Niveau 2 - Méthodes Numériques
Cours 5
Simulation de processus stochastiques
Estimation d'un Zéro Coupon

Jean-François Berger-Lefébure

17 Octobre 2024

Contents

1 Rappels

1.1 Formule générale : Simulation exacte d'un processus stochastique (Itô)

Un processus stochastique X_t est défini par :

$$X_t = x + \int_0^t \mu(s, X_s)ds + \int_0^t \sigma(s, X_s)dW_s$$

- x : Valeur initiale du processus à $t = 0$.
- $\int_0^t \mu(s, X_s)ds$: Composante déterministe (lisse), représentant l'accumulation des petites variations $\mu(s, X_s)$ à chaque instant.
  - Exemple : Une constante $\mu = 0.1$ ajoute une tendance de croissance de $0.1 \times t$.
- $\int_0^t \sigma(s, X_s)dW_s$: Composante stochastique, représentant l'effet du mouvement brownien W_s modulé par $\sigma(s, X_s)$.
  - Exemple : Si $\sigma = 0.2$, les fluctuations aléatoires sont amplifiées de 20 %.

1.2 Formule spécifique : Processus de Vasicek

Le processus de Vasicek est défini par :

$$dX_t = -a(X_t - \mu)dt + \sigma dW_t$$

Forme intégrale :

$$X_t = \mu + (X_0 - \mu)e^{-at} + \sigma \int_0^t e^{-a(t-s)}dW_s$$

- μ : Moyenne de long terme vers laquelle le processus tend.
- $(X_0 - \mu)e^{-at}$: Contribution de la valeur initiale X_0, qui diminue exponentiellement avec le temps t, à un rythme contrôlé par $a > 0$ (force de rappel).
- $\sigma \int_0^t e^{-a(t-s)}dW_s$: Contribution stochastique du mouvement brownien, pondérée par $e^{-a(t-s)}$, qui réduit progressivement l'impact des fluctuations passées.

1.3 Loi conditionnelle dans le processus de Vasicek

La loi conditionnelle entre deux instants t_j et t_{j+1} est :

$$X_{t_{j+1}} \mid X_{t_j} \sim \mathcal{N}\left(\mu + (X_{t_j} - \mu)e^{-a\Delta t}, \frac{\sigma^2}{2a}(1 - e^{-2a\Delta t})\right)$$

- **Moyenne :**
$$\mu + (X_{t_j} - \mu)e^{-a\Delta t}$$
  - La moyenne du processus évolue vers μ, avec un effet de rappel contrôlé par a.
  - Plus a est grand, plus le processus revient rapidement vers μ.
- **Variance :**
$$\frac{\sigma^2}{2a}(1 - e^{-2a\Delta t})$$
  - La variance dépend de σ^2 (intensité des fluctuations) et a (force de rappel).
  - Pour Δt petit, Variance $\approx \sigma^2\Delta t$, comme dans un mouvement brownien.

—

1.4 Points essentiels

1.4.1 Propriété Markovienne

- Tout processus défini par une EDS est **markovien**.
- Le futur dépend uniquement de la valeur actuelle X_t.

1.4.2 Simulation discrète

- Approximée par un schéma d'Euler avec Δt petit.
- Nécessite des lois conditionnelles pour chaque intervalle $[t_j, t_{j+1}]$.

1.4.3 Processus de Vasicek

- Présente une force de rappel vers la moyenne μ.
- La variance est contrôlée par σ (volatilité) et a (force de rappel).

1.5 Questions

Question 1 :

Quelle est la propriété principale d'un processus markovien ?

- (A) Il dépend de toute la trajectoire passée.
- (B) Le futur dépend uniquement de l'état actuel.
- (C) Il suit toujours une loi normale.

Bonne réponse : (B)
Explication : Dans un processus markovien, le futur est totalement déterminé par l'état présent, sans dépendance à la trajectoire passée.

Question 2 :

Quel est l'effet du paramètre $a > 0$ dans le processus de Vasicek ?

- (A) Il contrôle la volatilité du processus.
- (B) Il décrit la force de rappel vers la moyenne μ.
- (C) Il modifie la moyenne à long terme.

Bonne réponse : (B)
Explication : $a > 0$ détermine la rapidité avec laquelle le processus revient vers μ, appelé "force de rappel".

Question 3 :

Quelle est la loi conditionnelle de $X_{t_{j+1}}$ dans le processus de Vasicek ?

- (A) Une loi normale dont la variance dépend de Δt.
- (B) Une loi normale centrée en X_{t_j}.
- (C) Une loi exponentielle.

Bonne réponse : (A)
Explication : $X_{t_{j+1}}$ suit une loi normale dont les paramètres (moyenne et variance) dépendent de X_{t_j}, Δt, σ, et a.

2 Exercice: Estimation d'un Zéro Coupon

Exercice 11 (Estimation d'un Zéro Coupon). *Soit $(r_t)_{t\geq 0}$ un processus de Vašíček qui représente le taux instantanné sous la probabilité risque-neutre $\mathbb{Q}$ (on se place en marché complet en absence d'opportunité d'arbitrage). Soit $P(t,T)$ la valeur d'un zéro coupon de maturité T à la date t. On rappelle que*

$$\left(P(t,T)e^{-\int_0^t r_s ds}\right)_{t\geq 0}$$

est une martingale sous $\mathbb{Q}$. En conséquence,

$$P(t,T) = \mathbb{E}^{\mathbb{Q}}\left(e^{-\int_t^T r_s ds} \mid \mathcal{F}_t\right).$$

On se place en date 0 et on observe $r_0 = 0.01$. On pose $a = 0.25, \mu = 0.025, \sigma = 0.0075, T = 2$. On souhaite estimer $P(0,T)$ et on s'appuiera sur une simulation exacte de la trajectoire. On utilisera $n = 10^4$ et $\Delta t = T\times 10^{-3}$. On calculera à chaque fois l'erreur sous forme d'un intervalle de confiance à 95% (sans le biais).

- *Proposez une méthode de Monte Carlo simple pour estimer $P(0,t)$ sans méthode de réduction de variance.*
- *Proposez une méthode de réduction de variance par variable antithétique.*
- *Proposez une méthode de réduction de variance par variable de contrôle.*

Remarque : dans le cas simple de cet exercice, il est possible de calculer explicitement $P(0,T)$, voir en Annexes.

2.1 Lecture et interprétation de l'énoncé

2.1.1 Formule principale

$$P(t,T) = \mathbb{E}^{\mathbb{Q}}\left[\exp\left(-\int_t^T r_s\, ds\right)\Big|\mathcal{F}_t\right]$$

Comment lire cette formule ?

- $P(t,T)$: représente la valeur d'un zéro-coupon à la date t, pour une maturité T. Un zéro-coupon est un produit financier qui ne paie pas d'intérêt périodique mais une valeur nominale (ici 1) à sa maturité.
- $\mathbb{E}^{\mathbb{Q}}$: symbolise l'espérance sous la probabilité risque-neutre $\mathbb{Q}$, qui est utilisée dans les marchés financiers pour éviter les opportunités d'arbitrage.
- $\exp\left(-\int_t^T r_s\, ds\right)$:
  - r_s : taux d'intérêt stochastique à l'instant s.
  - $\int_t^T r_s\, ds$: intégrale de r_s sur l'intervalle $[t,T]$, qui représente l'accumulation des taux d'intérêt dans le temps.
  - $\exp(-x)$: actualise une valeur future en fonction du terme x, ici lié aux taux accumulés.
- $\mathcal{F}_t$: filtration à la date t, qui contient toutes les informations disponibles à cette date.

Interprétation : Le prix du zéro-coupon à la date t est donné par l'espérance actualisée (avec des taux stochastiques) du paiement futur (ici 1) à la maturité T.

2.1.2 Propriété de martingale

$$\left(P(t,T)\exp\left(-\int_0^t r_s\, ds\right)\right)_{t\geq 0} \text{ est une martingale sous } \mathbb{Q}.$$

Comment lire cette formule ?

- **Martingale :** Processus où, à chaque instant, la valeur actuelle est égale à la moyenne conditionnelle de ses valeurs futures, sous $\mathbb{Q}$. Cela reflète l'absence de "profits" attendus dans un marché sans arbitrage.

Interprétation : Cette propriété indique que le produit $P(t,T)$ actualisé par $\exp(-\int_0^t r_s ds)$ ne "dérive" pas dans le temps.

Données de l'exercice

- **Paramètres du modèle de Vasicek :**
  - $r_0 = 0.01$: taux initial (1 %).
  - $a = 0.25$: vitesse de retour à la moyenne, qui contrôle à quelle vitesse r_t converge vers μ.
  - $\mu = 0.025$: moyenne de long terme (2.5 %), le niveau vers lequel r_t converge en moyenne.
  - $\sigma = 0.0075$: volatilité absolue du taux d'intérêt.
- **Simulation :**
  - $n = 10^4$: nombre de trajectoires simulées pour r_t.
  - $\Delta t = T \times 10^{-3}$: pas de temps pour la discrétisation, avec $m = 1000$ points de temps.
- **Intervalle de confiance à 95 % :**
  - La précision de l'estimation $P(0,T)$ sera donnée avec un intervalle de confiance, basé sur la variance empirique des simulations.

2.2 Variables et leur application

Dans l'énoncé, les variables a, μ, σ, T, et r_0 sont directement liées à la formule du modèle de Vasicek utilisée pour simuler les trajectoires r_t. Cette formule se base sur l'équation différentielle stochastique (EDS) suivante :

$$dr_t = a(\mu - r_t)dt + \sigma dW_t$$

- $a = 0.25$ **(vitesse de retour à la moyenne) :**
  - Appliqué dans le terme $a(\mu - r_t)$.
  - Rôle : détermine à quelle vitesse le taux r_t retourne vers la moyenne de long terme μ.
    - Si a est élevé, le retour est rapide.
    - Si a est faible, r_t met plus de temps à converger vers μ.
- $\mu = 0.025$ **(moyenne de long terme) :**
  - Appliqué dans $(\mu - r_t)$.
  - Rôle : représente le niveau auquel r_t tend à se stabiliser en moyenne.
- $\sigma = 0.0075$ **(volatilité absolue) :**
  - Appliqué dans le terme σdW_t.
  - Rôle : mesure l'intensité des fluctuations aléatoires du taux r_t.
  - Une valeur faible ($\sigma = 0.0075$) indique que les variations aléatoires du taux sont limitées.
- $T = 2$ **(maturité) :**
  - Détermine l'horizon temporel de la simulation.
  - Rôle : fixe l'intervalle de temps sur lequel r_t est simulé (de $t = 0$ à $t = T$).
- $r_0 = 0.01$ **(taux initial) :**
  - Condition initiale du processus : r_t commence avec la valeur $r_0 = 0.01$ à $t = 0$.
  - Rôle : point de départ pour simuler les trajectoires.

Résumé des étapes clés

1. Simuler les trajectoires r_t à l'aide du modèle de Vasicek (via une discrétisation numérique de l'EDS).
2. Approximer l'intégrale $\int_0^T r_s ds$ par une somme discrète sur les trajectoires simulées.
3. Utiliser la méthode Monte Carlo pour calculer $P(0,T)$ en moyennant les résultats.
4. Évaluer la précision avec un intervalle de confiance.

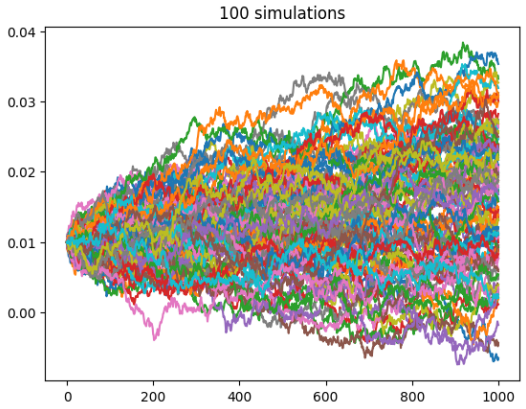
2.3 Code: Création des simulations

```python
n, m = 10**4, 10**3  # Nombre de trajectoires et nombre de pas de discrétisation
T = 2                # Maturité en années
dt = T / m           # Pas de temps entre deux instants discrétisés
sdt = np.sqrt(dt)    # Racine carrée du pas de temps (utile pour la simulation)

# Paramètres du modèle de Vasicek
r0 = 0.01
a = 0.25
mu = 0.025
sigma = 0.0075

R = np.empty((n, m+1), order='F')  # Matrice pour stocker les taux simulés
R[:, 0] = r0    # Initialisation : le taux initial est r0 pour toutes les trajectoires

for j in range(m):
    R[:, j+1] = np.random.normal(
        mu + (R[:, j] - mu) * np.exp(-a * dt),
        sigma * np.sqrt((1 - np.exp(-2 * a * dt)) / (2 * a)),n)
```



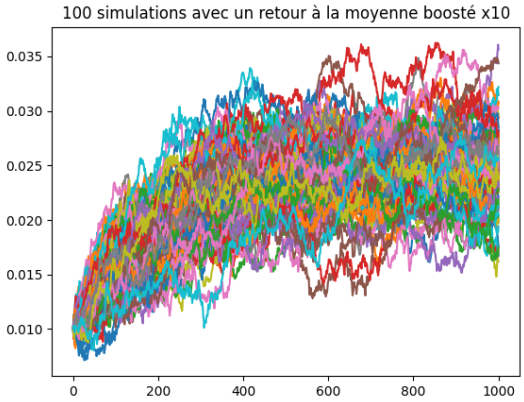


Figure 1: a = 0.25*10

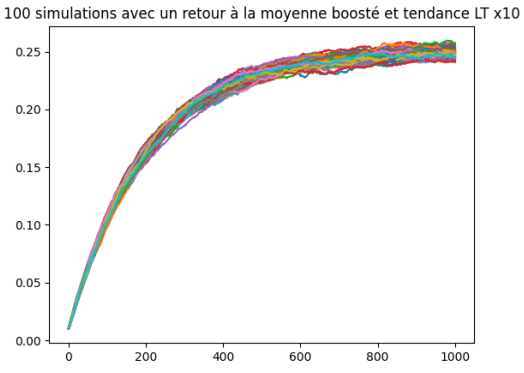


Figure 2: a = 0.25*10 & mu = 0.025*10

2.3.1 Explication de la ligne de code : `R = np.empty((n, m+1), order='F')`

- `np.empty` :
  - Fonction de NumPy utilisée pour créer un tableau vide (non initialisé).
  - Les valeurs initiales du tableau sont arbitraires (aléatoires) car la mémoire n'est pas remplie avec des zéros ou d'autres valeurs prédéfinies.
  - Avantage : cette méthode est plus rapide que `np.zeros` ou `np.ones` car elle ne nécessite aucune initialisation explicite.
- `(n, m+1)` :
  - Définit la forme (ou les dimensions) du tableau.
  - Ici :
    - n est le nombre de trajectoires simulées ($n = 10^4$).
    - $m + 1$ est le nombre de points temporels simulés pour chaque trajectoire :
      - m correspond au nombre de pas de discrétisation (par exemple, $m = 10^3$).
      - On ajoute $+1$ pour inclure le point initial ($t = 0$).
  - Résultat : la matrice R aura n lignes et $m + 1$ colonnes, chaque ligne représentant une trajectoire discrétisée dans le temps.
- `order='F'` :
  - Définit l'ordre de stockage en mémoire du tableau :
    - `'F'` (Fortran-like order) : colonnes consécutives en mémoire.
    - `'C'` (C-like order) : lignes consécutives en mémoire (par défaut).
  - Pourquoi `'F'` ici ?
    - Les calculs sont plus efficaces pour certaines opérations sur des colonnes, comme dans ce cas où chaque colonne représente un instant temporel simulé.

2.3.2 Rôle de cette ligne dans le contexte du code

- **Création de la matrice des trajectoires :**
  - R est la matrice qui va contenir toutes les trajectoires simulées du taux r_t.
  - Chaque ligne correspond à une trajectoire différente (parmi $n = 10^4$).
  - Chaque colonne correspond à un point temporel (parmi $m + 1$).
- **Initialisation pour la simulation :**
  - Après cette ligne, le tableau est prêt à être rempli.
  - La première colonne ($R[:, 0]$) est initialisée avec le taux initial r_0.
  - Les autres colonnes ($R[:, 1], R[:, 2], \ldots, R[:, m]$) seront remplies avec les taux simulés via l'équation de Vasicek.

Exemple concret (dimensions et contenu)

Supposons :

- $n = 4$ (4 trajectoires simulées).
- $m = 3$ (3 pas de discrétisation, donc $m + 1 = 4$ points temporels).

La commande :

```
R = np.empty((4, 4), order='F')
```

crée une matrice R de dimensions $(4, 4)$ (4 lignes et 4 colonnes). Elle ressemble à ceci avant d'être remplie :

$$R = \begin{bmatrix} \text{valeur arbitraire} & \text{valeur arbitraire} & \text{valeur arbitraire} & \text{valeur arbitraire} \\ \text{valeur arbitraire} & \text{valeur arbitraire} & \text{valeur arbitraire} & \text{valeur arbitraire} \\ \text{valeur arbitraire} & \text{valeur arbitraire} & \text{valeur arbitraire} & \text{valeur arbitraire} \\ \text{valeur arbitraire} & \text{valeur arbitraire} & \text{valeur arbitraire} & \text{valeur arbitraire} \end{bmatrix}$$

Ensuite :

- La colonne initiale ($R[:, 0]$) est remplie par r_0.
- Les autres colonnes seront remplies par la simulation.

Pourquoi utiliser `np.empty` ici ?

- **Optimisation des performances :**
  - La fonction `np.empty` est plus rapide que `np.zeros` pour initialiser une matrice.
  - Les valeurs initiales (aléatoires) sont écrasées par la simulation, donc il n'est pas nécessaire de commencer avec des zéros.
- **Gestion efficace de la mémoire :**
  - Avec $n = 10^4$ et $m = 10^3$, la matrice R contient 10^7 valeurs, donc une initialisation rapide est essentielle.

—

2.3.3 Explication de la boucle

```
for j in range(m):
    R[:, j+1] = np.random.normal(
        mu + (R[:, j] - mu) * np.exp(-a * dt),
        sigma * np.sqrt((1 - np.exp(-2 * a * dt)) / (2 * a)),n)
```

2.3.4 Décomposition des éléments du code

- `for j in range(m)` :
  - Cette boucle itère sur les pas de temps j de 0 à $m - 1$, où m est le nombre de pas de discrétisation.
  - Chaque itération correspond au calcul du taux simulé $r_{t+\Delta t}$ pour un instant discrétisé suivant $(t + \Delta t)$.
- `R[:, j+1]` :
  - Cela met à jour la colonne $j + 1$ de la matrice R, correspondant aux valeurs du taux r_t pour toutes les trajectoires (n).
  - $R[:, j]$ est utilisé pour accéder aux valeurs simulées au pas de temps précédent.
- `np.random.normal` :
  - Génère des variables aléatoires suivant une loi normale pour chaque trajectoire (n).
  - Deux paramètres clés sont utilisés :
    - `mu + (R[:, j] - mu) * np.exp(-a * dt)` : moyenne de la distribution.
    - `sigma * np.sqrt((1 - np.exp(-2 * a * dt)) / (2 * a))` : écart-type de la distribution.

2.3.5 Détails des paramètres dans la formule

- `mu + (R[:, j] - mu) * np.exp(-a * dt)` (moyenne) :
  - Représente la valeur attendue du taux $r_{t+\Delta t}$ dans le modèle de Vasicek.
  - mu : moyenne de long terme ($\mu = 0.025$).
  - $(R[:, j] - mu)$: écart par rapport à la moyenne pour le taux au pas précédent.
  - $\exp(-a * dt)$: effet de décroissance vers la moyenne, contrôlé par la vitesse de retour $a = 0.25$ et le pas de temps $\Delta t = T/m$.

- `sigma * np.sqrt((1 - np.exp(-2 * a * dt)) / (2 * a))` (écart-type) :
  - Écart-type de la distribution normale pour le taux $r_{t+\Delta t}$, mesurant les fluctuations stochastiques.
  - *sigma* : volatilité absolue ($\sigma = 0.0075$).
  - $(1 - \exp(-2 * a * dt))/(2 * a)$: facteur de réduction pour les fluctuations aléatoires, dépendant de la vitesse de retour a et du pas de temps Δt.

2.3.6 Rôle de cette ligne dans le contexte du code

- Cette ligne de code met à jour la matrice R avec les taux $r_{t+\Delta t}$ pour toutes les trajectoires simulées.
- Elle applique le modèle de Vasicek pour générer les trajectoires de manière itérative, en tenant compte :
  - de la moyenne conditionnelle $(mu + (R[:, j] - mu) * \exp(-a * dt))$,
  - de la variance conditionnelle $(sigma * \sqrt{((1 - \exp(-2 * a * dt))/(2 * a))})$.

2.3.7 Exemple concret

Supposons :

- $n = 3$ (3 trajectoires simulées),
- $m = 2$ (2 pas de discrétisation, donc $m + 1 = 3$ points temporels),
- $\mu = 0.025$, $a = 0.25$, $\sigma = 0.0075$, $dt = 0.01$.

La ligne suivante :

```
R[:, j+1] = np.random.normal(
    mu + (R[:, j] - mu) * np.exp(-a * dt),
    sigma * np.sqrt((1 - np.exp(-2 * a * dt)) / (2 * a)),n)
```

produira des valeurs $R[:, j+1]$ simulées à partir de $R[:, j]$, par exemple :

$$R = \begin{bmatrix} 0.01 & 0.015 & 0.02 \\ 0.01 & 0.014 & 0.022 \\ 0.01 & 0.016 & 0.019 \end{bmatrix}$$

2.3.8 Pourquoi utiliser cette méthode ?

- Modèle de Vasicek exact : Cette formule intègre la solution analytique de Vasicek pour simuler les trajectoires directement.
- Gestion des fluctuations : Les composantes moyenne et aléatoire permettent de simuler des trajectoires réalistes qui convergent vers μ.

—

2.4 Code: pour estimer $P(0, T)$

```
IntR = dt*np.sum(R, axis=1)

B = np.exp(-IntR)
eP, sdP = np.mean(B), np.std(B)

print("Estimation : %.4f +/- %.5f" % (eP, 2*sdP/np.sqrt(n)))
```

Code analysé :

```
IntR = dt * np.sum(R, axis=1)
B = np.exp(-IntR)
eP, sdP = np.mean(B), np.std(B)
```

1. **Calcul de l'intégrale discrétisée de r_s :**

```
IntR = dt * np.sum(R, axis=1)
```

- R : Matrice contenant les trajectoires simulées du taux r_t, où chaque ligne représente une trajectoire différente.
- `np.sum(R, axis=1)` :
  - Calcule la somme des taux r_t sur toutes les colonnes (tous les pas de temps) pour chaque trajectoire.
  - Cette somme discrétise l'intégrale $\int_0^T r_s ds$.
- `dt` $\times$ `np.sum` :
  - Multiplie cette somme par le pas de temps dt, pour obtenir une approximation de l'intégrale discrète :

$$\int_0^T r_s ds \approx dt \sum_{j=0}^{m} r_j.$$

Lien avec l'exercice : Cette ligne calcule l'intégrale discrétisée $\int_0^T r_s ds$, qui est utilisée dans la formule de $P(0,T)$.

—

2. **Calcul de** $\exp(-\int_0^T r_s ds)$ **pour chaque trajectoire :**

```
B = np.exp(-IntR)
```

- `np.exp(-IntR)` :
  - Applique l'exponentielle à la valeur $-\int_0^T r_s ds$ pour chaque trajectoire.
  - Cela donne une estimation de $\exp(-\int_0^T r_s ds)$ pour chaque trajectoire simulée.

Lien avec l'exercice : Cette étape traduit la formule du zéro-coupon dans le cadre de Monte Carlo :

$$P(0,T) \approx \mathbb{E}^{\mathbb{Q}}\left[\exp\left(-\int_0^T r_s ds\right)\right].$$

—

3. **Calcul de l'espérance et de l'écart-type de** $P(0,T)$ **:**

```
eP, sdP = np.mean(B), np.std(B)
```

- `np.mean(B)` :
  - Calcule la moyenne de B, qui représente l'estimation de $P(0,T)$ via Monte Carlo :

$$P(0,T) \approx \frac{1}{n} \sum_{i=1}^{n} B_i,$$

    où $B_i = \exp(-\int_0^T r_s^{(i)} ds)$ est la valeur simulée pour chaque trajectoire i.
- `np.std(B)` :
  - Calcule l'écart-type empirique de B, utilisé pour évaluer la variance de l'estimation et construire des intervalles de confiance.

Lien avec l'exercice : Ces calculs fournissent :

- L'estimation finale de $P(0,T)$ (valeur moyenne).
- La précision de cette estimation (via l'écart-type).

—

Lien avec la demande de l'exercice :

Le professeur répond à la demande suivante :

- **Proposer une méthode Monte Carlo pour estimer $P(0,T)$ sans méthode de réduction de variance.**

Les étapes réalisées dans ce code correspondent exactement à cette demande :

1. Simulation des trajectoires r_t (fait en amont du code présenté).
2. Approximation de l'intégrale $\int_0^T r_s ds$ par une somme discrète.
3. Calcul de $\exp(-\int_0^T r_s ds)$ pour chaque trajectoire.
4. Moyennage pour estimer $P(0,T)$.

3 Comparaison des deux codes : Méthode Antithétique et Méthode Variable de Contrôle

Voici une comparaison détaillée des deux codes, incluant les **formules mathématiques** et le **code associé** pour chaque méthode. Cette comparaison met en évidence les différences clés entre la méthode antithétique et la méthode de variable de contrôle.

1. Simulation des trajectoires

Méthode Antithétique

Formule mathématique :

$$r_{t+\Delta t} = \mu + (r_t - \mu)e^{-a\Delta t} + Z$$

$$r_{t+\Delta t}^{\text{antithétique}} = \mu + (r_t - \mu)e^{-a\Delta t} - Z$$

Code associé :

```
for j in range(m):
    Z = np.random.normal(0., Sigma, n)  # Composante stochastique
    R, RA = mu + (R - mu) * expmadt + Z, mu + (RA - mu) * expmadt - Z  # Trajectoires normales et antithét
    IntR += R  # Accumuler l'intégrale pour R
    IntRA += RA  # Accumuler l'intégrale pour RA
```

Méthode Variable de Contrôle

Formule mathématique :

$$r_{t+\Delta t} = \mu + (r_t - \mu)e^{-a\Delta t} + Z$$

Code associé :

```
for j in range(m):
    Z = np.random.normal(0., Sigma, n)  # Composante stochastique
    R = mu + (R - mu) * expmadt + Z  # Mise à jour des trajectoires normales
    IntR += R  # Accumuler l'intégrale pour R
```

2. Intégrale discrète $\int_0^T r_s ds$

Méthode Antithétique

Formule mathématique :

$$\int_0^T r_s^{\text{normal}} ds \approx \Delta t \sum_{j=1}^{m} r_j$$

$$\int_0^T r_s^{\text{antithétique}} ds \approx \Delta t \sum_{j=1}^{m} r_j^{\text{antithétique}}$$

Code associé :

```
IntR *= dt  # Ajustement final pour R
IntRA *= dt  # Ajustement final pour RA
```

Méthode Variable de Contrôle

Formule mathématique :

$$\int_0^T r_s ds \approx \Delta t \sum_{j=1}^{m} r_j$$

Code associé :

```
IntR *= dt  # Ajustement final pour R
```

3. Facteur d'actualisation

Méthode Antithétique

Formule mathématique :

$$B = \exp\left(-\int_0^T r_s^{\text{normal}} ds\right), \quad BA = \exp\left(-\int_0^T r_s^{\text{antithétique}} ds\right)$$

Code associé :

```
B, BA = np.exp(-IntR), np.exp(-IntRA)  # Facteurs d'actualisation pour R et RA
```

Méthode Variable de Contrôle

Formule mathématique :

$$B = \exp\left(-\int_0^T r_s ds\right)$$

Code associé :

```
B = np.exp(-IntR)  # Facteur d'actualisation pour R
```

4. Estimation et réduction de variance

Méthode Antithétique

Formule mathématique :

$$P(0,T) = \frac{1}{2}\left(\mathbb{E}[B] + \mathbb{E}[BA]\right)$$

Code associé :

```
ePA = 0.5 * np.mean(B + BA)  # Estimation moyenne combinée
sdPA = 0.5 * np.std(B + BA)  # Écart-type combiné
```

Méthode Variable de Contrôle

Formule mathématique :

$$a^* = \frac{\text{Cov}(B, \int_0^T r_s ds)}{\text{Var}(\int_0^T r_s ds)}$$

$$C = B - a^*\left(\int_0^T r_s ds - \mathbb{E}\left[\int_0^T r_s ds\right]\right)$$

Code associé :

```
COV = np.cov(B, IntR)  # Covariance entre B et IntR
a_star = COV[0, 1] / COV[1, 1]  # Coefficient optimal
eC = mu * T + (r0 - mu) / a * (1 - np.exp(-a * T))  # Espérance analytique de IntR
C = B - a_star * (IntR - eC)  # Correction par variable de contrôle
ePC, sdPC = np.mean(C), np.std(C)  # Moyenne et écart-type après contrôle
```

5. Résultats affichés

Méthode Antithétique

Formule mathématique :

$$\text{Intervalle de confiance} = P(0,T) \pm 2 \times \frac{\text{Écart-type}}{\sqrt{n}}$$

Code associé :

```
print("Estimation Antithétique : %.8f +/- %.8f" % (ePA, 2 * sdPA / np.sqrt(n)))
print("Amélioration Antithétique : %.1f, soit %.0f moins de simulations" % (sdP / sdPA, (sdP / sdPA)**2))
```

Méthode Variable de Contrôle

Formule mathématique :

$$\text{Intervalle de confiance} = P(0,T) \pm 2 \times \frac{\text{Écart-type}}{\sqrt{n}}$$

Code associé :

```
print("Estimation Contrôlée : %.8f +/- %.8f" % (ePC, 2 * sdPC / np.sqrt(n)))
print("Amélioration Contrôlée : %.1f, soit %.0f moins de simulations" % (sdP / sdPC, (sdP / sdPC)**2))
```

4 Pourquoi le modèle de Vasicek est adapté à la simulation d'un Zéro-coupon ?

a. Réversion vers la moyenne :

Le modèle de Vasicek :

$$dr_t = a(\mu - r_t)dt + \sigma dW_t,$$

implémente un terme de réversion ($a(\mu - r_t)$) qui force le taux r_t à revenir vers une moyenne de long terme μ au fil du temps.

Ce comportement est cohérent avec les observations empiriques des taux d'intérêt, qui fluctuent autour d'un niveau moyen plutôt que de dériver librement.

Pourquoi c'est utile pour un zéro-coupon ?

La valeur du zéro-coupon dépend directement des taux futurs. Un modèle sans réversion pourrait produire des trajectoires irréalistes (par exemple, des taux qui explosent ou dérivent trop loin), rendant l'estimation du prix peu fiable.

b. Solution analytique exacte pour $P(t,T)$:

Le modèle de Vasicek permet de calculer la valeur d'un zéro-coupon à partir d'une solution analytique de $P(t,T)$ sous certaines hypothèses :

$$P(t,T) = \exp(A(t,T) - B(t,T)r_t),$$

où $A(t,T)$ et $B(t,T)$ sont des fonctions déterminées par les paramètres a, μ, et σ.

Avantage :

Cela facilite la simulation et la validation des résultats. Une EDS classique n'offre pas nécessairement une telle solution fermée, ce qui compliquerait l'estimation des prix de zéro-coupon.

c. Taux négatifs évités grâce à la calibration :

Bien que le modèle de Vasicek puisse produire des taux négatifs en théorie, cela est rarement observé en pratique grâce à une calibration appropriée des paramètres (a, μ, σ).

Une EDS classique, en revanche, pourrait produire des trajectoires irréalistes beaucoup plus fréquemment.

Pourquoi c'est important pour un zéro-coupon ?

Les taux négatifs ou irréalistes faussent le calcul du facteur d'actualisation

$$\exp\left(-\int_t^T r_s ds\right),$$

qui dépend directement des trajectoires simulées.

d. Modèle simple et bien compris :

Vasicek est un modèle simple à simuler et largement étudié en finance, ce qui en fait un bon choix pour les applications pédagogiques (comme cet exercice).

Les paramètres a, μ, σ sont facilement interprétables :

- a : vitesse de retour à la moyenne.
- μ : niveau moyen du taux.
- σ : volatilité.