,j+1) + (1-p)*CallP(i+1,j,k+SousH(i+1,j))/R fi end;
CallP(0,0,0)
```

Calculer une estimation de l'option DICPc(0.4) pour $n = 14$ puis $n = 15$ et $n = 16$. Qu'observez-vous ?

4. Adapter le code précédent pour que la procédure `CallP` devienne une fonction de `(i,j,k,n)` , puis tracer en fonction du nombre n de pas de discrétisation, le prix d'une option DICPc, pour `n=1..30` (modifier éventuellement 30 à une valeur différente, en fonction de la puissance de votre machine). Rappel : la commande suivante représente les valeurs d'une fonction `f` pour `n=1..30`

```
with[plots] :plot(['[n,f(n)]'$n=1..30])
```

5. Qu'observez-vous pour les petites valeurs de n ? Comment expliquez-vous cela ?
6. Définir une procédure `DIC` donnant le prix d'une option DIC de même prix d'exercice K et même barrière H en fonction de n. NB : ceci peut se faire en adaptant la procédure CallP (expliquez comment).Qu'obtenez-vous pour `DIC(14) ?`
7. Représentez sur un même graphique le prix d'une DICPc et d'une DIC en fonction de n. Rappel : la commande suivante permet de représenter `Dessin1`, et `Dessin2` sur un même graphique :

```
plots[display](Dessin1,Dessin2);
```

Qu'observez-vous ? (il y a au moins deux remarques possibles).

8. Définir une procédure `CallPnc` calculant le prix d'une option Call parisienne *non-cummulative* et reprendre l'étude ci-dessus.