Pricing des options américaines et réseaux de neurones

Marwan AKROUCH - Hamza ZERHOUNI

Ecole des Ponts ParisTech

16 Février 2021

Sommaire

Plan

Introduction

- Option : titre financier qui donne à son détenteur le droit d'acheter ou de vendre
- Nature de l'option : option de call pour une option d'achat et option de put pour une option de vente
- On peut distinguer les options selon leur date d'expiration:
  - option américaine: l'option peut être exercée à n'importe quel instant avant l'échéance
  - option européenne: l'option ne peut être exercée qu'à la date d'expiration

Hypothèses et paramètres

- Le marché est complet.

Hypothèses et paramètres

- Le marché est complet.
- r le taux d'intérêt sans risque;
- σ la volatilité;

Hypothèses et paramètres

- Le marché est complet.
- r le taux d'intérêt sans risque;
- σ la volatilité;
- T temps de maturité;
- N nombre de pas de temps;
- S_n prix de l'action à l'instant $n\frac{T}{N}$, ou vecteur des prix en cas de panier d'actions;
- K prix d'exercice.
- - Option de call: $payoff(S, K) = (S - K)_+$
  - Option de put: $payoff(S, K) = (K - S)_+$

Plan

Modèle de Cox-Ross-Rubinstein en dimension 1

À chaque instant n, le cours de l'actif peut soit augmenter d'un facteur u avec proba p, soit diminuer d'un facteur d avec proba $1 - p$, tq $u > d > 0$

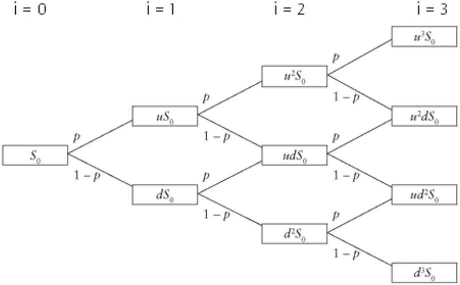


Figure: Arbre binomiale du modèle de Cox Ross pour N=3

Le prix de l'actif $(S_n)_{n\in\mathbb{N}}$ suit le processus dynamique stochastique suivant:
$\forall n \in \mathbb{N}, S_{n+1} = S_n \cdot (u \cdot 1_{\{U_{n+1}=1\}} + d \cdot 1_{\{U_{n+1}=0\}})$
avec $(U_n)_{n\in\mathbb{N}}$ une suite de variables aléatoires iid suivant une loi de Bernoulli de paramètre p.

Implémentation

La valeur de l'actif correspond à $u(0, S_0)$.

La valeur de l'actif à chaque instant n vérifie l'algorithme de programmation dynamique suivant:

- Pour une option européenne:
$$\begin{cases} u(N,x) & = payoff(x,K) \\ u(n,x) & = \frac{pu(n+1,xu)+(1-p)u(n+1,xd)}{1+r} \end{cases}$$
- Pour une option américaine:
$$\begin{cases} u(N,x) & = payoff(x,K) \\ u(n,x) & = max(payoff(x,K), \frac{pu(n+1,xu)+(1-p)u(n+1,xd)}{1+r}) \end{cases}$$

Convergence du modèle de Cox-Ross-Rubinstein vers le modèle de Black-Scholes pout une option européenne

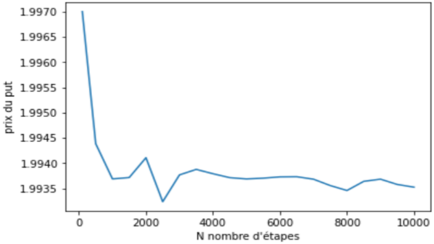


Figure: Convergence de CRR vers BS: variation du prix du put **européen** en fonction de N avec les paramètres $S_0 = 50,\ K = 40,\ r = 0.1,\ \sigma = 0.4,\ T = 1$

```
Number of steps used in Binomial Model: N =  1000
European call Binomial:        15.800194179898709
European call Analytic (BS):   15.800068798345148
European put Binomial:         1.9936909013350086
European put Analytic (BS):    1.9935655197835285
Call-put parity for binomial:   c-p= 13.8065032785637
       S0 exp(-qT)- K exp(-rT)= 13.806503278561621
```

Figure: Comparaison des résultats obtenus avec le modèle de CCR et le modèle de BS avec les paramètres $S_0 = 50,\ K = 40,\ r = 0.1,\ \sigma = 0.4,\ T = 1$

Résultats pour une option américaine

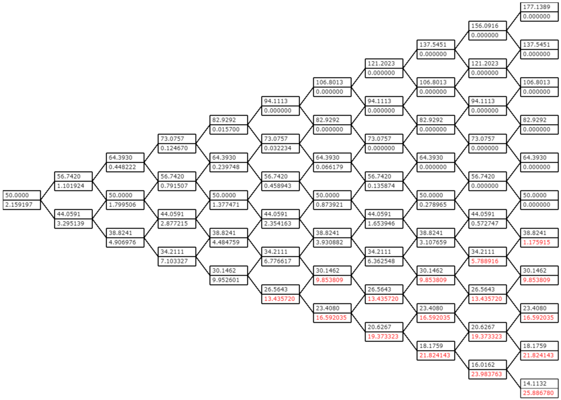


Figure: Arbre binomiale du modèle de Cox Ross pour le pricing d'un **put américain** avec les paramètres $S_0 = 50$, $K = 40$, $r = 0.1$, $\sigma = 0.4$, $T = 1$, $N = 10$.

Dans chaque noeud, on trouve en haut le prix de l'action, en bas la valeur de l'option. Les noeuds en rouge correpondent aux noeuds où il est optimal d'exercer.

Modèle de Cox-Ross-Rubinstein en dimension supérieure

- Dans le cas où le sous-jacent de l'option est un panier de d actions, on concidère le processus $S_n = (S_n^1, \dots, S_n^d)$ où S_n^i est le cours de l'action i à l'instant n. On fait l'hypothèse que les $S_n^i, 1 \leq i \leq d$ sont indépendantes.

Modèle de Cox-Ross-Rubinstein en dimension supérieure

- Dans le cas où le sous-jacent de l'option est un panier de d actions, on concidère le processus $S_n = (S_n^1, \dots, S_n^d)$ où S_n^i est le cours de l'action i à l'instant n. On fait l'hypothèse que les $S_n^i, 1 \leq i \leq d$ sont indépendantes.

- Dans le cas d=2, le couple (S_{n+1}^1, S_{n+1}^2) est égal à:

- $(u_1 S_n^1, u_2 S_n^2)$ avec proba $p_1 p_2$
- $(u_1 S_n^1, d_2 S_n^2)$ avec proba $p_1(1-p_2)$
- $(d_1 S_n^1, u_1 S_n^2)$ avec proba $(1-p_1)p_2$
- $(d_1 S_n^1, d_2 S_n^2)$ avec proba $(1-p_1)(1-p_2)$

Modèle de Cox-Ross-Rubinstein en dimension supérieure

- Dans le cas où le sous-jacent de l'option est un panier de d actions, on concidère le processus $S_n = (S_n^1, \ldots, S_n^d)$ où S_n^i est le cours de l'action i à l'instant n. On fait l'hypothèse que les $S_n^i, 1 \leq i \leq d$ sont indépendantes.

- Dans le cas d=2, le couple (S_{n+1}^1, S_{n+1}^2) est égal à:

- $(u_1 S_n^1, u_2 S_n^2)$ avec proba $p_1 p_2$
- $(u_1 S_n^1, d_2 S_n^2)$ avec proba $p_1(1-p_2)$
- $(d_1 S_n^1, u_1 S_n^2)$ avec proba $(1-p_1)p_2$
- $(d_1 S_n^1, d_2 S_n^2)$ avec proba $(1-p_1)(1-p_2)$

- Complexité exponentielle : $O(2^{dN})$

Plan

Arrêt optimal et option américaine

- On note $(Z_n = f(n, S_n))_{0\leq n\leq N}$ le processus des *payoff* actualisés. On a: $Z_n = f(n, S_n) = e^{-r\frac{nT}{N}} payoff(S_n, K)$

- On suppose que le processus des payoff actualisés (Z_n) est adapté à la filtration (F_n) et que $\mathbb{E}[max_{0\leq n\leq N}|Z_n|^2] < +\infty$.

Arrêt optimal et option américaine

- On note $(Z_n = f(n, S_n))_{0\leq n\leq N}$ le processus des *payoff* actualisés. On a: $Z_n = f(n, S_n) = e^{-r\frac{nT}{N}} payoff(S_n, K)$

- On suppose que le processus des payoff actualisés (Z_n) est adapté à la filtration (F_n) et que $\mathbb{E}[max_{0\leq n\leq N}|Z_n|^2] < +\infty$.

- Dans un marché complet, si $\mathbb{E}$ dénote l'espérance sous la probabilité risque neutre, la valeur de l'option à l'instant n s'écrit:

$$u(n, S_n) = sup_{\tau\in\mathcal{T}_n}\mathbb{E}[f(\tau, S_\tau)|F_n \ ou \ S_n]$$

Arrêt optimal et option américaine

- On note $(Z_n = f(n, S_n))_{0 \leq n \leq N}$ le processus des *payoff* actualisés. On a: $Z_n = f(n, S_n) = e^{-r\frac{nT}{N}} payoff(S_n, K)$

- On suppose que le processus des payoff actualisés (Z_n) est adapté à la filtration (F_n) et que $\mathbb{E}[max_{0 \leq n \leq N}|Z_n|^2] < +\infty$.

- Dans un marché complet, si $\mathbb{E}$ dénote l'espérance sous la probabilité risque neutre, la valeur de l'option à l'instant n s'écrit:

$$u(n, S_n) = sup_{\tau \in \mathcal{T}_n} \mathbb{E}[f(\tau, S_\tau) | F_n \ ou \ S_n]$$

En appliquant la théorie de l'enveloppe de Snell, la valeur de l'option américaine $u(0, S_0)$ est donnée par le problème de programmation dynamique:

$$\begin{cases} u(N, S_N) & = & f(N, S_N) \\ u(n, S_n) & = & max(\ f(n, S_n) \ , \ \mathbb{E}[\ u(n+1, S_{n+1}) \mid S_n \ ] \ ) \end{cases} \quad (1)$$

Algorithme de programmation dynamique

Soit τ_j^* le temps d'arrêt optimal après l'instant j, tel que

$$\tau_j^* = min\{n \geq j \mid u(n, S_n) = f(n, S_n)\}$$

Algorithme de programmation dynamique

Soit τ_j^* le temps d'arrêt optimal après l'instant j, tel que

$$\tau_j^* = min\{n \geq j \mid u(n, S_n) = f(n, S_n)\}$$

On a:

$$u(j, S_j) = \mathbb{E}[f(\tau_j^*, S_{\tau_j^*}) \mid S_j] \tag{2}$$

En utilisant les temps d'arrêt optimaux, l'algorithme (1) s'écrit:

$$\begin{cases} \tau_N^* & = & N \\ \tau_j^* & = & j1_{\{f(j,S_j) \geq u(j,S_j)\}} + \tau_{j+1}^* 1_{\{f(j,S_j) < u(j,S_j)\}} \end{cases}$$

Algorithme de programmation dynamique

Soit τ_j^* le temps d'arrêt optimal après l'instant j, tel que

$$\tau_j^* = min\{n \geq j \mid u(n, S_n) = f(n, S_n)\}$$

On a:

$$u(j, S_j) = \mathbb{E}[f(\tau_j^*, S_{\tau_j^*}) \mid S_j] \tag{2}$$

En utilisant les temps d'arrêt optimaux, l'algorithme (1) s'écrit:

$$\begin{cases} \tau_N^* & = & N \\ \tau_j^* & = & j1_{\{f(j,S_j) \geq u(j,S_j)\}} + \tau_{j+1}^* 1_{\{f(j,S_j) < u(j,S_j)\}} \end{cases}$$

On se débarasse de $u(j, S_j)$ en utilisant les formules (1) et (2), on obtient:

$$\begin{cases} \tau_N^* & = & N \\ \tau_j^* & = & j1_{\{f(j,S_j) \geq \mathbb{E}[f(\tau_{j+1}^*, S_{\tau_{j+1}^*} | S_j)]\}} \\ & + & \tau_{j+1}^* 1_{\{f(j,S_j) < \mathbb{E}[f(\tau_{j+1}^*, S_{\tau_{j+1}^*} | S_j)]\}} \end{cases} \tag{3}$$

Plan

Méthode de Monte Carlo

$$\frac{dS_t}{S_t} = rdt + \sigma dW_t$$

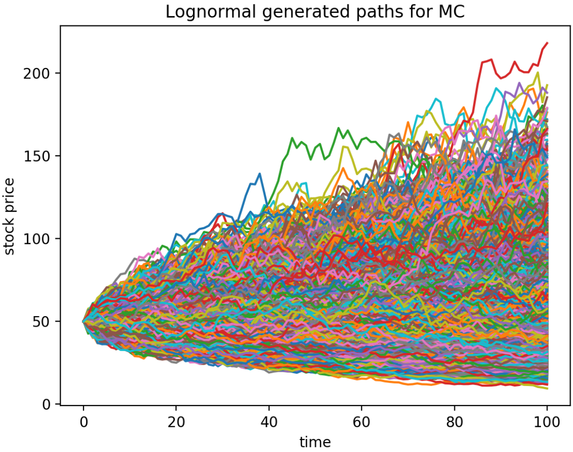


Figure: 10000 trajectoires lognormales d'une action avec les paramètres $S_0 = 50$, $\sigma = 0.4$, $r = 0.1$, $T = 1$

Schéma de l'algorithme

Générer M trajectoires de l'action $(S_j^{(m)}, 0 \leq j \leq N)$, $1 \leq m \leq M$

$$\Downarrow$$

Approximer $\mathbb{E}[f(\tau_{j+1}^{(m)}, S_{\tau_{j+1}^{(m)}}^{(m)})|S_j)]$

$$\Downarrow$$

Trouver les temps d'arrêt optimaux $\tau_j^{(m)}$, $1 \leq j \leq N$

$$\Downarrow$$

Déduire la valeur de l'option américaine

$$\text{Option value} = \frac{1}{M}\sum_{m=1}^{M} f(\tau_0^{(m)}, S_{\tau_0^{(m)}}^{(m)})$$

Algorithme de Longstaff Schwartz

- L'idée basique de Longstaff Schwartz est d'introduire une **méthode des moindres carrés**.

Algorithme de Longstaff Schwartz

- L'idée basique de Longstaff Schwartz est d'introduire une **méthode des moindres carrés**.

- On cherche ϕ_j dans L^2(loi de S_j) tq $\phi_j(S_j) = \mathbb{E}[f(\tau^*_{j+1}), S_{\tau^*_{j+1}} | S_j]$.

- L^2 étant un espace Hilbert, $\phi_j(S_j)$ peut être calculé en minimisant $\mathbb{E}[(f(\tau^*_{j+1}, S_{\tau^*_{j+1}}) - \phi_j(S_j))^2]$, en pratique cela revient à minimiser $\sum_{m=1}^{M}(f(\tau^{(m)}_{j+1}, S^{(m)}_{\tau^{(m)}_{j+1}}) - \phi(S^{(m)}_j))^2$ avec l'approximation de Monte-Carlo.

Algorithme de Longstaff Schwartz

- L'idée basique de Longstaff Schwartz est d'introduire une **méthode des moindres carrés**.

- On cherche ϕ_j dans L^2(loi de S_j) tq $\phi_j(S_j) = \mathbb{E}[f(\tau^*_{j+1}), S_{\tau^*_{j+1}}|S_j]$.

- L^2 étant un espace Hilbert, $\phi_j(S_j)$ peut être calculé en minimisant $\mathbb{E}[(f(\tau^*_{j+1}, S_{\tau^*_{j+1}}) - \phi_j(S_j))^2]$, en pratique cela revient à minimiser $\sum_{m=1}^{M}(f(\tau^{(m)}_{j+1}, S^{(m)}_{\tau^{(m)}_{j+1}}) - \phi(S^{(m)}_j))^2$ avec l'approximation de Monte-Carlo.

- Avec $(g_l, l \geq 1)$ une base hilbertienne de L^2, on écrit $\phi_j = \sum_{l\geq 1} \alpha_l g_l$.

Algorithme de Longstaff Schwartz

- L'idée basique de Longstaff Schwartz est d'introduire une **méthode des moindres carrés**.

- On cherche ϕ_j dans L^2(loi de S_j) tq $\phi_j(S_j) = \mathbb{E}[f(\tau^*_{j+1}), S_{\tau^*_{j+1}}|S_j]$.

- L^2 étant un espace Hilbert, $\phi_j(S_j)$ peut être calculé en minimisant $\mathbb{E}[(f(\tau^*_{j+1}, S_{\tau^*_{j+1}}) - \phi_j(S_j))^2]$, en pratique cela revient à minimiser $\sum_{m=1}^{M}(f(\tau^{(m)}_{j+1}, S^{(m)}_{\tau^{(m)}_{j+1}}) - \phi(S^{(m)}_j))^2$ avec l'approximation de Monte-Carlo.

- Avec $(g_l, l \geq 1)$ une base hilbertienne de L^2, on écrit $\phi_j = \sum_{l\geq 1} \alpha_l g_l$.

Implémentation

Data: M trajectoires simulées
Result: le calcul des temps optimaux τ_j^*
Initialization: $\tau_N^m \leftarrow N$ pour $0 \leq m \leq M$, j$\leftarrow N-1$
while $j \geq 1$ **do**
    Trouver le polynôme ϕ qui minimise $\sum_{m=1}^{M}(f(\tau_{j+1}^{(m)}, S_{\tau_{j+1}^{(m)}}^{(m)}) - \phi(X_j^{(m)}))^2$
    for *chaque trajectoire m* **do**
        if $f(j, S_j^{(m)}) \geq \phi(S_j^{(m)})$ **then**
            $\tau_j^{(m)} \leftarrow j$
        else
            $\tau_j^{(m)} \leftarrow \tau_{j+1}^{(m)}$
        end
    end
    $j \leftarrow j-1$
end

Algorithm 1: Calcul des temps optimaux

Résultats

Figure: évolution du prix d'un put bermudéen en fonction de M le nombre de trajectoire généré, le prix est calculé 50 fois pour chaque valeur de M avec nouvelle génération de trajectoires à chaque fois, en bleu la moyenne des valeurs obtenus pour chaque M, rappel: $S_0 = 50$, $K = 40$, $\sigma = 0.4$, $r = 0.1$, $T =$1an, $N = 10$

Dimension Supérieure

Dans le cas d'une option sur un panier de d actions, on suppose que chaque actif de prix S_t^i suit un modèle de Black Scholes guidé par un mouvement W_t^i,

Dimension Supérieure

Dans le cas d'une option sur un panier de d actions, on suppose que chaque actif de prix S_t^i suit un modèle de Black Scholes guidé par un mouvement W_t^i, de sorte que:

- $(W_t, t > 0)$ un mouvement brownien d-dimensionnel de composants indépendants
- une matrice Σ de taille $d \times d$

$$\frac{dS_t^i}{S_t^i} = rdt + [\Sigma W_t]_i \tag{4}$$

1. On génère M trajectoires du vecteur $S_n = (S_n^0, \dots S_n^d)$ à l'aide de (4)
2. on applique l'algorithme de Longstaff Schwartz avec ϕ_j un polynôme à d variables.

par exemple, si d=2 on cherche ϕ_j sous la forme de combinaison linéaire de $\{1, X_1, X_2, X_1^2, X_2^2, X_1X_2, X_1^3, X_2^3, X_1^2X_2, X_1X_2^2, \dots\}$

Plan

Pourquoi les réseaux de neurones?

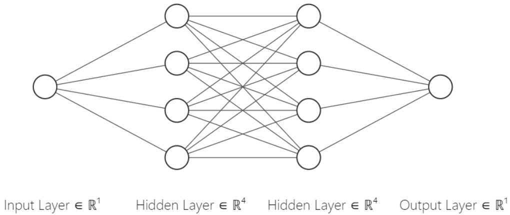
- La régression repose sur la composition **linéaire** en couches de fonctions simples, polynômes par exemple.
- Le but derrière DNN est d'approximer des fonctions **non linéaires** assez complexes.
- Théorème d'approximation universel.
- Les DNN ne subissent pas (en principe) la malédiction des dimensions élevées.

Approximation d'une espérance conditionnelle par réseaux de neurones

- On remplace la partie régression linéaire dans l'algorithme de Longstaff Schwartz par un réseau de neurones pour approximer $\mathbb{E}[f(\tau^*_{j+1}, S_{\tau^*_{j+1}})|S_j)]$.
- Le réseau est entraîné par le couple $(S_j^{(m)}, f(\tau_{j+1}^{(m)}, S^{(m)}_{\tau^{(m)}_{j+1}}))$ afin de trouver la fonction ϕ_j qui approxime l'espérance conditonnelle.

Implémentation

Après plusieurs expérimentations, on a choisi de prendre un réseau de neurones de la forme:

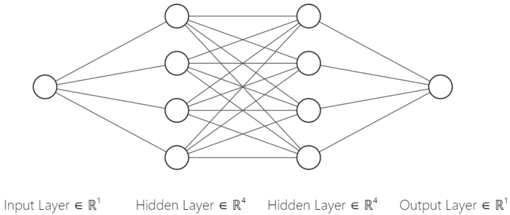


avec la fonction d'activation:

$$elu(x) = \begin{cases} x & \text{si } x \geq 0 \\ e^x - 1 & \text{si } x < 0 \end{cases}$$

Implémentation

Après plusieurs expérimentations, on a choisi de prendre un réseau de neurones de la forme:



avec la fonction d'activation:

$$elu(x) = \begin{cases} x & \text{si } x \geq 0 \\ e^x - 1 & \text{si } x < 0 \end{cases}$$

On a utilisé la bibliothèque **keras** pour implémenter ce RN.

résultats

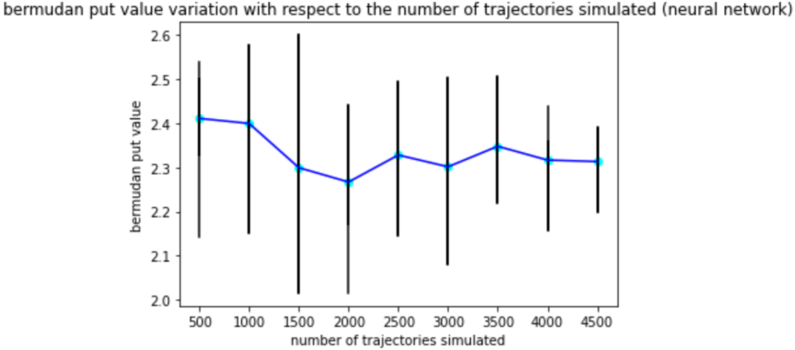


Figure: Réseau de neurones: évolution du prix d'un put bermudéen en fonction de M le nombre de trajectoire généré, le prix est calculé 10 fois pour chaque valeur de M avec nouvelle génération de trajectoires à chaque fois, en bleu la moyenne des valeurs obtenus pour chaque M, rappel: $S_0 = 50$, $K = 40$, $\sigma = 0.4$, $r = 0.1$, $T =$1an, $N = 3$

Dimension supérieure

Pour une option sur un panier de d actions, on génére M trajectoires de $S_n = (S_n^1, \dots, S_n^d)$ selon le modèle de Black Scholes, ensuite on adapte notre RN en prenant:

- une couche d'entrée de d neurones.
- une 1ère couche intérmédiaire de $4d$ neurones.
- une 2ème couche intérmédiaire de $4d$ neurones.
- une couche de sortie contenant 1 neurone.

Conclusion

Questions ?