The Cooper Union Department of Electrical Engineering
Prof. Fred L. Fontaine
ECE478 Financial Signal Processing
Problem Set III: Binomial Asset Pricing Model
February 28, 2022

Write code to simulate the BAPM and perform analysis as suggested. Notation and so forth follows as given in Shreve, *Stochastic Calculus for Finance vol. 1.* Your simulation should take r, d, u as given, and constant at all time steps. Building in error detection to ensure these satisfy the no-arbitrage constraint is optional. Might as well take $S_0 = 1$. We will assume the coin toss distribution is constant over time, but the particular (p, q) values can be variable. Two special cases are the "actual" probabilities (p_0, q_0), and the risk-neutral measure $(\tilde{p}, \tilde{q})$. Let $\mathcal{F}_n$ be the σ-algebra generated by the first n tosses, which covers the time span $\{0 \leq k \leq n\}$. Let $\vec{\omega}_n = (\omega_1 \cdots \omega_n) \in \{H, T\}^n$ denote a particular 'path' through the first n tosses. Assume every stochastic process here, say X_n is adapted to this filtration. Your model should cover the time steps $\{0 \leq n \leq N\}$. When you perform a Monte Carlo simulation, M is the number of samples used (i.e., for purposes of Monte Carlo, you repeat the experiment M times and average over those results).

There are two types of derivatives: one type has a payout $V_N = V_N(S_N)$ that depends only on the final value of the security; this is called 'path independent'. Another type can depend on the values of the security over the whole time span, i.e., $V_N = V_N(S_0, S_1, \cdots, S_N)$. Here we will only build out simulation for the case of path-independent payouts, so feel free to write your code assuming that condition holds. [A more complete BAPM simulation would have to be able to handle the more general case.]

Finally, in what follows, for any stochastic process X_n, the *discounted* process is $\tilde{X}_n = \frac{1}{(1+r)^n} X_n$.

1. **Distributions:** Before you start coding: Assuming the coin-toss distribution (p, q), find the distribution of S_n. That is, list all the possible values (there are $n+1$ of them), and their probabilities. Also, if $R_n = \log(S_n/S_0)$, show that $R_n = c_n Y_n + d_n$ where $Y_n \sim$binomial (n, p) (find the constants c_n, d_n). **Note:** In the continuous-time case, something similar will occur: we will get $R_t = \log(S_t/S_0) \sim$ Gaussian (what else?), and hence the distribution of S_t will be what is called lognormal.

2. **Exact simulation:** Assume we have code to compute the payout function $V(S_N)$ for a path-independent derivative.

   (a) Write code to compute $V_0 = E_{\tilde{p}}\left(\tilde{V}_N\right)$ using the known distribution for S_N.

   (b) Write code to generate one step of the replicating portfolio. If at time n there are Δ_n shares of the stock, and wealth X_n, then the amount $M_n = X_n - \Delta_n S_n$ is held in the money market. From $n \longrightarrow n+1$, the stock price changes from S_n to S_{n+1}, and the amount in the money market grows to $(1+r) M_n$. Thus, at time $n+1$, the *wealth equation* states:

$$X_{n+1} = \Delta_n S_{n+1} + (1+r)(X_n - \Delta_n S_n)$$

For this to be a replicating portfolio, $X_n = V_n$, the price of the derivative at time n, where at $n = N$ this should match$V_N(S_N)$ exactly. Give $\vec{\omega}_n$, $0 \le n \le N-1$, there are two possibities for $\vec{\omega}_{n+1}$, namely $(\vec{\omega}_n H)$ and $(\vec{\omega}_n T)$. Assuming $V_{n+1}(\vec{\omega}_n H)$, $V_{n+1}(\vec{\omega}_n T)$ are both known, and of course $S_{n+1}(\vec{\omega}_n H), S_{n+1}(\vec{\omega}_n T), S_n(\vec{\omega}_n)$ are all known, with given r, the wealth equation yields two linear equations in the unknowns $\Delta_n(\vec{\omega}_n), X_n = V_n(\vec{\omega}_n)$. By running this recursively backwards, you should be able to derive $\{\Delta_n(\vec{\omega}_n)\}_{\text{all } \vec{\omega}_n, \text{ for } 0 \le n \le N-1}$, and of course $X_0 = V_0$. Write code to do this! Also as you do this, keep track of whether any Δ_n or $X_n - \Delta_n S_n$ values are negative (the first is short selling the stock, the second is "short selling the money market", i.e., borrowing money to buy stock).

(c) Now let us take a specific example to test your code. Take $r = 0.05$, $u = 1.1$, $d = 1.01$, $N = 5$, and the derivative a European call option with strike price $K = E_{\tilde{p}}(S_N)$ (you know the distribution of S_N so should be able to find a simple formula for this!) There are only $2^N = 32$ paths so it should not be unreasonable to compute and store the complete set of replicating portfolio values $\{\Delta_n(\vec{\omega}_n)\}$. In any case, compute V_0 using each of your two methods, and check that they match! **Question:** Do you have to do any short selling the stock or borrowing from the money market along the way?

3. **Monte Carlo simulation:** Next is to deal with the case $N = 100$. There are 2^{100} paths, so rather than computing over all paths, we will use a Monte Carlo approach. So let me first describe the code you should create. For fixed m, and an adapted stochastic process X_n, we want to compute $Y_n = E_{\tilde{p}}(X_{n+m}|\mathcal{F}_n)$. What this means is $Y_n = Y_n(\vec{\omega}_n)$, a function of the path associated with the first n tosses. So assume that is given. You will generate M random paths $(\omega_{n+1} \cdots \omega_{n+m})$ according to the distribution, compute $X_{n+m}(\vec{\omega}_n \omega_{n+1} \cdots \omega_{n+m})$ for each, and average over them to get Y_n. The details of computing X_{n+m} should be contained in a function that is called by the Monte Carlo "wrapper". It is ok if you have to modify your wrapper code a little bit in terms of the variables you pass to or receive back from this core function.

(a) As a first step, let us check the underlying behavior of your Monte Carlo code. Go back to the previous case, with $N = 5$ and r, u, d as given. Taking $M = 1, 5, 10, 32$, estimate S_0 from S_N, and V_0 from V_N. Note that you are not going through every path systematically! You are generating paths independently, and so it is possible (especially in the low order case) the same path will occur multiple times, so even with $M = 32$ there is no guarantee you will get all paths! Hence your S_0, V_0 estimate will not be exact. The idea is to see how close your estimates are for various M.

(b) Now take the case $N = 100$. In this case, we should change some of the parameters. Take $r = 10^{-3}$ (so over 100 steps there is a total return of about 10%), $u = 1 + 5 \times 10^{-3}$, $d = 1 + 10^{-4}$. Take different values of M, starting small so your computer doesn't take too long to run, and increasing it somewhat. We want to hopefully see an effect of increasing M, without forcing you to run simulations overnight! First, as a check, we know we should get $S_0 = E_{\tilde{p}}\left(\tilde{S}_N\right)$ and fortunately we know $S_0 = 1$. See how this works for various M. Next, again let $V(S_N)$

be a European call with $K = E_{\tilde{p}}(S_N)$ [Note: Careful, this is not the discounted value, since the strike price applies at tine N; also we are computing the expectation assuming $\tilde{p}$; there is nothing formal here, this is just a SUGGESTION I am making so you have a reasonable value K that lies somewhere between $\min S_N$ and $\max S_N$] Compute $V_0 = E_{\tilde{p}}\left(\tilde{V}_N\right)$ for the same M as before. Also compute V_{10} assuming the first ten tosses are all heads, and again assuming all are tails. Last thing to do: instead of using $\tilde{p}$, take two cases for the "actual" probabilities: $p_0 = 0.9\tilde{p}$, and $p_0 = 1.1\tilde{p}$. For each case, compute $E_{p_0}\left(\tilde{S}_N\right)$, and compare with S_0, and $E_{p_0}\left(\tilde{V}_N\right)$. In the latter case, if the answer with p_0 is below V_0 then you should not have purchased the option (putting the cash into a money market would have been better), if above V_0 then purchasing the option gave you a higher return than stashing you money in the money market :-D So last question: even if I told you that your **expected return** was better than putting it into a money market, you may still have decided not to purchase this option. Why not? **Hint:** Even if you know p_0 (which in this case would lead to this result), we still don't have an arbitrage here. What does that mean?