Projet de Mathématiques : Modélisation d'un marché financier

Mame Diarra Touré
Nisrine Moummou

Le 14 mai 2019

ENSIIE
École Nationale Supérieure d'Informatique pour l'Industrie et l'Entreprise
1 Square de la Résistance, 91025 Evry cedex
https ://www.ensiie.fr/

Sommaire

1 Introduction

Ce projet réalisé en binome consiste en la modélisation d'un marché financier afin de déterminer le prix d'options européennes. Le marché financier est composé de deux actifs que l'on peut échanger à un prix fixé par le marché :
— un actif que l'on appelle sans risque dont le prix à l'instant t est noté S_t^0 que l'on suppose connu dès l'instant initial 0 (ce prix est une variable aléatoire déterministe),
— un actif risqué dont le prix S_t à l'instant t est une variable aléatoire (typiquement une action), dont on modélisera la loi par la suite.
Nous voulons donc déterminer le prix P qu'un vendeur doit faire payé à un investisseur à l'instant initiale pour qu'à la date T l'investisseur reçoive $f(S_T)$ mais également la couverture que doit mettre en place le vendeur pour couvrir ces risques. En resumé on cherche la proportion d'argent investie dans l'actif risqué et la proportion d'argent investie dans l'actif sans risque à n'importe quelle date pour qu'à la date T lorsqu'on vend tout pour récuperer du cash le vendeur a exactement la somme $f(S_T)$.
Pour résoudre ce problème nous modéliserons dans un premier temps l'évolution des prix de manière discrète avec une progression par arbre en suivant le modèle de Cox-Ross-Rubinstein, puis dans un second temps nous modéliserons l'évolution des prix de manière continue avec le modèle de Black-Scholes. Dans un troisième temps nous étudierons la convergence du prix donné par le modèle de Cox-Ross-Rubinstein vers celui donné par le modèle de Black-Scholes.

2 I-Modéle de Cox-Ross-Rubinstein

2.1 Q1-Calcul de q_N

$q_N = Q(T_1^{(N)} = 1 + h_N)$ représente la probabilité que $T_1^{(N)} = 1 + h_N$.Soit p la probabilité que $T_1^{(N)} = 1 + b_N$ alors on a $q_N + p = 1$ et $(1 + h_N) * q_N + (1 + b_N) * p = 1 + r_N$ d'où

$$\boxed{q_N = \frac{r_N - b_N}{h_N - b_N}}$$

2.2 Q-2 Expression de p_N

$p_N = \frac{1}{(1+r_N)^N} E_Q[f(S_{t_N}^{(N)})]$
$E_Q[f(S_{t_N})] = \sum_{k=0}^{N} f(k)P(S_{t_N} = k)$
$S_{t_N} = s(1 + h_N)^X(1 + b_N)^{N-X}$ avec $X- > B(q_N)$ donc
$P(S_{t_N} = k) = C_n^k q_N^k (1 - q_N)^{N-k}$
d'où $E_Q[f(S_{T_N})] = \sum_{k=0}^{N} f(s(1 + h_N)^k(1 + b_N)^{N-k} C_n^k q_N^k (1 - q_N)^{N-k}$ et donc
$p_N = \frac{1}{(1+r_N)^N} \sum_{k=0}^{N} f(s(1 + h_N)^k(1 + b_N)^{N-k} C_n^k q_N^k (1 - q_N)^{N-k}$ avec $q_N = \frac{r_N - b_N}{h_N - b_N}$

$$\boxed{p_N = \frac{1}{(1+r_N)^N} \sum_{k=0}^{N} f(s(1 + h_N)^k(1 + b_N)^{N-k}) C_n^k (\frac{r_N - b_N}{h_N - b_N})^k (\frac{h_N - r_N}{h_N - b_N})^{N-k}}$$

2.3 Q-3 Implémentation fonction price1- pseudo-code

Algorithm 1 Fonction price1

Require: $N \in \mathbb{N}, r_N, h_N, b_N \in \mathbb{R} \quad etf$
Ensure: $p_N = \frac{1}{(1+r_N)^N}\ E_Q[f(S_{t_N}^{(N)})]$
$q_N = \frac{r_N - b_N}{h_N - b_N}$ $somme = 0$ $a = \frac{1}{(1+rN)^N}$
for $i \quad in \quad 0 : N$ **do**
$somme = somme + (choose(N, i) * q_N^i * 1 - q_N * *N - i * f(s * ((1 + h_N)^i * (1 + b_N)^{N-i}$
end for
return $p_N = a * somme$

2.4 Q-4 Test du premier Pricer

```
52 print(paste("le prix donner par la fonction price1 est",teste_pricer1))
53 ```
```

```
[1] "le prix donner par la fonction price1 est 33.9067238510778"
```

2.5 Q-5 Implémentation du Deuxième Pricer

Le prix est donné par la fonction v_k qui est définie par récurrence par :

$$v_k(S_{t_k}{}^{(N)}) := \frac{1}{1+r_N} E_Q[V_{k+1}(S_{t_{k+1}}{}^{(N)})|S_{t_k}{}^{(N)}]$$

. On a :

$$E_Q[v_{k+1}(S_{t_{k+1}}{}^{(N)})|S_{t_k}{}^{(N)}] = q_N * v_{k+1}(S_{t_k}^{(N)}(1+h_N)) + (1-q_N) * v_{k+1}(S_{t_k}^{(N)}(1+b_N))$$

Alors :

$$v_k(S_{t_k}{}^{(N)}) = \frac{1}{1+r_N}(q_N * v_{k+1}(S_{t_k}^{(N)}(1+h_N)) + (1-q_N) * v_{k+1}(S_{t_k}^{(N)}(1+b_N)))$$

sachant que $v_N(S_{t_N}{}^{(N)}) := f(S_{t_N}{}^{(N)})$ on peut alors obtenir par récursion montante $p(N) = v_0(S_0^{(N)})$

2.6 pseudo-code

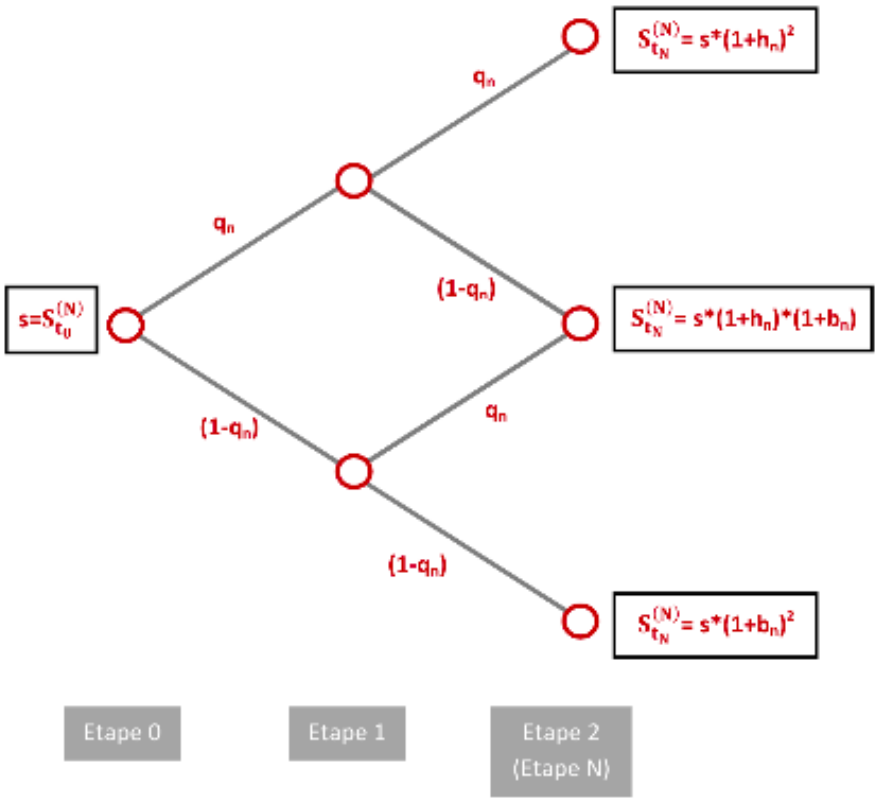
2.7 Q-6 Test du pricer 2

Algorithm 2 Fonction price2

Require: $N \in \mathbb{N}, r_N, h_N, b_N s \in \mathbb{R} \quad etf$
Ensure: $p(N) = v_0(S_0^{(N)})$
$q_N = \frac{r_N - b_N}{h_N - b_N}$ M=matrice N+1 lignes et N+1 colonnes
for $i \quad in \quad 0 : N$ **do**
$M[i+1, N+1] = f(s * ((1+h_N)^{N-i} * (1+b_N)^i$
end for
for $j \quad in \quad N : 1$ **do**
for $i \quad in \quad 1 : N$ **do**
$M[i,j] = (\frac{1}{1+r_N}) * (M[i,j+1] * q_N + M[i+1,j+1] * 1 - q_N)$
end for
end for
return $p(N) = a * M[1,1]$

```
99 ```
```

```
[1] "le prix donner par la fonction price2 est 12.9116636788667"
```



2.8 Q-7 Comparaison des deux pricer

```
110 ```

[1] "pour N= 7"
[1] "le prix donné par price1 est 16.6305165015708"
[1] "le prix donné par price2 est 16.6305165015708"
[1] "la difference est de 3.5527136788005e-15"
```

Nous remarquons que la différence entre les deux pricer est de l'ordre de 10^{-15} quelque soit N entre [5,15]

2.9 Q-8 Couverture

On a le sytéme d'équation suivant

$$\begin{cases} \alpha_{N-1}(S^{(N)}_{t_{N-1}}) * (1+h_N) * S^{(N)}_{t_{N-1}} + \beta_{N-1}(S^{(N)}_{t_{N-1}}) * S^0_{t_N} = f((1+h_N) * S^{(N)}_{t_{N-1}}) & (1) \\ \alpha_{N-1}(S^{(N)}_{t_{N-1}}) * (1+b_N) * S^{(N)}_{t_{N-1}} + \beta_{N-1}(S^{(N)}_{t_{N-1}}) * S^0_{t_N} = f((1+b_N) * S^{(N)}_{t_{N-1}}) & (2) \end{cases}$$

Résolution du systéme $(1)-(2) =>$

$$\alpha_{N-1}(S^{(N)}_{t_{N-1}}) * ((1+h_N)-(1+b_N)) * S^{(N)}_{t_{N-1}} = f((1+h_N) * S^{(N)}_{t_{N-1}}) - f((1+b_N) * S^{(N)}_{t_{N-1}})$$

d'où

$$\alpha_{N-1}(S^{(N)}_{t_{N-1}}) = \frac{f((1+h_N) * S^{(N)}_{t_{N-1}}) - f((1+b_N) * S^{(N)}_{t_{N-1}})}{(h_N - b_N) * S^{(N)}_{t_{N-1}}}$$

et

$$\beta_{N-1}(S^{(N)}_{t_{N-1}}) * S^{(0)}_{t_N} = f((1+b_N) * S^{(N)}_{t_{N-1}}) - \alpha_{N-1}(S^{(N)}_{t_{N-1}}) * (1+b_N) * S^{(N)}_{t_{N-1}}$$

On injecte la valeur de $\alpha_{N-1}(S^{(N)}_{t_{N-1}})$ et on obtient

$$\beta_{N-1}(S^{(N)}_{t_{N-1}}) * S^{(0)}_{t_N} = f((1+b_N) * S^{(N)}_{t_{N-1}}) - \frac{f((1+h_N) * S^{(N)}_{t_{N-1}}) - f((1+b_N) * S^{(N)}_{t_{N-1}})}{(h_N - b_N) * S^{(N)}_{t_{N-1}}} * (1+b_N) * S^{(N)}_{t_{N-1}}$$

or $S^{(0)}_{t_N} = (1+r_N)^N$ donc

$$\beta_{N-1}(S^{(N)}_{t_{N-1}}) = \frac{(1+h_N) * f((1+b_N) * S^{(N)}_{t_{N-1}}) - (1+b_N) * f((1+h_N) * S^{(N)}_{t_{N-1}})}{(1+r_N)^N * (h_N - b_N)}$$

2.10 Q-9 Couverture à la date t_k

$$\begin{cases} \alpha_{k-1}(S^{(N)}_{t_{k-1}}) * (1+h_N) * S^{(N)}_{t_{k-1}} + \beta_{k-1}(S^{(N)}_{t_{k-1}}) * S^0_{t_k} = v_k((1+h_N) * S^{(N)}_{t_{k-1}}) & (1) \\ \alpha_{k-1}(S^{(N)}_{t_{k-1}}) * (1+b_N) * S^{(N)}_{t_{k-1}} + \beta_{k-1}(S^{(N)}_{t_{k-1}}) * S^0_{t_k} = f((1+b_N) * S^{(N)}_{t_{k-1}})) & (2) \end{cases}$$

De la même manière qu'à la question précédente on a

$$\alpha_{k-1}(S_{t_{k-1}}^{(N)}) = \frac{v_k((1+h_N)*S_{t_{k-1}}^{(N)}) - v_k((1+b_N)*S_{t_{k-1}}^{(N)})}{(h_N - b_N)*S_{t_{k-1}}^{(N)}}$$

$$\beta_{k-1}(S_{t_{k-1}}^{(N)}) = \frac{(1+h_N)*v_k((1+b_N)*S_{t_{k-1}}^{(N)}) - (1+b_N)*v_k((1+h_N)*S_{t_{k-1}}^{(N)})}{(1+r_N)^k*(h_N - b_N)}$$

2.11 Q-10 Couverture à la date 0 et à la date 1

A la date 0

$$\alpha_0(S_{t_0}^{(2)}) = \frac{v_1((1+h_N)*S_{t_0}^{(2)}) - v_1((1+b_N)*S_{t_0}^{(2)})}{(h_N - b_N)*S_{t_0}^{(2)}}$$

Dans l'exemple considéré $S_{t_0}^{(2)} = s = 100$,

$$v_1((1+h_N)*S_{t_0}^{(2)}) = \frac{1}{1+r_N}(q_N * f(S_{t_0}^{(2)}(1+h_N)^2) + (1-q_N)*f(S_{t_0}^{(2)}(1+b_N)*(1+h_N)))$$

et de même on a

$$v_1((1+b_N)*S_{t_0}^{(2)}) = \frac{1}{1+r_N}(q_N * f(S_{t_0}^{(2)}(1+b_N)(1+h_N)) + (1-q_N)*f(S_{t_0}^{(2)}(1+b_N)^2))$$

d'où

$$\alpha_0(S_{t_0}^{(2)}) = \frac{(q_N * f(S_{t_0}^{(2)}(1+h_N)^2) + (1-q_N)*f(S_{t_0}^{(2)}(1+b_N)*(1+h_N)))-}{(1+r_N)*(h_N - b_N)*S_{t_0}^{(2)}} -$$

$$\frac{(q_N * f(S_{t_0}^{(2)}(1+b_N)(1+h_N)) + (1-q_N)*f(S_{t_0}^{(2)}(1+b_N)^2))}{(1+r_N)*(h_N - b_N)*S_{t_0}^{(2)}}$$

```
170
171 ```
```

```
[1] "Alpha0 vaut: 0.796116504854369"
```

$$\beta_0(S_{t_0}^{(2)}) = \frac{v_1((1+h_N)*S_{t_0}^{(2)}) - \alpha_0(S_{t_0}^{(2)})*(1+h_N)*S_{t_0}^{(2)}}{(1+r_N)}$$

$$v_1((1+h_N)*S_{t_0}^{(2)}) = \frac{1}{1+r_N}(q_N * f(S_{t_0}^{(2)}(1+h_N)^2) + (1-q_N)*f(S_{t_0}^{(2)}(1+b_N)*(1+h_N))$$

```
[1] "Beta0 vaut: -73.4282213215194"
```

A la date 1 on a

$$\alpha_1(S_{t_1}^{(2)}) = \frac{f((1+h_N) * S_{t_1}^{(2)}) - f((1+b_N) * S_{t_1}^{(2)})}{(h_N - b_N) * S_{t_1}^{(2)}}$$

$$\beta_1(S_{t_1}^{(2)})* = \frac{f((1+b_N) * S_{t_1}^{(2)}) - \alpha_1(S_{t_1}^{(2)}) * (1+b_N) * S_{t_1}^{(2)}}{(1+r_N)^2}$$

Il y'a deux cas à considerer Cas 1 : $S_{t1}^{(2)} = (1+h_N) * S_{t_0}^{(2)}$ donc

$$\alpha_1(S_{t_1}^{(2)}) = \frac{f((1+h_N)^2 * S_{t_0}^{(2)}) - f((1+b_N) * (1+h_N) * S_{t_0}^{(2)})}{(h_N - b_N) * S_{t_1}^{(2)}}$$

```
[1] "alpha1 vaut: 0.976190476190476"
```

$$\beta_1(S_{t_1}^{(2)})* = \frac{f((1+b_N) * (1+h_N) * S_{t_0}^{(2)}) - \alpha_1(S_{t_1}^{(2)}) * (1+b_N) * (1+h_N) * S_{t_0}^{(2)}}{(1+r_N)^2}$$

```
[1] "Beta1 vaut: -91.7852766518993"
```

Cas 2 : $S_{t1}^{(2)} = (1+b_N) * S_{t_0}^{(2)}$ donc

$$\alpha_1(S_{t_1}^{(2)}) = \frac{f((1+h_N) * (1+b_N) * S_{t_0}^{(2)}) - f((1+b_N)^2 * S_{t_0}^{(2)})}{(h_N - b_N) * (1+b_N) * S_{t_0}^{(2)}}$$

```
[1] "alpha1 vaut: 0"
```

$$\beta_1(S_{t_1}^{(2)})* = \frac{f((1+b_N)^2 * S_{t_0}^{(2)}) - \alpha_1(S_{t_1}^{(2)}) * (1+b_N)^2 * S_{t_0}^{(2)}}{(1+r_N)^2}$$

```
[1] "Beta1 vaut: 0"
```

3 Modèle de Black-Scholes

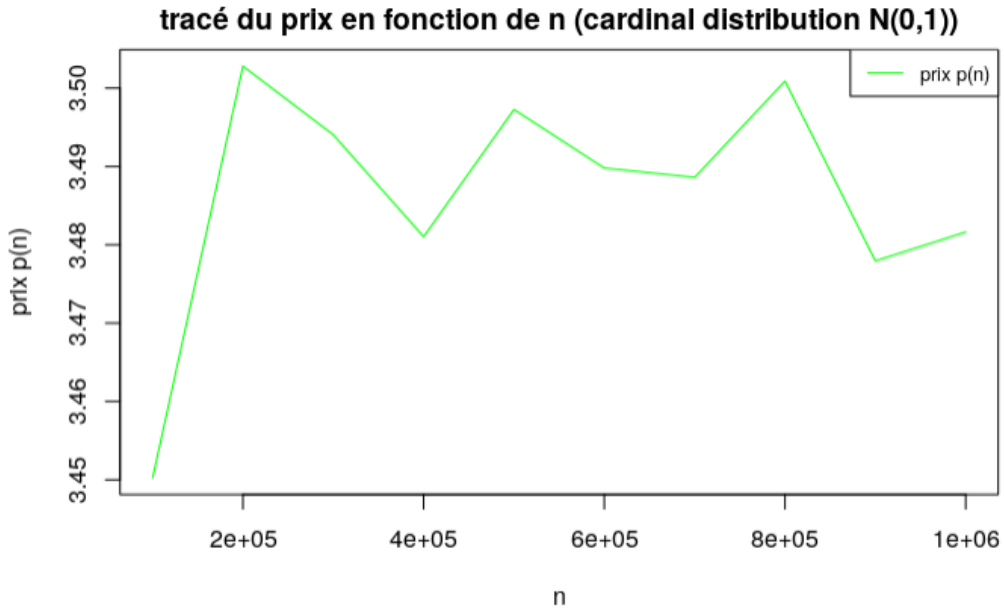
3.1 Q-11 Application de la formule d'Ito

On a dS_t=$S_t(rdt + \sigma dB_t)$
Et la formule d'Ito est donnée par, pour toute fonction g C^2, la différentielle du processus $g(S_t)$ satisfait :
$dg(S_t) = g'(S_t)dS_t + \frac{|\sigma S_t|^2}{2} g"(S_t)dt$
Alors en appliquant la formule d'Ito à $ln(S_t)$, on trouve :
$dln(S_t) = ln'(S_t)dS_t + \frac{|\sigma S_t|^2}{2} ln"(S_t)dt$
Donc :
$dln(S_t) = \frac{1}{S_t} dS_t + \frac{|\sigma S_t|^2}{2} (\frac{-1}{S_t^2})dt$
Alors en remplaçant dS_t par sa valeur, on retrouve :
$dln(S_t) = rdt + \sigma dB_t - \frac{\sigma^2}{2} dt = \sigma dB_t + (r - \frac{\sigma^2}{2})dt$
D'où :
$lnS_t = \sigma B_t + (r - \frac{\sigma^2}{2})t + C$
Or d'après les conditions initiales on a $B_0 = 0$ et $S_0^0 = s$ pour t=0, alors
$ln(s) = C$, donc la formule de S_t est :
$S_t = s * exp(\sigma B_t + (r - \frac{\sigma^2}{2})t)$

3.2 Q-12 Implémentation de price3

Nous avons utilisé la fonction rnorm pour générer les ε_i

3.3 Q-13 Tracé du prix



3.4 Q-14 Démonstration mathématique de la convergence presque sur de $p(n)$ vers p

Montrons que la suite $(p_{(n)})_{n\epsilon N}$ converge presque sûrement vers p
On a :
p :=E$[e^{-rT}f(S_T)]$
Et $p_{(n)} := \frac{1}{n}\sum_{i=1}^{n} e^{-rT}f(sexp((r-\frac{\sigma^2}{2})T+\sigma\sqrt{T}\varepsilon_i))$
Pour celà, on utilisera la loi forte des grands nombres dont l'énoncé est le suivant :
Soit (X_i) une suite de variables aléatoires indépendantes identiquement distribuées, on a alors :
X_n converge presque sûrement vers μ_n où $X_n = \frac{1}{n}\sum_1^n X_i$ et $\mu_n = E[X_1]$

On prend $X_i = e^{-rT}f(s\exp(r-\frac{\sigma^2}{2})T+\sigma\sqrt{T}\varepsilon_i)$
On sait que les (ε_i) sont une suite de variables aléatoires indépendantes et identiquement distribuées de loi N(0,1), donc on peut appliquer la loi pour les X_i.

Donc $\hat{p}_{(n)}$=$\frac{1}{n}\sum_1^n X_i$ converge presque sûrement vers $E[X_1]$

Or $X_1 = e^{-rT}f(s\exp(r-\frac{\sigma^2}{2})T+\sigma\sqrt{T}\xi_i)]$

Et d'après la question 11, on sait que : $p = E[e^{-rt}f(S_t)]$= $E[e^{-rT}f(sexp(\sigma B_T+(r-\frac{\sigma^2}{2})T]$

où B_T un mouvement Brownien qui réalise : $\forall s \in [0,T] : B_t - B_s \sim N(0,t-s)$.
Alors en prenant s=0 on retrouve : $B_T \sim N(0,T)$

d'où $\frac{B_T}{\sqrt{T}} \sim N(0,1)$
Donc : B_T=$\varepsilon_t\sqrt{T}$
En remplaçant dans la formule de p, on trouve :
$p = E[exp^{-rT}f(s\exp(r-\frac{\sigma^2}{2})T+\sigma\sqrt{T}\varepsilon_i)]$
Donc $p = E[X_1]$
Alors $\hat{p}_{(n)}$ converge presque sûrement vers p.

3.5 Pricer par formule fermée

3.6 Q-15 Implémentation de la fonction put

Le prix de l'option est donné par la formule

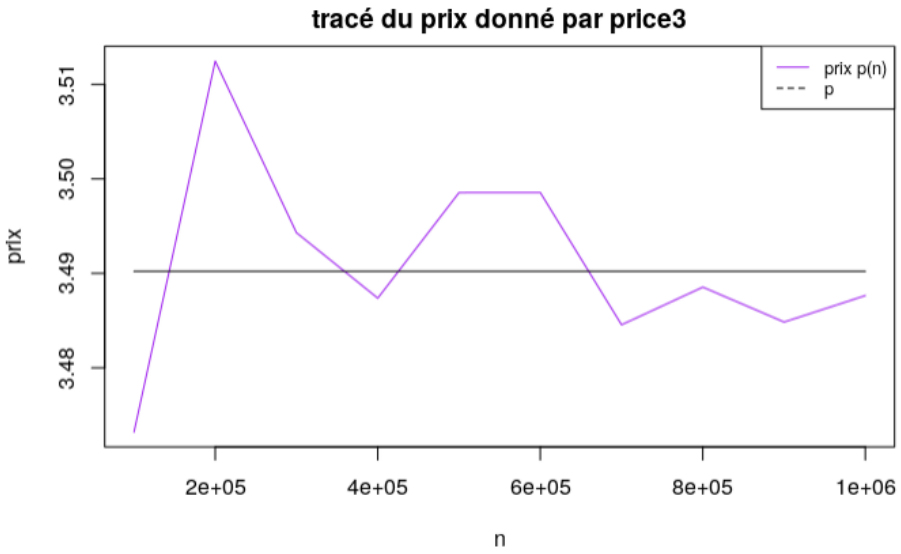
$$p = -sF(-d) + Ke^{-rT}F(-d+\sigma\sqrt{T})$$

avec $d = \frac{1}{\sigma\sqrt{T}} * (ln(\frac{s}{K}) + (r+\frac{\sigma^2}{2}) * T)$ On utlise la fonction *pnorm* pour obtenir la fonction de répartition $F(x) = P(Y \leqslant x)$ avec $Y -> N(0,1)$

3.7 Q-16 Test de la fonction put

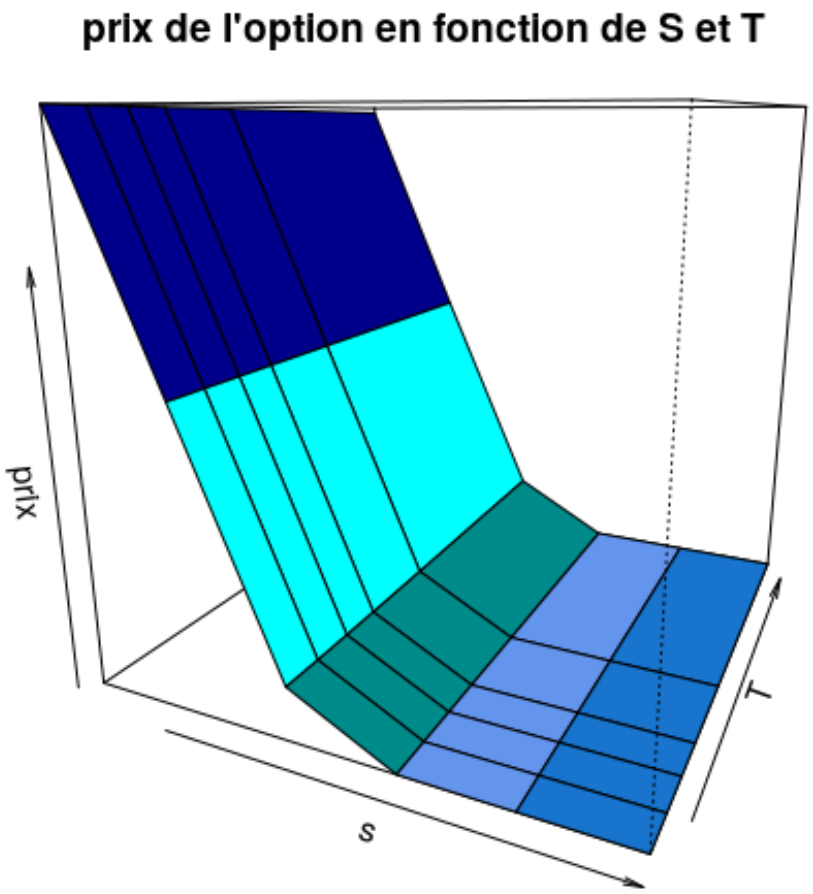
```
347 ```

[1] "pour s= 100,r=0.01,T=1, K=100 et sigma=0.1  le put vaut: 2.25740546863713"
```

3.8 Q-17 Tracé du prix donné par price3 en fonction de n



Ce tracé est en accord avec le résultat démontré à la question 14 : $p(n)$ converge presque sûrement vers p

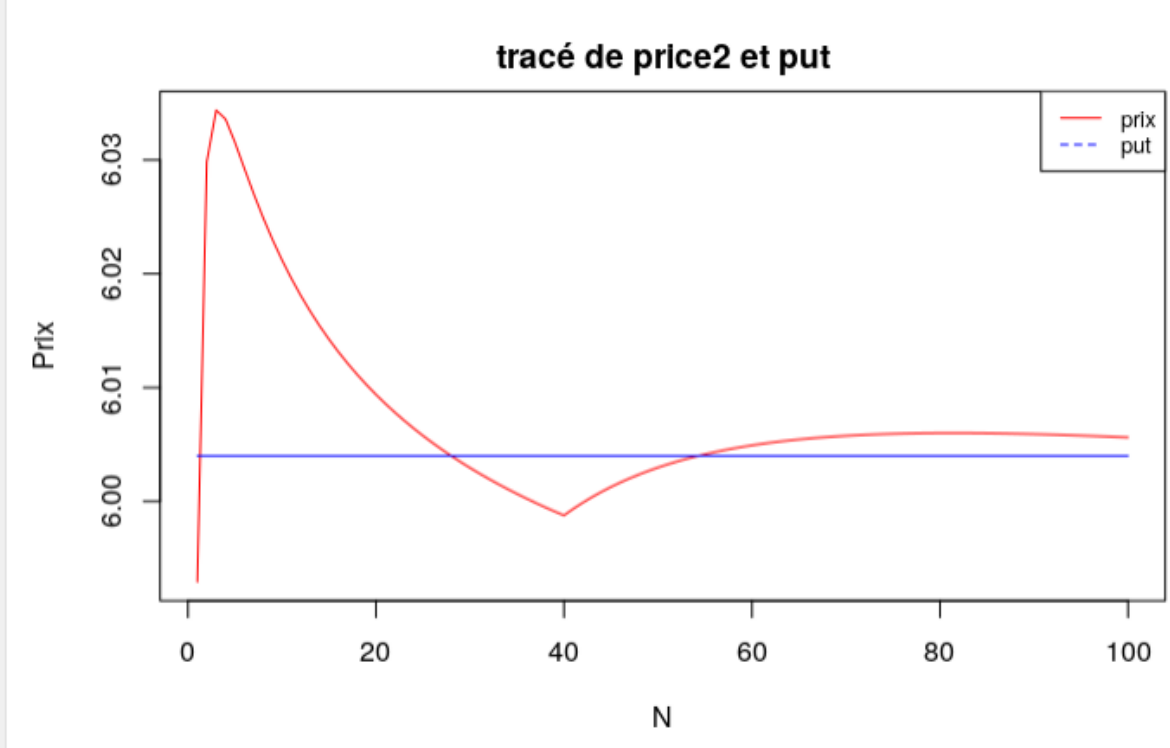
3.9 Q-18 Tracé du prix de l'option en fonction de S et T



On remarque qu'à S constant le prix de l'option varie très peu. On peut donc en déduire que le prix de l'option ne dépend pas beaucoup du temps T.

4 Convergence des prix

4.1 Q-19 Tracé du prix donné par price2 et du put p



On remarque que le price2 semble converger vers le prix du put pour N » 1. Or à la question 17 nous avons vu que le pricer3 convergeait aussi vers le put. Donc ce graphe montre bien la convergence des prix dans le cas du modèle de Cox-Ross-Rubinstein(pricer2) vers les prix donnés dans le cas du modèle de Black-Scholes(pricer3).

5 EDP de Black-Scholes

La fonction prix du put P vérifie l'équation différentielle suivante $\forall t, S \in [0,T] \times [0,L]$:

$$\frac{\partial P}{\partial t} + rS\frac{\partial P}{\partial S} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 P}{\partial S^2} = rP$$

Les conditions de bords nous donnent les valeurs de P en :

$$(T,s) \quad t.q \quad s \in [0,L]$$

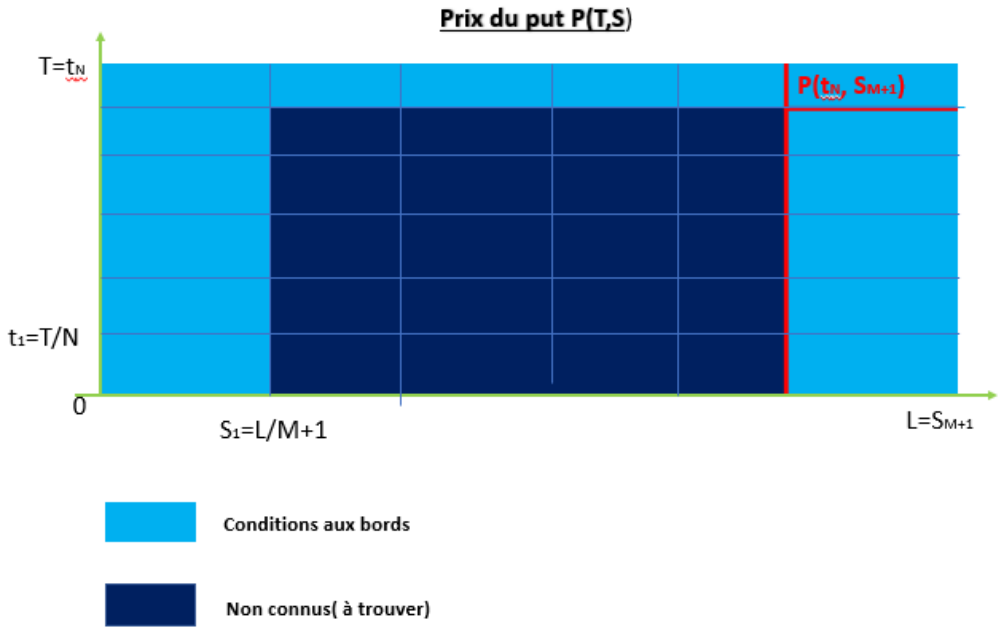
$$(t,0) \quad t.q \quad t \in [0,T]$$

$$(t,L) \quad t.q \quad t \in [0,T]$$

Nous avons la discrétisation suivante : Soit $N, M \in \mathbb{N}^*$:

$$\begin{cases} \Delta T = \frac{T}{N} \quad et \quad \forall n \in \{0,...,N\}, \boxed{t_n = n\Delta T} \\ \Delta S = \frac{L}{M+1} \quad et \quad \forall i \in \{0,...,M+1\}, \boxed{S_i = i\Delta S} \end{cases}$$

5.1 Grille schématisant la résolution



5.2 Changement de variable et passage d'une condition terminale à une condition initiale

On veut une condition initiale et non terminale donc posons $u(t,S) = P(T-t,S)$ on obtient donc

$$\frac{\partial u}{\partial t}(t,S) = -\frac{\partial P}{\partial t}(T-t,S)$$

l'équation différentielle

$$\frac{\partial P}{\partial t} = -rS\frac{\partial P}{\partial S} - \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 P}{\partial S^2} + rP$$

devient :

$$\frac{\partial u}{\partial t}(t,S) = rS\frac{\partial u}{\partial S}(t,S) + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 u}{\partial S^2}(t,S) - ru(t,S)$$

5.3 Approche numérique des EDP de Black-Scholes par différences finies

Nous allons utiliser un θ-schéma qui permet de prendre en compte la méthode des différences finies explicite($\theta = 1$), implicite($\theta = 0$) et de Crank-Nicholson($\theta = \frac{1}{2}$)le Schéma est le suivant $u_{n,i}$ sera l'approximation de $u(t_n, s_i)$ $\forall (n,i) \in \{0,...,N-1\} \times \{1,...,M\}$ on a

$$\frac{\partial u}{\partial t}(t_n, s_i) \simeq \frac{1}{\Delta T}\left(u_{n+1,i} - u_{n,i}\right)$$

$$\frac{\partial u}{\partial S}(t_n, S_i) \simeq \frac{\theta(u_{n+1,i+1} - u_{n+1,i}) + (1-\theta)(u_{n,i+1} - u_{n,i})}{\Delta S}$$

$$\frac{\partial^2 u}{\partial S^2}(t_n, S_i) \simeq \frac{\theta(u_{n+1,i+1} - 2u_{n+1,i} + u_{n+1,i-1}) + (1-\theta)(u_{n,i+1} - 2u_{n,i} + u_{n,i-1})}{\Delta S^2}$$

En approximant l'équation différentielle $\forall (n,i) \in \{0, ..., N-1\} \times \{1, ..., M\}$:

$$\boxed{\frac{\partial u}{\partial t}(t_n, S_i) = rS_i \frac{\partial u}{\partial S}(t_n, S_i) + \frac{1}{2}\sigma^2 S_i^2 \frac{\partial^2 u}{\partial S^2}(t_n, S_i) - ru(t_n, S_i)}$$

On remplace donc les derivées partielles par leur expression et on obtient

$$\frac{1}{\Delta T}(u_{n+1,i} - u_{n,i}) = rS_i \frac{\theta(u_{n+1,i+1} - u_{n+1,i}) + (1-\theta)(u_{n,i+1} - u_{n,i})}{\Delta S} +$$

$$\frac{1}{2}\sigma^2 S_i^2 \frac{\theta(u_{n+1,i+1} - 2u_{n+1,i} + u_{n+1,i-1}) + (1-\theta)(u_{n,i+1} - 2u_{n,i} + u_{n,i-1})}{\Delta S^2} - ru_{n,i} \quad (1)$$

condition initiale $u_{0,i} = max(K - S_i, 0) \quad \forall i \in \{0, ..., M+1\}$

condition au bord

$$u_{n,0} = Ke^{-rt_n} \quad \forall n \in \{1, ..., N\}$$

$$u_{n,M+1} = 0 \quad \forall n \in \{1, ..., N\}$$

5.4 Traduction matricielle de (2)

Afin de simplifier l'expression de (2), on pose :

$$a = r\Delta T\theta i$$

$$b = \sigma^2 \Delta T\theta i^2$$

$$c = r\Delta T(1-\theta)i$$

$$d = \sigma^2 \Delta T(1-\theta)i^2$$

Alors (2) devient :

$$(1 + a + b)u_{n+1,i} - (a + \frac{1}{2}b)u_{n+1,i+1} - \frac{1}{2}bu_{n+1,i-1} =$$

$$(1 - c - r\Delta T - d)u_{n,i} + (\frac{1}{2}d + c)u_{n,i+1} + \frac{1}{2}du_{n,i-i}$$

On pose encore :

$$A_1 = 1 + a + b$$
$$A_2 = -(a + \frac{1}{2}b)$$
$$A_3 = -\frac{1}{2}b$$

Et

$$B_1 = 1 - c - d$$
$$B_2 = \frac{1}{2}d + c$$
$$B_3 = \frac{1}{2}d$$

(2) devient donc :

$$A_1 u_{n+1,i} + A_2 u_{n+1,i+1} + A_3 u_{n+1,i-1} = B_1 u_{n,i} + B_2 u_{n,i+1} + B_3 u_{n,i-1}$$

Ce qu'on peut représenter sous forme de :

$$\boxed{AU_{n+1} = BU_n + C_n}$$

Où Un est un vecteur colonne tel que :

$$U_n = \begin{pmatrix} u_{n,1} \\ ... \\ u_{n,M} \end{pmatrix}$$

Et A et B sont deux matrices carrées tridiagonales de taille M :

$$A = \begin{pmatrix} A_{1,1} & A_{1,2} & 0 & 0 & ... & ... & ... & ... & ... & 0 \\ A_{2,1} & A_{2,2} & A_{2,3} & 0 & ... & ... & ... & ... & ... & 0 \\ 0 & A_{3,2} & A_{3,3} & A_{3,4} & ... & ... & ... & ... & ... & ... \\ 0 & 0 & ... & ... & ... & ... & ... & ... & ... & ... \\ ... & ... & ... & ... & ... & ... & ... & ... & ... & ... \\ ... & ... & ... & ... & A_{j,j-1} & A_{j,j} & A_{j,j+1} & ... & ... & ... \\ ... & ... & ... & ... & ... & ... & ... & ... & ... & ... \\ ... & ... & ... & ... & ... & ... & ... & ... & ... & 0 \\ ... & ... & ... & ... & ... & ... & ... & ... & ... & A_{M-1,M} \\ 0 & 0 & ... & ... & ... & ... & ... & 0 & A_{M,M-1} & A_{M,M} \end{pmatrix}$$

Et

$$B = \begin{pmatrix} & & & & & & & & & \\ B_{1,1} & B_{1,2} & 0 & 0 & ... & ... & ... & ... & ... & 0 \\ B_{2,1} & B_{2,2} & B_{2,3} & 0 & ... & ... & ... & ... & ... & 0 \\ 0 & B_{3,2} & B_{3,3} & ... & ... & ... & ... & ... & ... & ... \\ 0 & 0 & ... & ... & ... & ... & ... & ... & ... & ... \\ ... & ... & ... & ... & ... & ... & ... & ... & ... & ... \\ ... & ... & ... & ... & Bj, j-1 & B_{j,j} & B_{j,j+1} & ... & ... & ... \\ ... & ... & ... & ... & ... & ... & ... & ... & ... & ... \\ ... & ... & ... & ... & ... & ... & ... & ... & ... & 0 \\ ... & ... & ... & ... & ... & ... & ... & ... & ... & B_{M-1,M} \\ 0 & 0 & ... & ... & ... & ... & ... & 0 & B_{M,M-1} & B_{M,M} \end{pmatrix}$$

avec :

$$A_{j,j} = 1 + a + b \quad \forall j \in \{1, ..., M\}$$

$$A_{j,j+1} = -(a + \frac{1}{2}b) \forall j \in \{1, ..., M-1\}$$

$$A_{j,j-1} = -\frac{1}{2}b \forall j \in \{2, ..., M\}$$

et on a

$$B_{j,j} = 1 - c - d - r\Delta T \quad \forall j \in \{1, ..., M\}$$

$$B_{j,j+1} = \frac{1}{2}d + c \quad \forall j \in \{1, ..., M-1\}$$

$$B_{j,j-1} = \frac{1}{2}d \quad \forall j \in \{2, ..., M\}$$

et C_n est un vecteur colonne tel que :

$$C_n = \begin{pmatrix} C_{n,1} \\ \\ ... \\ \\ C_{n,M} \end{pmatrix}$$

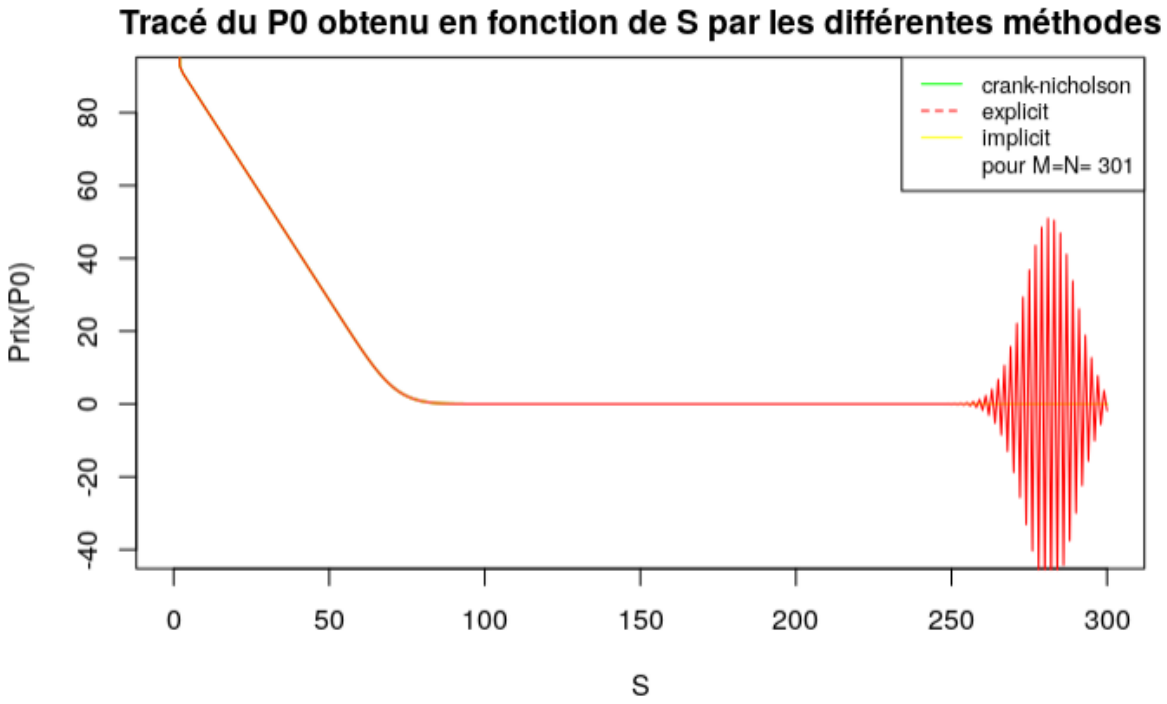
Et d'après les conditions de bords ainsi que les conditions initiales, on trouve que $C_{n,1} = \frac{1}{2}Ke^{-rt_n}(be^{-r\Delta T} + d)$ et $C_{n,i} = 0 \forall i \in \{2, .., M\}$

5.5 Résolution numérique de l'équation différentielle de Black-Scholes pseudo-code

Nous avons utiliser la fonction solve de R pour résoudre le système d'équations qui optimise les calculs en faisant la décomposition LU de la matrice A

Algorithm 3 Résolution numérique de l'équation différentielle de Black-Scholes

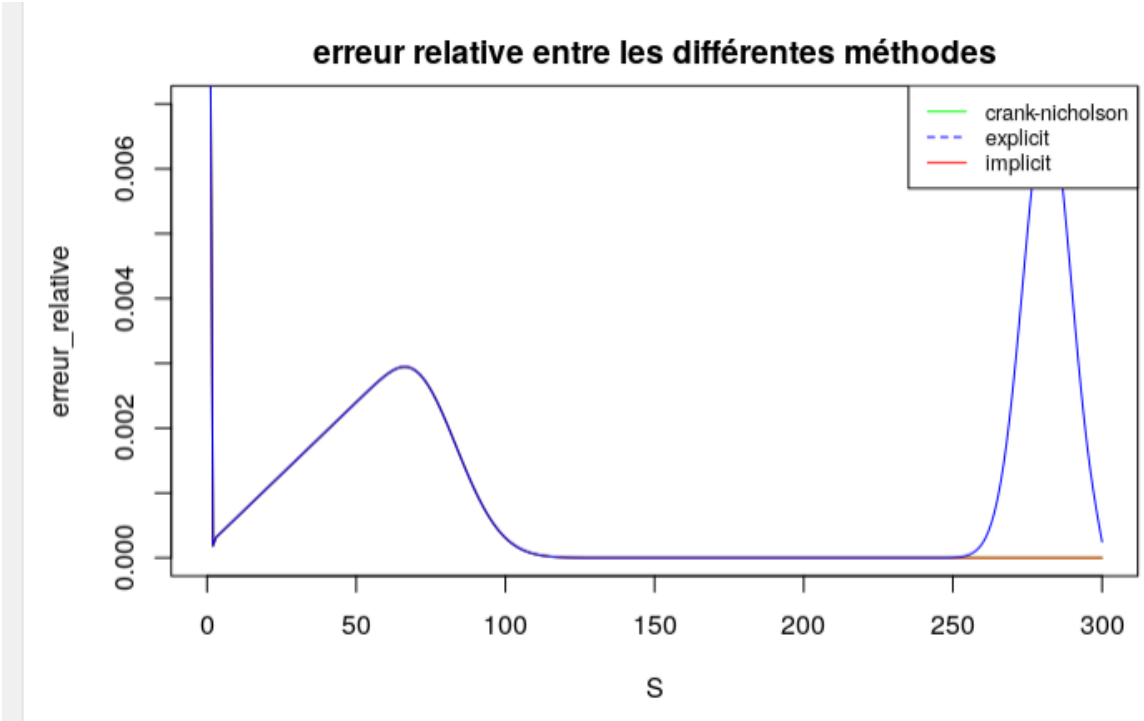
Require: $M, N \in \mathbb{N}, K, r, \sigma \in \mathbb{R} \quad \theta \in [0,1]$
Ensure: $U_{N+1} = A^{-1}BU_N + A^{-1}C_N$
  $dt = \frac{T}{N} \quad ds = \frac{L}{M+1}$
  Initialiser A et B grâce aux conditions (CI,CB)
  Calculer U_1 grâce aux conditions aux bords (U_1 pas U_N car changement de variable)
  for $n \quad in \quad 1 : N$ **do**
    Calculer C_n
    Calculer $b_n = BU_n + C_n$
    $U_{n+1} = solve(A, b_n)$
  end for
  return U_{N+1}



Nous remarquons sur le tracé que la méthode explicite diverge pour M=N=301

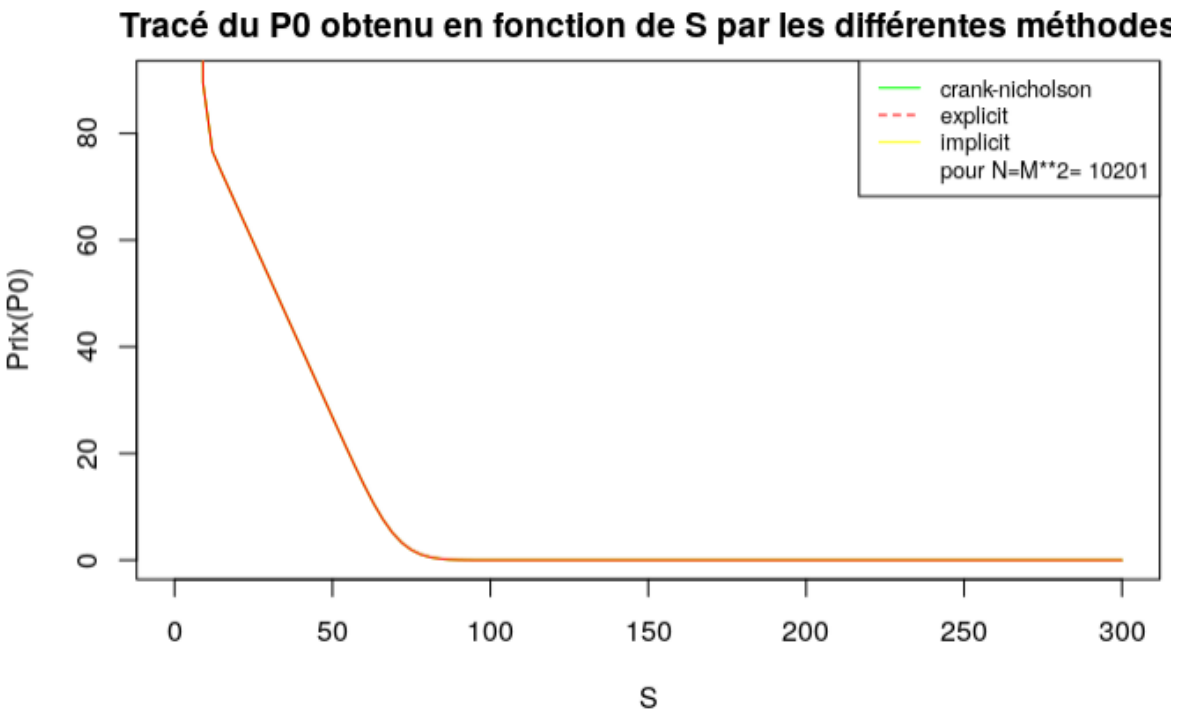
5.6 Tracé de l'erreur relative entre les differentes méthodes

Nous avons utilisé la formule suivante pour calculer l'erreur relative entre les 3 méthodes erreur_relative(A,B)=$\frac{2*||A-B||}{||A+B||}$



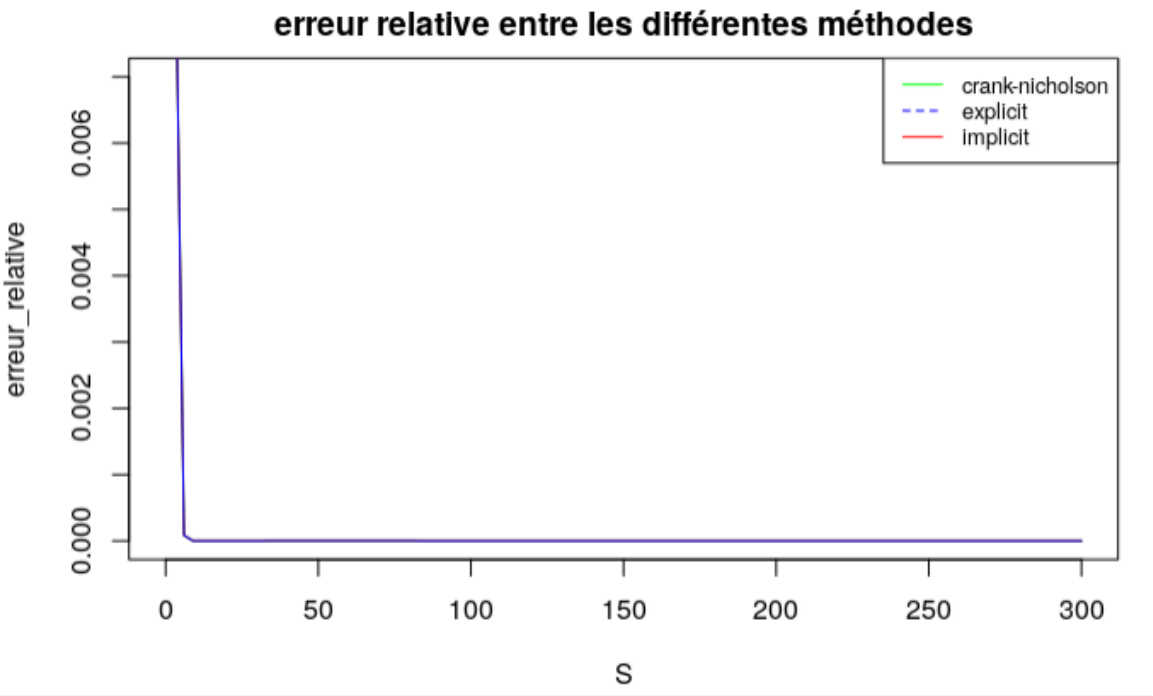
On en déduit que lorsque les 3 méthodes convergent, elles donnent des résultats très proches. Mais pour M $>$ 290 la divergence de la méthode explicite fait exploser l'erreur relative.

5.7 Utilisation de la CFL pour stabiliser le schéma explicite



Nous remarquons que la méthode explicite ne diverge plus lorsque la condition CFL est remplie La méthode explicite est stable si le pas de temps est de l'ordre du carré du pas d'espace. Autrement dit, si : $\Delta t = \Delta S^2$

On obtient le tracé suivant pour l'erreur relative



Comme le montre le tracé l'erreur relative entre les 3 méthodes et put est la même lorsque la CFL est remplie.