ECG790C - Problem Set 3 - Spring 2009

1. Heston's stochastic volatility option pricing model uses a two factor stochastic process

$$dS = (r - \delta)Sdt + \sqrt{v}SdW_1$$

and

$$dv = \lambda(m - v)dt + \sigma\rho\sqrt{v}dW_1 + \sigma\sqrt{1-\rho^2}\sqrt{v}dW_2$$

Given S_0 and v_0, use Monte Carlo with 1000 paths to find an estimate of a put option with strike price K expiring in T periods. To do so, first convert the model to one with the first factor being $\ln(S)$ rather than S (the simulation is more accurate this way).

The drift in v is affine in v and hence the expected time path of v is $m + e^{-\lambda t}(v_0 - m)$. Define $\bar{v}$ to be the average expected value of v over the period, i.e.

$$\bar{v} = E\left[\frac{1}{T}\int_0^T v_t dt \Big| v_0\right] = \frac{1}{T}\int_0^T E\left[v_t|\, v_0\right]dt$$

It can be shown that

$$\bar{v} = m + \frac{(1 - \exp(-\lambda T))}{\lambda T}(v_0 - m)$$

Use this value of the volatility with the Black-Scholes option pricing formula. as a control variate to (hopefully) improve the estimate for the Heston model. Perform a numerical experiment to estimate the improvement with 5000 replications. Provide estimates of the variance and standard deviations of the simple and control variate MC approximations.

Use the parameter values $r = 0.05$, $\delta = 0$, $\lambda = 1$, $m = 0.15$, $\sigma = 0.4$, $\rho = -0.5$, $K = 1$, and $T = 1$. Assume that $S_0 = K$ and $v_0 = 0.1$.

Note that `demfin03.m` uses the PDE method to compute a value for this option (with different parameters).

2. A common problem in computation is finding the inverse of a cumulative distribution function (CDF). A CDF is a function, F, that is nondecreasing over some domain $[a,b]$ and for which $F(a)=0$ and $F(b)=1$. Write a function that uses Newton's method to solve inverse CDF problems. Your function should implement Newton's method internally rather than calling a generic version of Newton's method. The function should take the following form:

   ```
   x=icdf(p,F,x0,varargin)
   ```

   where `p` is a probability value (a real number on [0,1]), `F` is the name of a MATLAB function file, and `x0` is a starting value for the Newton iterations.

   The function file should have the form:

   ```
   [F,f]=cdf(x,additional parameters)
   ```

   For example, the normal CDF with mean μ and standard deviation σ would be written:

   ```
   function [F,f]=cdfnormal(x,mu,sigma)
     z=(x-mu)./sigma;
     F=cdfn(z);
     f=exp(-0.5*z.^2)./(sqrt(2*pi)*sigma);
   ```

   You can test your code with the statement:

   ```
   x-icdf(cdfnormal(x,0,1),'cdfnormal',0,0,1)
   ```

   which should return a number close to 0 for any value of x.

3. Quasi-Newton methods for solving non-linear equations ($f(x)=0$) are typically defined as iterative methods of the form

   $$x_{k+1} \leftarrow x_k - \lambda_k B_k f(x_k)$$

   where the sequence of matrices B_k satisfies the so-called "secant condition"

   $$B_{k+1}(f(x_{k+1}) - f(x_k)) = x_{k+1} - x_k$$

   You may recall that the Broyden method uses the B_{k+1} that satisfies the secant condition and is, in a certain sense, as close to B_k as possible. Suppose instead that you pick B_{k+1} so that B_{k+1} differs from B_k only in a single column. This can be written as

   $$B_{k+1} \leftarrow B_k + u e_j^\top$$

   where e_j is the jth column of an identity matrix and u is a vector to be determined. Find the u that satisfies the secant condition.

To define a fully constructive algorithm, one must have a method for picking j. Suppose j is picked to be the index of the largest absolute value of $f(x_{k+1}) - f(x_k)$, i.e., the column of B picked to change is associated with the largest change in the function value. Modify the CompEcon procedure `broyden` to use this updating rule for B. Test your function and compare it to the performance of `broyden` (be sure to test it on a multivariate function).