Analytic Formulas

Tyler J. Brough

November 26, 2017

Black Scholes and Geometric Asian Option Formulas

Black-Scholes

We can code the Black-Scholes Call pricing formula and the Black-Scholes Delta formula as follows:

```
from scipy.stats import norm
import numpy as np

def BlackScholesCall(S, K, r, v, q, tau):
    d1 = (np.log(S/K) + (r - q + 0.5 * v * v) * tau) / (v * np.sqrt(tau))
    d2 = d1 - v * np.sqrt(tau)
    callPrc = (S * np.exp(-q * tau) * norm.cdf(d1)) - (K * np.exp(-r * tau) * norm.cdf(d2))
    return callPrc

def BlackScholesDelta(S, K, r, v, q, tau):
    d1 = (np.log(S/K) + (r - q + 0.5 * v * v) * tau) / (v * np.sqrt(tau))
    delta = np.exp(-q * tau) * norm.cdf(d1)
    return delta
```

We can use these formulas as follows (for the familiar case):

```
S = 41.0
K = 40.0
r = 0.08
v = 0.30
q = 0.0
T = 1.0

callPrc = BlackScholesCall(S, K, r, v, q, T)
callDelta = BlackScholesDelta(S, K, r, v, q, T)
print("The Call Price is: {0:0.3f}".format(callPrc))
print("The Call Delta is: {0:0.3f}".format(callDelta))
```

```
The Call Price is: 6.961
The Call Delta is: 0.691
```

Geometric Asian Options

We can also use the Black-Scholes formula to price a Geometric Asian option. See the appendix of Chapter 14 in the McDonald text for details.

```
def GeometricAsianCall(S, K, r, v, q, tau, N):
    dt = tau / N
    nu = r - q - 0.5 * v * v
    a = N * (N + 1) * (2.0 * N * 1.0) / 6.0
    V = np.exp(-r * tau) * S * np.exp(((N + 1.0) * nu / 2.0 + v * v * a / (2.0 * N * N)) * dt)
    vavg = v * np.sqrt(a) / (N**(1.5))
    return BlackScholesCall(V, K, r, vavg, 0, tau)
```

We can use this formula as follows:

```
geoPrc = GeometricAsianCall(100.0, 100.0, 0.06, 0.2, 0.03, 1.0, 10)
print("The Price of the Geometric Asian Call is: {0:0.3f}".format(geoPrc))
```

```
The Price of the Geometric Asian Call is: 5.210
```