Assignment #1 (Due February 6)

1. (**Computing Problem:** Stock options) Consider a stock whose price fluctuations over an *N* day period can be well approximated by a *lognormal random walk model*: if the initial stock price is S_0, then the price at the end of day *n* is $S_n = S_0 e^{X_1+\cdots+X_n}$ for $1 \le n \le N$, where $X_1, X_2, \ldots$ are i.i.d. normal random variables with mean μ and variance σ^2. Suppose that, at the beginning of day 1, you are given an option to purchase one unit of this stock at a fixed price *K* (the *strike price*) at the end of any of the *N* days. If you exercise this option when the price is *S* (> *K*), then you can immediately sell your stock for a *gain* of $S-K$. One strategy for deciding when to exercise the option is the following *expected gain* (EG) policy: at the end of day *m* ($1 \le m < N$), if you have not yet exercised the option, then exercise the option if $S_m > K$ and if, for $i = 1,2,\ldots,N-m$, the gain from this action is better than the expected gain $\mathrm{EG}(S_m, i)$ from letting exactly *i* days go by and then either exercising (if $S_{m+i} > K$) or giving up on ever exercising (if $S_{m+i} \le K$). That is, exercise the option on day *m* if $S_m > K$ and $S_m - K > \mathrm{EG}(S_m, i)$ for $i = 1,2,\ldots,N-m$, where $\mathrm{EG}(s,i) = E[(se^{X_1+\cdots+X_i} - K)^+]$ and $(x)^+ = \max(x,0)$. If, at the end of day *N*, you have not exercised the option on any previous day, your gain is $(S_N - K)^+$. It can be shown that $\mathrm{EG}(s,i) = se^{i\alpha}\Phi\left(\sigma\sqrt{i} + b_i\right) - K\Phi\left(b_i\right)$, where $\alpha = \mu + 0.5\sigma^2$, $b_i = (i\sigma^2)^{-1/2}[i\mu - \log(K/s)]$, and Φ is the cumulative distribution function for a standard normal random variable: $\Phi(x) = \int_{-\infty}^{x} \frac{1}{\sqrt{2\pi}} e^{-x^2/2}\,dx$.

   a) Write a simulation program for estimating the expected gain under the EG policy when *N* = 20 days, the initial price is $S_0 = \$100$, the strike price is $K = \$100$, and the random walk parameters are $\mu = -0.05$ and $\sigma = 0.3$. Based on a trial run of 1000 replications, determine the number *n* of replications required to estimate the expected gain to within ±5% with 95% probability. Using the number *n* that you have obtained, compute a final point estimate and 95% confidence interval for the expected gain based on (at least) *n* replications. When programming the simulation in Python, you can compute the function Φ using the `norm.cdf()` function; start your code with

   ```
   from scipy.stats import norm
   ```

   To generate normal random variates, you may use the Box-Muller method (Law, p. 453): generate U_1 and U_2 as i.i.d. uniform(0,1) random variates, and then set $X_1 = \left(-2\ln U_1\right)^{1/2}\cos 2\pi U_2$ and $X_2 = \left(-2\ln U_1\right)^{1/2}\sin 2\pi U_2$. It can be shown that X_1 and X_2 are i.i.d. *N*(0,1) random variables. (One convenient way to code this is to fill up an array with an even number of normal variates, and use them as needed; when the array empties out, fill it up again.)

   b) Same as above, but for the *take the money and run* (TMR) policy: at the end of day *m* ($1 \le m \le N$), exercise the option if the option has not been exercised on a previous day and $S_m > K$.

   c) Using 20,000 repetitions, compute a 95% confidence interval for the expected gain under the EG policy. Do the same for the TMR policy, making sure to use a completely different set of random numbers. Explain how to combine these two confidence intervals into a 95% confidence interval for difference between the expected gain under the EG policy and the expected gain under the TMR policy. Compute this confidence interval. [Hint: consider the distribution of the difference $X-Y$ of two independent normal random variables *X* and *Y*.]

d) Modify your program to compute, for each repetition of an *N*-day option period, the difference between the gain under the EG policy and the gain under the TMR policy. Run 20,000 simulation repetitions to generate 20,000 such differences, and compute a 95% confidence interval for the expected difference in the gains (which equals the difference in the expected gains). Is this interval wider or narrower than the interval computed in part (c)? Give an intuitive explanation for your result.

2. (Monte Carlo integration) As mentioned in class, we can use simulation to solve deterministic numerical computation problems. Suppose in the following problems that we have available sequences $U_1, U_2, \ldots$ and $V_1, V_2, \ldots$ of i.i.d. uniform(0,1) random numbers.

   a) Suppose that we wish to evaluate the integral $I = \int_0^1 h(x)\,dx$, where the nonnegative function *h* is so complicated that analytical or numerical evaluation of the integral is impossible. Explain why we can estimate *I* by $\hat{I}_n = (1/n)\sum_{i=1}^{n} Z_i$, where $Z_i = h(U_i)$ for $i \ge 1$ and *n* is sufficiently large. How can we choose *n*?

   b) Now describe an estimation procedure as above, but for the integral $I = \int_a^b h(x)\,dx$, where *a* and *b* are specified numbers. [Hint: consider the transformation $y = (x-a)/(b-a)$.]

   c) Same as part (b), but for the integral $I = \int_0^\infty h(x)\,dx$. [Hint: consider the transformation $y = 1/(x+1)$.]

   d) Same as part (b), but for the integral $I = \int_0^1 \int_0^1 e^{(x+y)^2}\,dxdy$ and using both sequences $U_1, U_2, \ldots$ and $V_1, V_2, \ldots$.

   e) Same as part (b), but for the integral $I = \int_0^\infty \int_0^x e^{-(x+y)^2}\,dydx$. [Hint: using the function $g(x,y)$ which equals 1 if $y \le x$ and 0 otherwise, rewrite *I* so that the upper limit of the inner integral doesn't depend on *x*.]

3. (A probability refresher) Let *X* and *Y* be independent uniform(0,1) random variables. Compute and graph the probability density function (pdf) of *Z* = *X* + *Y*. [Hint: Use the law of total probability to write

$$P(Z \le z) = P(X+Y \le z) = \int_0^1 P(X+Y \le z \mid Y=y) f_Y(y)\,dy = \int_0^1 P(X \le z-y) f_Y(y)\,dy$$

and consider the cases $z \in [0,1]$ and $z \in [1,2]$ separately.] Note: If you want to "cheat" and get a rough idea of what the solution should look like, use simulation! E.g., the following R code will do the trick:

```
x = runif(1000000)
y = runif(1000000)
z = x + y
hist(z, probability=TRUE)
```

4. (An alternative Monte Carlo integration method) **Note: do this problem individually**. Suppose that we want to evaluate the two-dimensional integral $I = \int_0^1 \int_0^1 e^{-(x-0.5)^2-(y-0.5)^2} dxdy$ using Monte Carlo integration. A standard way to do this would be to generate N uniform number pairs $\mathbf{Z}_1 = (X_1, Y_1), \ldots, \mathbf{Z}_N = (X_N, Y_N)$ and estimate I by $\hat{I}_N = (1/N) \sum_{i=1}^N h(\mathbf{Z}_i)$. Confidence intervals are of the form $\hat{I}_N \pm z s_N / \sqrt{n}$, where z is an appropriate quantile of the normal distribution and $s_N^2 = \frac{1}{N-1} \sum_{i=1}^N \left(h(\mathbf{Z}_i) - \hat{I}_N \right)^2$. Here all of the X's and Y's are uniformly distributed on [0,1] and mutually independent, and $h(\mathbf{z}) = e^{-(x-0.5)^2-(y-0.5)^2}$ for $\mathbf{z} = (x, y)$. Now consider the following alternative algorithm. Choose K such that K divides N and set $n = N/K$. Then execute the following unbiased estimation procedure:

   A. Generate a sequence $Q_1, Q_2, \ldots, Q_n$ as follows
      1. Generate K uniform pairs $\mathbf{Z}_1 = (X_1, Y_1), \ldots, \mathbf{Z}_K = (X_K, Y_K)$.
      2. Generate two independent random permutations π_1 and π_2 of $(1, 2, \ldots, K)$. E.g., one possible realization of π_1 is given by $\pi_1(1) = K,\ \pi_1(2) = K-1,\ \ldots\ ,\ \pi_1(K) = 1$.
      3. Define $\mathbf{W}_k = (U_k, V_k)$ for $k = 1, 2, \ldots, K$ by
         $U_k = \left(\pi_1(k) - X_k \right) / K$ and $V_k = \left(\pi_2(k) - Y_k \right) / K$.
      4. Set $Q = \left(1/K \right) \sum_{k=1}^K h(\mathbf{W}_k)$.
      5. Repeat Steps 1-4 above n times to create i.i.d. random variables $Q_1, Q_2, \ldots, Q_n$.

   B. Estimate I by $\hat{I} = (1/n) \sum_{i=1}^n Q_i = (1/N) \sum_{i=1}^n \sum_{k=1}^K h(\mathbf{W}_{i,k})$.

   C. Compute a $100(1-\delta)\%$ confidence interval as $\hat{I} \pm z s_n / \sqrt{n}$, where z is the usual $(1-\delta/2)$-quantile of the standard normal distribution and s_n is the sample standard deviation of $Q_1, Q_2, \ldots, Q_n$.

a) Using the R, Matlab, or Octave software packages, estimate the standard error (i.e., sample standard deviation divided by square root of sample size) for the standard and alternative estimation methods described above, with $K = 5$, and $N = 150$. Which method is superior? [The file hw1code.txt on the course web site contains R code and Matlab/Octave code for doing the calculations, along with download instructions for the open source R and Octave packages. One point of this exercise is to introduce you to these useful packages for Monte Carlo simulation. If you already know them, you are encouraged to try writing the code yourself.]

b) Draw a sketch in which the unit square is divided into $K \times K$ square subregions (for $K = 5$). On this sketch, indicate a typical realization of K two-dimensional points $\mathbf{W}_1, \mathbf{W}_2, \ldots, \mathbf{W}_K$. Using your drawing, give an intuitive explanation of why one of the estimation methods is better than the other. (Compare the locations of the K points with the locations of K points chosen randomly and uniformly as in the standard algorithm.)

c) For the alternative method, we unbiasedly estimate the variance of $\hat{I}$ as s_n^2 / n when forming a confidence interval, where $s_n^2 = \frac{1}{n-1}\sum_{i=1}^{n}\left(Q_i - \hat{I}\right)^2$. Since $\hat{I} = (1/N)\sum_{i=1}^{n}\sum_{k=1}^{K} h(\mathbf{W}_{i,k})$, could we also have estimated the variance of $\hat{I}$ as s^2 / N, where $s^2 = \frac{1}{N-1}\sum_{i=1}^{n}\sum_{k=1}^{K}\left(h(\mathbf{W}_{i,k}) - \hat{I}\right)^2$?