Valorisation des options américaines à l'aide de la méthode de Gauss-Seidel projetée

MOHAMED AHMED Mohamed Lemine , Mohamedou Chrif M'Hamed

Date : 18 novembre 2024

Contents

1 Introduction

Ce travail est consacré à l'application et à l'analyse de la méthode de Gauss-Seidel projetée (GSP) dans un contexte numérique et financier. Le but est d'étudier une méthode itérative pour résoudre des systèmes non linéaires contraints, puis de l'appliquer à la valorisation d'options américaines.

Dans ce rapport, nous présentons les étapes suivies, les résultats obtenus, et répondons aux questions posées dans le sujet. Les graphiques insérés illustrent les observations réalisées à chaque étape.

À la fin du rapport, vous trouverez une annexe contenant le code Python utilisé pour réaliser les calculs et générer les graphiques présentés tout au long de l'analyse.

2 Implémentation de la méthode de Gauss-Seidel projetée (GSP)

Dans cette première tâche, nous avons implémenté la routine GSP, une méthode itérative permettant de résoudre un système de contraintes non linéaires sous la forme :

$$X \geq G, \quad AX - B \geq 0, \quad et (AX - B)^T (X - G) = 0.$$

La méthode repose sur une condition de stationnarité définie par :

$$\frac{\|X^{k+1} - X^k\|}{\|X^k\|} \leq \epsilon,$$

ou bien un nombre maximal d'itérations K_{max}. Les étapes principales sont les suivantes :

1. Initialisation des paramètres α, β, γ, des vecteurs B et G, et de la solution initiale X_0.
2. Calcul des itérés successifs $X^{k+1} = \Psi(X^k)$, en tenant compte des contraintes.
3. Arrêt lorsque la condition de stationnarité est respectée ou que K_{max} est atteint.

Les résultats de cette tâche sont les fondations nécessaires aux étapes suivantes.

3 Le problème d'obstacle

Dans cette partie, nous avons utilisé la routine GSP pour résoudre un système défini avec une fonction $g(x)$. Nous avons ensuite analysé les résultats obtenus et les performances de l'algorithme en fonction de la taille du système N.

Courbe 1 : Comparaison entre $g(x)$ et $u(x)$

La première courbe compare la fonction $g(x)$ et la solution $u(x)$ obtenue après convergence :

- $g(x)$ représente les contraintes imposées à chaque point.
- $u(x)$ suit $g(x)$ lorsqu'il y a contact ($u(x) = g(x)$), et s'en éloigne lorsque $u(x) > g(x)$.

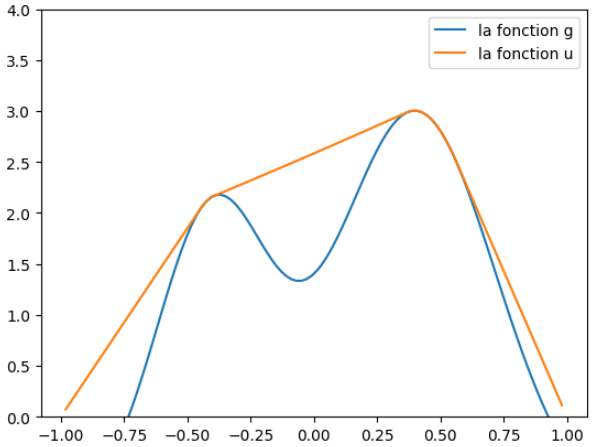

Figure 1: Comparaison entre la fonction $g(x)$ et la solution $u(x)$ obtenue après convergence.

Courbe 2 : Évolution de k en fonction de N

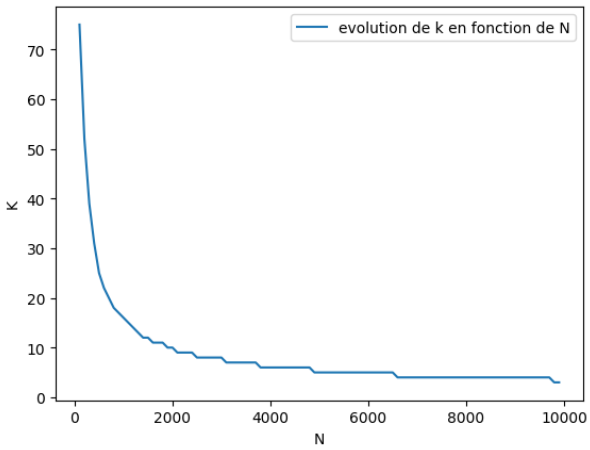

Figure 2: Évolution du nombre d'itérations k en fonction de la taille du système N.

La deuxième courbe illustre comment le nombre d'itérations k nécessaires pour converger évolue avec la taille du système N.

- Pour de grandes tailles N, k diminue rapidement après une phase initiale. Cela reflète le fait que la méthode GSP converge mieux pour des systèmes de plus grande taille, car les approximations successives deviennent plus précises. Limite : Lorsque
- GSP approche une limite, mais l'algorithme reste une approximation et peut ne pas respecter entièrement les trois conditions d'optimalité mentionnées dans le sujet

Réponses aux questions posées

1. **Évolution de k avec N :** Le nombre d'itérations k diminue après un seuil critique lorsque N devient très grand.
2. **Conditions non respectées :** La condition d'orthogonalité $(AX - B)^T(X - G) = 0$ souvent celle qui n'est pas parfaitement respectée, car GSP est une méthode approximative et non exacte.

4 Options américaines

4.1 Résolution et Implémentation :

Dans cette partie, l'objectif est de valoriser une option américaine (call, put) avec une approche EDP en utilisant la méthode de GSP implémentée dans la partie 1.
On sait que le calcul de $u(x_n, t_m)$, i.e. la valeur de l'option à l'instant t_m sachant que la valeur de l'actif sous-jacent à cet instant vaut x_n, revient à calculer $\mathbf{U_n}^m$, où $\mathbf{U}$ est une matrice de taille $N \times M$.

Le calcul de $\mathbf{U}$ se fait avec une méthode itérative en calculant, à chaque itération, $\mathbf{U}^m$, où $\mathbf{U}^m$ représente la i-ème colonne de la matrice $\mathbf{U}$.

On a que $\mathbf{U}^{m+1}$ est la solution du problème suivant :

$$\mathbf{A}\mathbf{U}^{m+1} - (\mathbf{U}^m + \mathbf{F}) \geq 0, \quad \mathbf{U}^{m+1} - \mathbf{G} \geq 0, \left(\mathbf{A}\mathbf{U}^{m+1} - (\mathbf{U}^m + \mathbf{F})\right)^T \left(\mathbf{U}^{m+1} - \mathbf{G}\right) = 0.$$

Donc, en posant à chaque itération $B = \mathbf{U}^m + F$, on trouve que $\mathbf{U}^{m+1}$ est la solution d'un problème de complémentarité linéaire que l'on peut résoudre avec la méthode de GSP.

4.2 Réponses aux questions posées

- **choix de X^0 à chaque itération :** pour chaque itération, on choisit la valeur U^m comme la valeur initiale dans la méthode GSP pour calculer U^{m+1} .
- **le cas d'un call sans dividence $\delta = 0$** : dans ce cas, le call américain est égal au call européen donc on n'exerce qu'à l'échéance.

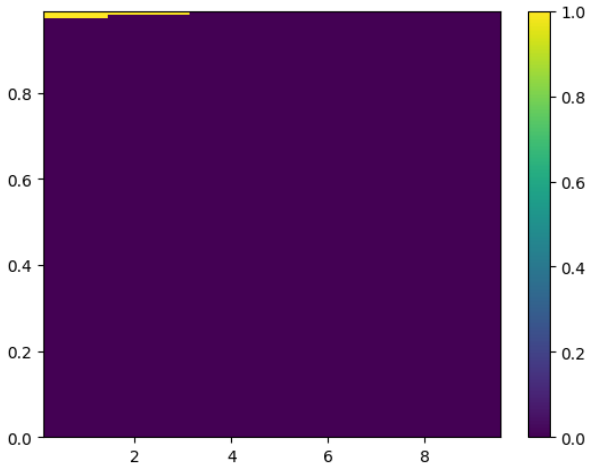

Figure 3: zone d'exercie d'un call sans dividence

- **le cas d'un put sans inéret $r = 0$** : dans ce cas, le put américain est égal au put européen si la taux de dividence $\delta > 0$

4.3 Zone d'exercice CALL :

Zone d'exercie pour le CALL

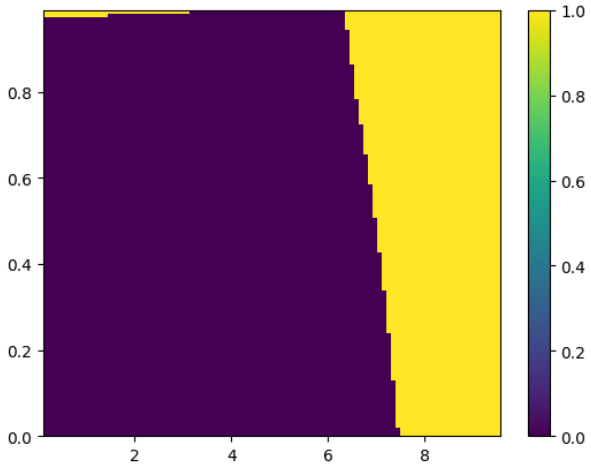

Figure 4: zone d'exercie pour le CALL

4.4 Zone d'exercie pour le PUT :

Zone d'exercie pour le PUT

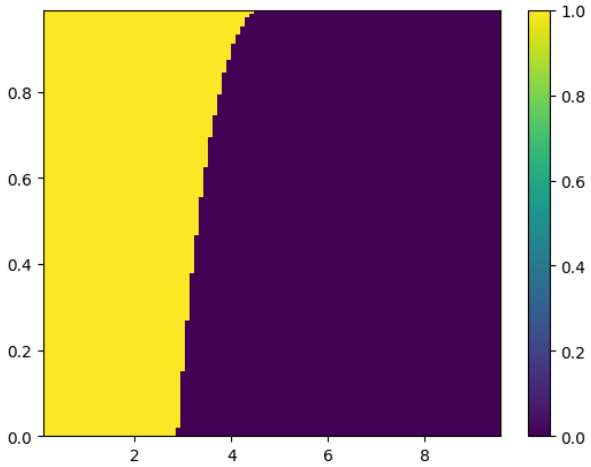

Figure 5: zone d'exercie pour le PUT

5 Conclusion

Ce TP nous a permis de découvrir la méthode de Gauss-Seidel projetée et de l'appliquer à des problèmes numériques et financiers. Les résultats obtenus sont satisfaisants et conformes aux attentes, bien que des ajustements soient nécessaires pour garantir des solutions encore plus précises. Cette étude met en lumière les limites des méthodes numériques et l'importance des paramètres dans la résolution de problèmes contraints.

6 Annexe

6.1 Implémentation de la routine GSP:

```
def GSP(alpha , beta , gamma , B , G, x_0 , epsilon , K) :
    """
    Méthode de Gauss-Seidel projetée (GSP).

    Arguments :
    - alpha, beta, gamma : paramètres du système.
    - B : vecteur des termes indépendants.
    - G : vecteur des contraintes de projection.
    - x_0 : vecteur initial.
    - epsilon : tolérance pour le critère d'arrêt.
    - K : nombre maximal d'itérations.

    Retourne :
    - res : solution approximative.
    - k : nombre d'itérations effectuées.
    """
    def psi(X) :
        n = X.shape[0]
        Z = np.zeros(n)
        Z[0] = max((1/alpha)*(B[0] - beta*X[1]) , G[0])
        for i in range(1,n-1):
            Z[i] = max((1/alpha)*(B[i] - gamma*Z[i-1]- beta*X[i+1]) , G[i])

        Z[n-1] = max((1/alpha)*(B[n-1] - gamma*Z[n-2]) , G[n-1])
        return Z

    res = x_0
    k = 0
    while k <= K :
        res_prec = res
        res = psi(res)
        if np.linalg.norm(res - res_prec) / np.linalg.norm(res_prec) <= epsilon :
            return (res , k)
        k+=1

    return (res , K)
```

6.2 Valorisation de l'option américaine:

```
def price_amr(N,M,K,T,sigma,r, delta, payOff,epsilon,K_max):
    U = np.zeros((N,M))
    C = np.zeros((N,M))

    a = np.log(K/10)
    b = np.log(10*K)
```

```
    delta_x = (b-a)/(N+1)
    delta_t = T/M

    alpha = (1+ sigma**2 * delta_t/delta_x**2 + r*delta_t)
    beta = -(sigma**2 * delta_t/(2*delta_x**2) +
        (r - delta - sigma**2/2 )*delta_t/(2*delta_x))
    gamma = (-sigma**2 * delta_t/(2*delta_x**2) +
        (r - delta - sigma**2/2 )*delta_t/(2*delta_x))

    F = np.zeros(N)
    F[0]= -gamma*payOff(np.exp(a),K)
    F[N-1]= -beta*payOff(np.exp(b),K)

    G = np.array([payOff(np.exp(a+n*delta_x),K) for n in range(1,N+1)])

    # cas initiale :
    U[:,0]=GSP(alpha , beta , gamma , G+F , G, G , epsilon , K_max)[0]
    for i in range(N):
        C[i,0] = 1 if U[i,0] == G[i] else 0
    for m in range(1,M):
        B = U[:,m-1] + F
        x_0 = U[:,m-1]
        U[:,m] = GSP(alpha , beta , gamma , B , G, x_0 , epsilon , K_max)[0]
        for i in range(N):
            C[i,m] = 1 if U[i,m] == G[i] else 0


    return U, C


def payoff_call(S,K):
    return max(S-K,0)


def payoff_put(S,K):
    return max(K-S,0)
```