Derivative Securities, Courant Institute, Fall 2006
Assignment 8, due November 15

Important: Always check the class bboard on the blackboard site from home.nyu.edu (click on academics, then on Derivative Securities) before doing any work on the assignment. In particular, some questions may be deleted or added depending on how much material we cover in class.

1. Suppose $f(s,t)$ is the price of a European option that depends on a stock price process that satisfies $dS = \mu S dt + \sigma S dW$. Suppose that risk free borrowing and lending has constant known interest rate r (the risk free rate).

   (a) Find the number λ (depending on σ and r and μ) so that $h(t) = f(S(t),t)/S(t)$ is a martingale for

   $$dS = \lambda S dt + \sigma S dW \ . \qquad (1)$$

   If λ does not depend on one of the parameters, explain why not. *Hint*: Calculate $dh = adt + bdW$ and figure out how to set $a = 0$.

   (b) Since $h(S(t),t)$ is a martingale, it should satisfy $E[h(S(T),T)] = h(s_0,0)$. Verify that this is so for a final time European payout $f(S(T),T) = V(S(T))$, i.e.

   $$E_\lambda\left[\frac{V(S(T))}{S(T)}\right] = \frac{f(s_0,0)}{s_0} = \frac{e^{-rT}E_r\left[V(S(T))\right]}{s_0} \ . \qquad (2)$$

   The notation $E_\lambda[\cdot]$ means to take the expectation supposing that $S(t)$ satisfies (1). It goes without saying that we also suppose that $S(0) = s_0$. This derivation is surprisingly involved. First, write $S(T)$ in terms of $W(T)$, then in terms of Z, a standard normal random variable. We've done this before. Then use the Gaussian density formula to write the left side of (2) as an explicit integral involving V. Then do a change of z variable to put this integral into the form of the integral that defines $f(s_0,0)$ for the risk free process in the right most part of (2). The formula for λ in terms of r and σ and μ is used several times.

   (c) *Discussion*: Whenever you have a new formula or fact, it's helpful to see that you really understand it by testing it on a fact you know. The difficulty in verifying it in this case makes the formula seem useful as a simple way to understand something that is hard to understand directly. Here, we are checking the discussion in Section 25.3 of Hull on martingale measures. Note that the *market price of risk*, $\lambda - r$, does not depend on the option payout, only on the fact that f satisfies the Black Scholes equation. Also, the formula (2) is a different way to evaluate $f(s_0,0)$, which might have advantages over the standard way if we are using Monte Carlo methods.

2. We are going to add some functionality, some greeks and implied vol. Let `BS(s0, K, T, sig, r)` be the VBA function you wrote last week to implement the Black Scholes formula for a European put price.

   (a) Write VBA functions, `BSD`, `BSG`, and `BSV`, that compute the Black Scholes values for $\Delta = \partial_S f$, $\Gamma = \partial_S^2 f$, and $\Lambda = \partial_\sigma f$ (Note: the Greek letter Λ is called Vega in this context.). The notes by Kohn and Allen have some helpful hints and identities for this.

   (b) Use finite differences of values produced by `BS` itself to check that these greeks are calculated correctly. Do this in the spreadsheet itself. Hand in the VBA code and a printout of the spreadsheet that shows that the three functions are correct.

   (c) Use the `BSV` function to implement `BSIV`, which uses Newton's method to compute Black Scholes implied volatility. Here are suggestions for details of this function, but I don't consider these particularly sound, just quick and dirty. Take the initial guess to be $\sigma_0 = 20\%$. The *Newton step* is (leaving out all other arguments) $p_k = -f(\sigma_k)/\Lambda(\sigma_k)$, and Newton's method is $\sigma_{k+1} = \sigma_k + p_k$. Modify this formula so that $|p_k| \leq \sigma_k/2$. This prevents the computation from going nuts with a poor initial guess. Raise a popup error message using `ErrMsg` if you reach $k = 100$ without finding an acceptable answer. Stop if $|f(\sigma_k) - f_*| < \epsilon f_*$, where f_* is the desired price. You should be able to make ϵ very small. Verify in a few cases that `BSIV` gives the right answer. Pick some easy and hard cases. It would take more work to make this solver *bulletproof* (i.e. work in every possible case).

   (d) *Discussion*: We will use all this infrastructure in the next assignment. For now, just get the stuff working. We need Λ for Newton's method, so it's a reasonable time to code the other greeks.