Binomiale bomen

J. Tempere, cursus Padintegralen voor Optieprijzen

Maak 3 programma's die optieprijzen berekenen via binomiale bomen. Elk van die programma's moet zowel call als put opties aankunnen, met willekeurige looptijd T en strike K/S_0. De parameters r, σ van het model moeten ingegeven kunnen worden, alsook het aantal intervallen N waarin je de looptijd T wil opdelen. Hiermee bepaal je natuurlijk de tijdsstap $\delta t = T/N$, de stapgrootte $\delta x = \sigma\sqrt{\delta t}$. Dit bepaalt ook de kans $p = \left(e^{r\delta t} - e^{-\sigma\sqrt{\delta t}}\right) / \left(e^{\sigma\sqrt{\delta t}} - e^{-\sigma\sqrt{\delta t}}\right)$ om een stapje omhoog te zetten in de boom.

PROGRAMMA 1. Via de "eindige differentie methode" (discretisatie van de differentiaalvergelijking)

- Je programma berekent en bewaart eerst de optieprijzen op de eindtijd $t_N = T = N\delta t$ voor elk van de $N+1$ eindpunten van de vertakkingen van de binomiale boom.
- Dan stapt het programma in stapjes δt naar eerdere tijden, waarbij het de formule $V(x, t-\delta t) = [pV(x+\delta x, t) + (1-p)V(x-\delta x, t)]\, e^{-r\delta t}$ gebruikt. De array met x-waarden op tijdstip t_n is $n+1$ entries groot, wanneer je t_0 bereikt is de overgebleven waarde $V(0,0)$ de prijs van de optie.

PROGRAMMA 2. Via het "sommeren over paden". Later gaan we zien hoe we padintegralen analytisch uitrekenen, maar je kan ook via brute kracht alle paden opsommen en er over uitmiddelen.

Je programma doorloopt alle 2^N paden. Bvb. voor een driestapsboom zijn er acht paden, genummerd van 0 tot 7. Pad nummer 5 is in binair 101, wat een "udu" pad voorstelt dat dus eerst op gaat, dan neer, dan terug op. Dus "1" is op, "0" is neer. De som

$$\sum_{j=0}^{2^N-1} P(\text{pad } j) V(\text{pad } j)$$

bevat de kans dat het pad j gerealiseerd wordt, en de waarde van de optie wanneer pad j gevolgd wordt. Aangezien het pad gemaakt wordt uit onderling onafhankelijke random walks hebben we

$$P(\text{pad } j) = p^{\#\text{ stappen op}} \times (1-p)^{\#\text{ stappen neer}}$$

De waarde van de optie is in het algemeen een functie van het ganse pad, maar voor de plain vanilla call/put hangt het enkel van het laatste punt af. De sommatie geeft de verwachte prijs van de optie, die je nog moet verdisconteren met e^{-rT}.

PROGRAMMA 3. Via een Monte-Carlo simulatie. Wanneer N groot wordt, wordt het ondoenlijk om te sommeren over alle 2^N paden. We gaan daarom een Monte-Carlo algoritme gebruiken, en random n paden genereren (met n natuurlijk kleiner dan 2^N).

Je programma genereert n paden, waarbij het dit doet door N random getallen te nemen tussen 0 en 1. Als een getal kleiner is dan p dan neem je een stapje omhoog, anders ga je omlaag. De N random getallen geven je een random wandeling die een pad voorstelt. Je berekent voor dat pad de prijs van de optie. Je sommeert alle n op deze manier bekomen prijzen en deelt door n. Dit geeft de verwachte prijs van de optie, die je nog moet verdisconteren met e^{-rT}.

BIJVRAAG: veralgemeen je programma zodat het een barrière optie kan prijzen: dit is een call of put optie waarbij de waarde van de optie nul wordt wanneer het onderliggende boven een bepaalde barrière B/S_0 uitstijgt (call) of onder een bepaalde barriere B/S_0 daalt (put).