Assignment 4, due March 14

Corrections: (March 8: Black Scholes formula included (page 5), March 13, Problem 2 formulas corrected to be $M'(0)$ instead of $M'(\lambda)$, etc.)

1. **True/False**. In each case, state whether the statement is true or false and explain your answer in a few words or sentences
   (a) Vanilla puts and calls are publicly traded on exchanges and have market prices.
   (b) The CRR or Black Scholes theoretical option price is a function of market parameters (volatility), not just market prices.
   (c) The theoretical (CRR or Black Scholes) price of a put or call option does not depend on whether the underlier pays a dividend.
   (d) If the risk free rate is not constant in time, then the theoretical price of a European option that expires at time T depends on Z_T (the price today of a zero coupon bond that matures at time T).

Review of probability

2. Suppose that X is a random variable with probability density $p(x)$. The *moment generating function* function is

$$M(\lambda) = \mathrm{E}\big[\, e^{\lambda X} \big] = \int_{-\infty}^{\infty} e^{\lambda x} p(x)\, dx \; .$$

Assume that the integral defining $M(\lambda)$ converges for all λ. Show that

$$M(0) = 1 \; , \quad M'(0) = \mathrm{E}[X] \; , \quad M''(0) = \mathrm{E}\big[\, X^2 \big] \; , \quad \text{etc.} \tag{1}$$

(The expected values of powers of X are called *moments* (don't know why). The expected value of the exponential "generates" the moments through differentiation.) Don't worry about the mathematical rigor unless you feel like it.

3. Suppose X is uniformly distributed in $[-1, 1]$. What is the probability density of X? Find $M(\lambda)$ for this density (calculate the integral). Check that the formulas (1) are satisfied up to the second moment (which is $\mathrm{E}\big[\, X^2 \big]$).

4. Suppose that X is a Gaussian random variable with mean μ and variance σ^2. The probability density of X is

$$p(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \; .$$

You may use the formula

$$\int_{-\infty}^{\infty} e^{-\frac{z^2}{2}}\, dz = \sqrt{2\pi}\ .$$

Calculate the moment generating function. (Hint: use a linear change of variables $x = az + b$ or $z = ax + b$, whichever is more convenient, so the integral becomes $C\int_{-\infty}^{\infty} e^{-\frac{z^2}{2}}\, dz$.) Verify the formulas (1) up to the second moment.

5. We say that a random variable S is *lognormal* if $X = \log(S)$ is normal. In particular, S is lognormal with parameters μ and σ if $X = \log(S)$ is normal with mean μ and variance σ^2. It is possible to write a formula for the probability density $p(s)$, but that is not the easiest way to answer questions about the lognormal.

   (a) Find formulas for the mean and variance of S in terms of the parameters μ and σ. (Hint: $S = e^X$. Use the Gaussian moment generating function.)

   (b) For what values of λ is the moment generating function defined?

6. The *cumulative normal*, or the cumulative normal *distribution function*, is

$$N(a) = \Pr(Z < a) = \int_{-\infty}^{a} e^{-\frac{z^2}{2}}\, dz\ .$$

   In this formula Z is a *standard normal* random variable, which means Gaussian (normal) with mean 0 (standard) and variance 1 (standard).

   (a) Suppose X is Gaussian with mean μ and variance σ^2. Find a formula for $\Pr(X < a)$ in terms of μ, σ, a and the cumulative normal. Explain the statement that $\Pr(X < a)$ depends on the number of standard deviations X is from the mean.

   (b) Suppose S is lognormal. Suppose $K > 0$. Recall the notation $(S - K)_+$ for the *positive part* of $S - K$. This is $S - K$ if $S > K$ and zero if $S < K$. Find a formula for $\mathrm{E}[(S - K)_+]$ if S has parameters μ and σ. This is the *Black Scholes* formula for pricing a European style call option.

Computing problem. This exercise asks you to build software that finds theoretical option prices for European or American style puts. This code will require several functions, defined in different script files, to work together. It is the first experience in this class of software engineering. You "build" (code) the pieces separately and then assemble them. You test each piece separately. Any ten lines of code, as first written, is likely to have a bug. Make that five lines if there is math – formulas and such – or complicated array indexing. This software involves both. For each **step** below, print the testing code (probably just a few lines) and some output showing that the function being tested works.

Step 1, write a function that evaluates the intrinsic value of the put option. This takes argument S (the price of the underlier) and K (the strike price) and returns $(K-S)_+$. Write an `R` script that `calls` this function (uses it in a command) to evaluate the intrinsic value at many points (maybe 100 or 1000?) in a reasonable range around K and makes a plot. The handout `Plotting` explains how to do this. Put the parameter K in the title of the plot.

Step 2, write a function that evaluates the Black Scholes formula for the theoretical price of the same put. You can get the formula from the book or from online web sources. If you use the book, you will have to use put-call parity since only the call formula is given. Put this plot in the same graph with the intrinsic value so you can see the difference. Take volatility $\sigma = .4$ (40%/year), $r = .02$, and $T = .25$ (a three month option) or $T = .5$ (a six month option). Make some plots with various values of T and r. Describe how the Black Scholes put value depends on T and r. Put the parameters T, r, σ, and K in the title or subtitle. Make the legend explain which curve is which. Automation is key. It will take time to get the code to work. But once it works, it will be very easy to play with.

Step 3, build a function that does one time step of the binomial tree for a European style option. This function calculates the numbers V_{jk}, which are the prices at time t_k, from the numbers $V_{j,k+1}$, which are the prices at time $t_{k+1} = t_k + \Delta t$. This function does a `for` loop over j with k fixed. It takes arguments `V_n` (the "next" V array, which represents the numbers $V_{j,k+1}$), and `q_u` (representing the risk neutral probability of $S \to uS$), and the one-step discount factor $D = e^{-r\Delta t}$. The first command in your function can be `M = length(V_n)`. The `R` function `length()` returns the length of its argument. If `V_n` is a array of length M, then `length(V_n)` will return the number M. The output is an array `V` of length $M-1$ that holds the numbers V_{jk}. These are calculated using the formula

$$V_j = D\left(q_u V_{\text{next},j+1} + q_d V_{\text{next},j}\right) \ , \quad j = 1, \ldots, M-1 \ . \tag{2}$$

Your function should create the array `V` before applying the formula (2), for example, using `V = 1:(M-1)`. (Parens are important here. If you type `V - 1:M-1`, you will get `0, 1, ..., M-1`, which is `1, 2, ..., M` with 1 subtracted from each number. If you type `for ( j in 1:M-1 )`, a similar bad thing will happen. You must type `for ( j in 1:(M-1) )`.) Test your function by writing a script that calls it once with V_{next} having length 5 (say, but not hardwired) and all zeros except a single 1 somewhere in the middle. Invent a $q_u = .4$ (say, not hardwired) and print the numbers V_j to check that they are correct (all zero except one equal to q_u and another equal to $q_d = 1 - q_u$).

Step 4, build function that evaluates $V(S_0, T)$ for a European put using a full CRR binomial tree. This function should take as argument S_0, K, T, r, σ, and n (the number of time steps). First it should compute Δt, and u and d and q_u using formulas from class. Then it should evaluate the final time stock prices

$S_{n,j} = S_0 u^{j-1} d^{n-j+1}$ in order to evaluate the numbers $V_{n,j}$, which is the cash flow generated at time T from $S = S_{n,j}$. Use the function that evaluates the intrinsic value (from step 1). Then (this is the part that may take computer time) it should call the function from step 3 n times to evaluate the CRR option value today for price S_0. This should be the return value for the function. Write a script to test this function by seeing that the return value converges to the Black Scholes value as $n \to \infty$. The script should apply the formula with fixed parameters except that it should use an increasing sequence of n values (you choose the sequence). For each evaluation it should print a nicely formatted line with the CRR and Black Scholes values, their difference, and n.

Step 5, do it for an American style put. Modify your time step function from step 3 to take the max with the intrinsic value (a function you wrote in step 1). This should change only two lines of code. One is the one that evaluates the max. The other is the one `source("...")` that tells the time step function about the intrinsic value function. Make a plot of the intrinsic value and the American option value together. This requires you to modify the function from step 2, replacing a call to the Black Scholes function with a call to the American put pricer. You should see that the curves are the same below the early exercise point and that the American option value is higher when S_0 is larger than the early exercise point. If you take a large n (for accurate value) and a lot of plot points, you should be able to observe the smooth pasting condition, that the derivative of the American style option price is continuous at the early exercise value.

Conclusion: This sequence of steps takes you from a beginner level coder to an intermediate level coder. You have to assemble pieces, test them separately, and put them together to form a larger piece of software. When the pieces work separately, you should be pleased to see that they work together.

Black Scholes formula for the European option on an underlier that does not pay dividends.

$$
\begin{aligned}
s &= \text{ the spot price today} \\
K &= \text{ the strike price} \\
T &= \text{ time remaining until the option expires} \\
r &= \text{ the risk free rate} \\
\sigma &= \text{ the volatility}
\end{aligned}
$$

$$
\begin{aligned}
V(s,K,T,r,\sigma) &= Ke^{-rT}N(-d_2) - sN(-d_1) \\
d_1 &= \frac{\log(s/K) + \left(r + \frac{\sigma^2}{2}\right)}{\sigma\sqrt{T}} \\
d_2 &= \frac{\log(s/K) + \left(r - \frac{\sigma^2}{2}\right)}{\sigma\sqrt{T}} \;=\; d_1 - \sigma\sqrt{T}\;.
\end{aligned}
$$