Homework 7

ECON 6204 (8204) - 001

Fall 2017

100 Points

Reading Assignment: Lectures 6 - 8

Due: Tuesday, November 7 Student Name: _____

Instruction. On the due date, you should email your completed homework with a single zipped folder named "hw7_YourName. The folder should include a driver file named "main.py" for your Python program and some other items, as specified below.

Problem. There are an **European call** option and an European put option on the underlying asset S that follows a geometric Brownian motion, $dS = \mu S dt + \sigma S dW$, where μ is mean return, σ is the volatility parameter, W is the standard Wiener process. Under risk-neutral evaluation, the Brownian motion can be transformed into $dx = (r - \frac{\sigma^2}{2})dt + \sigma dW$, where $x = \ln(S)$. The Fourier transform techniques allow us to compute option prices for models with a known characteristic function of x. From Lecture 7, we have derived the following pricing integral using Fourier transform and inverse Fourier transform:

$$\begin{aligned} V(k) &\approx \text{Re}\left\{\frac{e^{-\alpha k}}{\pi}\int_0^B e^{i\omega k}\hat{\nu}(\omega)d\omega\right\} \\ &= \text{Re}\left\{\frac{e^{-\alpha k}}{\pi}\sum_{m=0}^{N} e^{i\omega_m k}\hat{\nu}(\omega_m)\triangle\omega_m\right\}, \ \triangle\omega_m \begin{cases} = \frac{h}{2}, & m = 0 \text{ or } N \\ = h, & m \neq 0 \text{ or } N \end{cases} \end{aligned}$$

where V is the present-time price of the European call or the European put, depending on the sign of the damping parameter α; i is the imaginative unit, α is the damping factor; $\nu = Ve^{-\alpha k}$ is a normalized option price; $\hat{\nu}$ is the Fourier transform of ν due to $\hat{\nu} = \int_{-\infty}^{\infty} e^{-i\omega k}\nu(\omega)d\omega$; $B > 0$ is chosen to be an appropriate upper bound; the frequency domain is discretized by the equidistance $h = B/N$ so that $[0, B]$ partitioned into N subintervals, and the summation from $m = 0$ to N is based on the well-known Trapezoid rule. To apply the fast Fourier transform (FFT) algorithm, we dropped the very last term in the summation so that

$$\begin{aligned} V(k_n) &\approx \text{Re}\left\{\frac{e^{-\alpha k_n}}{\pi}\sum_{m=0}^{N-1} e^{i\omega_m k_n}\hat{\nu}(\omega_m)\triangle\omega_m\right\}, \ \triangle\omega_m \begin{cases} = \frac{h}{2}, & m = 0 \\ = h, & m \neq 0 \end{cases} \\ &= \text{Re}\left\{\frac{e^{-\alpha k_n}}{\pi}\sum_{m=0}^{N-1} e^{2\pi i\left(\frac{mn}{N}\right)} e^{i\omega_m k_0}\hat{\nu}(\omega_m)\triangle\omega_m\right\} \end{aligned}$$

where $m, n \in \{0, 1, \ldots, N\}$ and $e^{i\omega_m k_n}$ is written as $e^{i(mh)(k_0 + n\triangle k)} = e^{2\pi i\left(\frac{mn}{N}\right)} e^{i\omega_m k_0}$ by setting $h\triangle k = \frac{2\pi}{N}$.

Note that $\hat{\nu}(\omega_m)$ can be derived as

$$\hat{\nu}(\omega_m) = \frac{e^{-r(T-t_0)} \cdot \hat{q}(\omega_m + (\alpha+1)i)}{(\alpha - i\omega_m)(\alpha - i\omega_m + 1)}$$

where $t_0 = 0$ is the present time and $\hat{q}$ is equal to $\overline{\varphi}_x$, which is a complex conjugate of the known characteristic function of $\varphi_x(\omega_m) = e^{i[x_0+(r-\frac{1}{2}\sigma^2)t]\omega_m - \frac{1}{2}\sigma^2\omega_m^2 t}$. Therefore, we can write:

$$\hat{q}(\omega_m + (\alpha+1)i) = \overline{\varphi}_x(\omega_m + (\alpha+1)i)$$

$$\overline{\varphi}_x(\omega_m^{'}) == e^{-i(x_0+(r-\frac{\sigma^2}{2})t)\omega^{'} - \frac{\sigma^2 t}{2}\omega_m^{'2}}, \quad \omega_m^{'} = \omega_m + (\alpha+1)i$$

Given these parameters ($S_0 = 100$, $K = 80$, $r = 0.05$, $T = 1$, $t_0 = 0$, $\sigma = 0.50$), you are asked to compute V for both European call and put with Python by evaluating the two pricing integrals based on the FFT algorithm. For this algorithm, we set B=50, $N = 2^{10}$, $h = B/(N-1)$, $\triangle k = \frac{2\pi}{hN}$, $\omega_m = mh$, $k_n = k_{\min} + n\triangle k$ with $k_{\min} = \log(K_{min} = 20)$, $\alpha = [2.5, 5, 10]$ for the European call, and $\alpha = [-2.5, -5, -10]$ for the European put.

Note that $k = \log(K = 80)$ may not be exactly in $\{k_0, k_1, \ldots, k_N\}$. So, you can apply crude approximation, or more precisely, interpolation (scipy.interpolate) to find the option value corresponding to $K = 80$.

Note that when your Python program is executed, it should be able to generate console outputs that are well tabulated and plot the relationship between V_n and $K_n = e^{k_n}$ in each case.. It is also required that your Python program should include remarks specifying the homework's purpose, algorithms, and author (i.e., your name) on top of your main.py function. To make your code reader-friendly, you should add remarks when necessary.