Simulation Methods for Option Pricing
46-932
Homework #5

Due: Thursday, February 22, 2018, 5:30pm

1. Practice on stratification.
   Suppose there are two independent assets $S_i,\ i = 1, 2$, having price processes under the risk neutral measure given by
   $$dS_i(t) = rS_i(t)dt + \sigma_i S_i(t)dW_i(t),\ i = 1, 2,$$
   where $W_1(t)$ and $W_2(t)$ are independent Wiener processes. We want to price a call option on the average asset price, $\frac{S_1+S_2}{2}$ using Monte Carlo simulation. For simplicity, assume $S_i(0) = 100$, $\sigma_i = .2$, $i = 1, 2$, $T = 1$, $r = .05$, $K = 100$. Thus, the payoff function is $\max(\frac{S_1+S_2}{2} - 100, 0)$. Assume a sample size of $n = 1,000$.

   (a) Price this asset using standard Monte Carlo, drawing $n = 1000$ pairs of independent uniform random variables, convert them into standard normal variables (using the norminv function in Matlab), use each normal to determine the final price of each underlying, then the discounted payoff of the call on the average price. Report your estimate and standard error.

   (b) Consider bivariate stratification as follows. Divide the unit rectangle into 100 identically sized squares, each .1 on a side. In each of the 100 squares, take generate a random sample of points within the square. Convert the X and Y coordinates into (conditionally) normally distributed random variables using the probability integral transform (norminv function in Matlab). Use those two normal variables to create final stock prices for the two assets, then compute the discounted call payoff. Combine the 100 estimates and associated standard errors to obtain an overall estimate and its standard error.

   (c) Now reconsider this problem using the idea of conditioning on a projection. Specifically, divide the unit interval into 250 equal sized bins. For each bin, draw a uniformly distributed random sample of size 4 from the bin. Transform each uniform variable into a (conditionally) normally distributed variable using the probability integral transform (norminv function in Matlab). For each conditionally normal random variable so generated, call it Z, generate a pair of random normal variables (Z_1, Z_2) for which $\frac{1}{\sqrt{2}}(Z_1 + Z_2) = Z$ using the idea of conditioning on a project described in page 223-226 of Glasserman's text with the $\frac{1}{\sqrt{2}}(1, 1)$ projection. Carry out this one-dimensional stratification with a sample of size 1000 of uniforms and report the estimate and its standard error. Compare your results for the three parts of this problem.

2. Practice on the Brownian bridge method.

   (a) Refer to Table 1 on page 64 of the paper by Beaglehole, Dybvig and Zhou, "Going to extremes: Correcting simulation bias in exotic option valuation." Attempt to replicate the standard simulation estimate and standard for time = 0.25, number of periods = 30 and number of draws = 1,000. Also replicate the corresponding entries for the Brownian bridge simulation. Note, your answer **should be different** from the values in the table.

   (b) Repeat the previous problem, except change the strike price. Rather than assuming that the strike is S_0, the value at the start of the period, assume instead that it is S_T, the value at the end of the period. Compare with the results in Table 1.

   (c) Now consider Table 2 on page 66. Replicate the two entries (estimate and standard error) in the table for time = .25, periods = 30, number of draws = 100,000 for both the standard simulation and for the Brownian bridge simulation.

3. Two-Asset Down-and-Out Call Option Pricing (Discrete and Continuous)
   The material on pages 104-105 in Glasserman's book presents exact path generation for multiple

dimension Geometric Brownian motion. Basically, for a d-dimension GBM, one builds the d individual paths in the usual way, except that the standard normal random variables one would use for the increments of the GBM, which would ordinarily be independent, are correlated. Thus for a given correlation matrix, Σ, one would generate a d-dimensional multivariate normal vector Z with mean 0 and covariance matrix Σ and multiply by the square root of the time increment. The Glasserman pages mention a set of applications, and the paper by Broadie, Glasserman, and Kou ("A continuity correction for discrete barrier options," *Mathematical Finance*, **7(4)**, 1997, 325-348.) provides some examples. The authors present their own way of dealing with the continuity correction. This problem asks you **not** to implement their methods but rather to implement the Brownian bridge methodology. Specifically consider one case in Table 2.5, p335: Two-Asset Down-and-Out Call Option. The two assets follow a multivariate GBM with $S_1(0) = S_2(0) = K = 100$, $r = .10, \sigma_1 = \sigma_2 = .30, \rho = 0.5, T = .2$. The option payoff is $\max(0, S_1(T) - K)$ at time T provided the option has not been knocked out by S_2 reaching or falling below $H = 95$.

a) Consider a discrete pricing problem with $N = 50$ time steps within $[0, T]$ and use ordinary Monte Carlo simulation to price the option. The paper indicates that the true price is 3.645 for the discrete time option.

b) Approximate the continuous barrier option by repeating part a) except also implement the Brownian bridge methodology. Specifically, for prices $S_2(i\Delta)$ and $S_2((i+1)\Delta)$, assume that the S_2 process behaves as a Brownian bridge at intermediate times. Generate an observation from the distribution of the **minimum** of a Brownian bridge over this time period. Use the minima so obtained over the subintervals to determine if S_2 reached the knock-out barrier. The paper claims that the continuous time price for this option is 3.158.

Note that both the relevant Glasserman pages and the paper are in the Files area of Canvas. Note that there are no standard errors in the paper, but you must accompany your estimates with standard errors. Finally, although not required, once you have a functioning program, you can easily vary ρ to see the difference ρ makes in the price, and the differences between the continuous and discrete options. The special cases of $\rho = 0$ (independent underlyings) and $\rho = +1$ (identical underlyings) might help to debug your program.

4. Practice on credit derivatives and copulas
In this problem we give a very brief and highly simplified introduction to one type of credit derivative. This material will be followed up in the credit derivatives course next fall. Some MSCF students will also encounter credit derivatives in the course of their internships this summer.

Consider a \$1 zero coupon bond and let r be the risk-free interest rate. If the bond matures at time T, then if it is default-free, its price at time 0 is e^{-rT}. If the bond is defaultable, then the owner must worry about the probability of default during $[0, T]$, π, and, in the event of default, the recovery amount, R. In the defaultable case, the expected payout at time T is given by $\pi R + (1 - \pi)$. Thus the price at time 0 is $e^{-rT}(\pi R + (1 - \pi))$. We define the *credit spread*, s, to be the annualized compensation required to assume the *credit risk* associated with the defaultable bond. In particular, we discount the price of the default-free bond, e^{-rT} by s and set it to the price of the defaultable bond, i.e. $e^{-sT}e^{-rT} = e^{-rT}(\pi R + 1 - \pi)$, or $e^{-sT} = (\pi R + 1 - \pi)$. Let us consider a simple model for default. Assume there is a constant default rate, λ, hence the probability of default during $[0, T]$ is given by $\pi = 1 - e^{-\lambda T}$ The equation relating s, T, λ, R is given by

$$e^{-sT} = (1 - (1 - e^{-\lambda T})(1 - R)).$$

Some algebra leads to

$$e^{-\lambda T} = \frac{e^{-sT} - R}{1 - R},$$

(assuming $e^{-sT} > R$), which leads to

$$\lambda = -\frac{1}{T}\log(\frac{e^{-sT} - R}{1 - R}).$$

We assume that default is rare, hence both s and λ will be small, so we can approximate the above using a Taylor expansion which yields $\pi = \lambda T$ provided π is small. The earlier equation thus becomes $1 - sT = \lambda TR + 1 - \lambda T$, or $s = \lambda(1 - R)$. Given any two values, the third can be calculated by this formula, recognizing that it is approximate. Despite its approximate nature, the formula is widely used.

One could buy a derivative security to protect against the default of any single defaultable bond, but suppose that we have a portfolio of such bonds issued by different firms, and we are interested in buying a credit derivative to protect against default of some bond in the portfolio. Assume for simplicity that each bond has the *same* values of λ and R. Consider the instrument, the ith to default (i2D), $1 \leq i \leq N$ where N is the number of different bonds. The first to default, FtD, pays off the loss (presumably $1 - R$) at time T if any one of the N bonds defaults. The second to default, 2tD, pays off $(1 - R)$ if two or more of the bonds defaults by time T. Similarly, the ith to default, itD, pays off if i or more default over the period (and then pays $1 - R$). Finally, the NtD pays off (1-R) if all N of the bonds default. The point of this problem is to prices these different credit derivatives under various dependency models.

Let $N = 5$, $T = 5$ years, $r = .04$ and assume the individual spreads and recovery rates are the same for each and are given by $s = .01$ and $R = .35$. Using these values, one can compute λ using the above equation. To determine if any particular bond defaults during [0,T], generate an exponential random variable with parameter λ to represent the default time of that bond. If that random variable exceeds T, then the bond did not default. To assess the portfolio of bonds and the credit derivatives one must generate 5 exponential(λ) random variable and note how many of the 5 are less then T, hence defaulted.

Using a Gaussian copula with correlation matrix Σ for which $\sigma_{ii} = 1$ and $\sigma_{ij} = \rho, j \neq i$, $1 \leq i, j \leq 5$ to generate the 5 different default times, each of which has an exponential(λ) marginal distribution, value each of the credit derivatives, FtD, 2tD,$\cdots$, 5tD for $\rho = 0, .2, .4, .6, .8, 1.0$. Use a sample size of 100,000 if possible. Note, the cases $\rho = 0$ (independence) and $\rho = 1$ (linearly related, hence equal) can be computed analytically. Although analytic calculations are not required, this could help in debugging.