WSTĘP DO MATEMATYKI FINANSOWEJ

BARTOSZ KOŁODZIEJEK

WYDZIAŁ MATEMATYKI I NAUK INFORMACYJNYCH

Laboratoria 5

Projekt „NERW 2 PW. Nauka - Edukacja - Rozwój - Współpraca" współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego.

Zadanie 10 pn. „Modyfikacja programów studiów na kierunkach prowadzonych przez Wydział Matematyki i Nauk Informacyjnych", realizowane w ramach projektu „NERW 2 PW. Nauka - Edukacja - Rozwój - Współpraca", współfinansowanego jest ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego.

April 24, 2025

Legenda: 📘 – Definicja, **TW.** – Twierdzenie, ⚙ – Przykład, ⚠ – Uwaga, **LEM.** – Lemat, 👁 – Oznaczenie, 💣 – dla chętnych (może być trudne)

1. L5 - Monte Carlo do wyceny w modelu CRR

Przypomnij sobie model CRR z pliku `L3.pdf`. Przypomnij sobie metody Monte Carlo z pliku `L4.pdf`.

(1) Aby dokładnie obliczyć wartość wypłaty na rynku CRR, która potencjalnie może zależeć od całej trajektorii, trzeba przejrzeć wszystkie trajektorie. Dla większych wartości T, szybko robi się to bardzo wymagające obliczeniowo.
(2) Można stosować metody Monte Carlo do estymacji wartości wypłaty. Dzięki temu wynik otrzymamy dużo szybciej, choć nie będziemy mieli pewności co do jego poprawności.
(3) MPWL gwarantuje, że estymator

$$\hat{V}_{\mathrm{MC}}(M) \;=\; \frac{1}{(1+r)^T}\frac{1}{M}\sum_{i=1}^{M}\mathrm{payoff}(\{S_t^{(i)}\}_{t=0}^{T})$$

zbiega się do wartości oczekiwanej wypłaty wraz ze wzrostem liczby ścieżek M, a typowy błąd maleje jak $O(1/\sqrt{M})$.

2. Zadania

2.1. **Implementacja funkcji Monte Carlo.** Do swojej implementacji Python z rynkiem CRR dopisz funkcję `evaluate_mc(M)` z odpowiednimi dodatkowymi parametrami (zależnymi od Twojej implementacji), która:

- Generuje M ścieżek $(S_t^{(i)})_{t=0}^{T}$ według modelu CRR (z prawdopodobieństwem martyngałowym).
- Liczy payoff na końcu każdej ścieżki.
- Zwraca estymowaną wartość $\hat{V}_{\mathrm{MC}}(M) = \dfrac{1}{(1+r)^T}\dfrac{1}{M}\sum_{i=1}^{M}\mathrm{payoff}((S_t^{(i)})_{t=0}^{T})$.

2.2. **Eksperyment zbieżności.** Wyceń wypłatę $X = \max_{t\in T}\{S_t\}$.

1) Oblicz wartość dokładną V_{exact} za pomocą kodu z L3.
2) Narysuj wykres zbieżności estymatora $\hat{V}_{\mathrm{MC}}(M)$, czyli na osi OX liczbę wylosowanych ścieżek M, a na osi OY wartość $\hat{V}_{\mathrm{MC}}(M)$ obliczona na podstawie pierwszych M ścieżek.
3) Na wykresie dodatkowo zaznacz V_{exact}. Zinterpretuj.
4) Porównaj szybkość zbieżności dla różnych T. Mały i duży (duży T to taki, dla którego Twoja implementacja jeszcze pozwala na obliczenie dokładnej wartości).
5) Wyestymuj przy pomocy Monte Carlo wartość dla T tak dużego, że Twoja implementacja nie pozwala na znalezienie dokładnej wartości, ale pozwala na ustabilizowanie estymatora Monte Carlo.

2.3. 💣**Szybkość obliczeń.**

- Czy implementacja pętli w numpy zamiast w Python wpływa na szybkość obliczeń?
- Czy implementacja w C (Cython ipt.) poprawia szybkość obliczeń?
- Czy można zrównoleglić obliczenia? (użyj np. moduł `multiprocessing` lub `joblib`)

Użyj `timeit.timeit(...)` lub `%timeit` do precyzyjnego pomiaru czasu wykonania.