Lecture 16 (10/8). Please enable your video if you can.

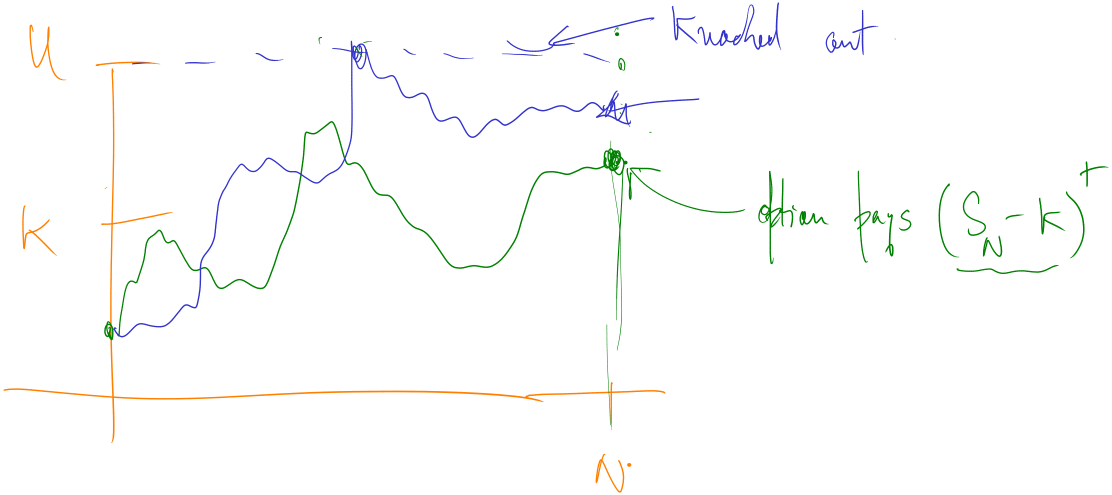
Last time: ① AFP formula: $V_n = \frac{1}{D_n} \tilde{E}_n (D_N V_N)$ ⟵ takes $O(2^N)$ time to compute

② If $\boxed{V_N = g(S_N)}$ ⟶ then can compute in time $O(N^2)$

Claim: $V_n = f_n(S_n)$ & $f_n(x) = \frac{1}{1+r}\left(f_{n+1}(ux)\tilde{p} + f_{n+1}(dx)\tilde{q}\right)$

$(x \in \text{Range}(S_n))$

Example 6.19 (Knockout options). An *up and out call* option with strike K and barrier U and maturity N gives the holder the option (not obligation) to buy the stock at price K at maturity time N, *provided* the stock price has never exceeded the barrier price U. If the stock price exceeds the barrier U before maturity, the option is worthless. Find an efficient algorithm to price this option.



① Payoff of option: Let $M_N = \max_{n \leq N} S_n$

$$V_N = \text{Payoff of the up \& out option} = \begin{cases} (S_N - K)^+ & M_N \leq U \\ 0 & M_N > U \end{cases}$$

$$\boxed{V_N = \mathbb{1}_{\{M_N \leq U\}} (S_N - K)^+ .}$$

(Notation: if $A \subseteq \Omega$, define $\mathbb{1}_A$ to be the random variable which takes value 1 on the event A & 0 outside A.

i.e $$\mathbb{1}_A(\omega) = \begin{cases} 1 & \omega \in A \\ 0 & \omega \notin A. \end{cases}$$

$$\Rightarrow \mathbb{1}_{\{M_N \le u\}} (S_N - K)^+ = \begin{cases} (S_N - K)^+ & \text{on } \{M_N \le u\}, \\ 0 & \text{on } \{M_N \le u\}^c \end{cases}$$

u

Idea: Let $M_n = \max_{k \leq n} S_k$ (M_n is an adapted process)

$\boxed{\text{Make}}$ $V_n = f_n(M_n, S_n)$ & find a recurrence rel for f_n.

↳ ① Know for $n = N$, $V_N = f_N(M_N, S_N)$

where $f_N(m, s) = \begin{cases} (s-K)^+ & m \leq U \\ 0 & m > U \end{cases}$

② Backward induction: Suppose for time $n+1$ we know

$$V_{n+1} = \underbrace{f_{n+1}(M_{n+1}, S_{n+1})}$$

want $V_n = f_n(M_n, S_n)$ for some f_n f_n.

Know $V_n = \frac{1}{D_n} \tilde{E}_n (D_{n+1} V_{n+1}) = \frac{1}{1+r} \tilde{E}_n V_{n+1}$

$$= \frac{1}{1+r} \tilde{E}_n \overbrace{f_{n+1}(M_{n+1}, S_{n+1})}$$

$\underline{\underline{\text{Want}}}$ $f_n(M_n, S_n)$

To find f_n write M_{n+1} & S_{n+1} in terms of M_n & S_n.

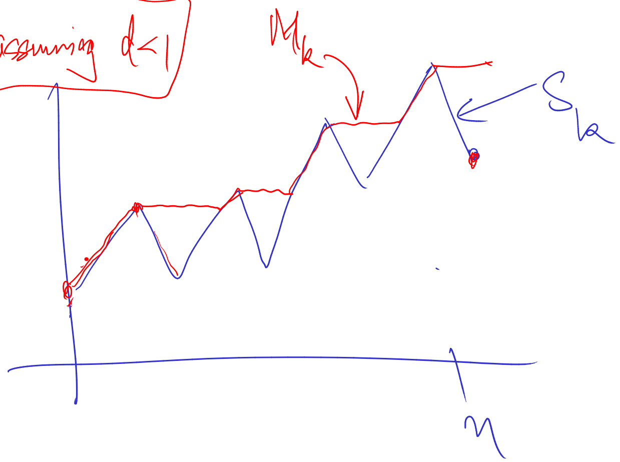
$$S_{n+1} = \begin{cases} uS_n & \omega_{n+1} = 1 \\ dS_n & \omega_{n+1} = -1 \end{cases} \Rightarrow S_{n+1} = X_{n+1} S_n, \text{ when } X_{n+1} = \begin{cases} u & \omega_{n+1} = 1 \\ d & \omega_{n+1} = -1 \end{cases}$$

$$M_{n+1} = \begin{cases} \max\{M_n, uS_n\} & \text{if } \omega_{n+1} = +1 \end{cases}$$

$\Big(\quad M_n$

otherwise. $\boxed{\text{assuming } d<1}$

M_k



$$\Rightarrow M_{n+1} = \Big(M_n \overset{\text{maximum}}{\vee} uS_n\Big)\,\mathbb{1}_{\{\omega_{n+1}=1\}} + M_n\,\mathbb{1}_{\{\omega_{n+1}=-1\}}$$

$$\Rightarrow V_n = \frac{1}{1+r}\,\tilde{E}_n\, v_{n+1}\big(M_{n+1}, S_{n+1}\big)$$

$$= \frac{1}{1+r}\,\tilde{E}_n\, f_{n+1}\Big(M_n v(uS_n)\,\mathbb{1}_{\{\omega_{n+1}=1\}} + M_n\,\mathbb{1}_{\{\omega_{n+1}=-1\}}\,,\; X_{n+1} S_n \Big)$$

indep lemma

$\mathcal{F}_n$- meas

$\mathcal{F}_n$- indep

$$= \frac{1}{1+r}\Big(\tilde{p}\, f_{n+1}\big(M_n v(uS_n)\cdot 1 + 0,\; uS_n\big)$$

$$+ \tilde{q}\, f_{n+1}\big(M_n v(uS_n)\cdot 0 + M_n,\; dS_n\big)\Big)$$

$$V_n = \frac{1}{1+r}\left(\tilde{p}\, f_{n+1}\left(M_n v(uS_n), uS_n\right) + \tilde{q}\, f_{n+1}\left(M_n, dS_n\right)\right)$$

$$\Rightarrow V_n = f_n(\underbrace{M_n, S_n}_{\text{process } Y}) \quad \text{where} \quad \boxed{f_n(m, s) = \frac{1}{1+r}\left(\tilde{p}\, f_{n+1}(m\, v(us), us) + \tilde{q}\, f_{n+1}(m, ds)\right)}$$

Gives me a recursive relation to compute f_n in terms of f_{n+1}

Definition 6.20. We say a d-dimensional process $Y = (Y^1, \ldots, Y^d)$ process is a *state process* if for any security with maturity $m \leqslant N$, and payoff of the form $V_m = f_m(Y_m)$ for some (non-random) function f_m, the arbitrage free price must also be of the form $V_n = f_n(Y_n)$ for some (non-random) function f_n.

Remark 6.21. For state processes given f_N, we typically find f_n by backward induction. The number of computations at time n is of order Range(Y_n).

Remark 6.22. The fact that S_n is Markov (under $\tilde{\boldsymbol{P}}$) implies that it is a state process.

E.g. $Y_n = (S_n)$ is a state process.

Eg: $M_n = \max_{k \leq n} S_k$ & set $\left.\begin{array}{l} Y^1_n = M_n \\ Y^2_n = S_n \end{array}\right\}$ $Y = (Y^1, Y^2)$

Y is a state process (as follows from Knock out eg.)