Overview

Black-Scholes model is widely used in modeling asset price dynamics of an asset. It provides an important benchmark to evaluate the performance of other models. However, the Black Scholes Model assumes a constant risk-free interest rate, no default risk and a smooth frictionless trade which are generally quite unrealistic partly to its inability to generate the volatility and the skewness in the distribution of the return. Therefore, a variety of models have suggested to capture such properties of the return. In particular, Heston has proposed a stochastic volatility model with a closed-form solution for the price of a European call option when the underlying assets are correlated with a latent volatility stochastic process. Below we review the Heston model. Consider at time t the spot asset S(t) which obeys a diffusion process:

In this work we model local volatility using the Heston Model Dynamics and Fourier pricing technique to get an estimation of the call value of an option. The Heston model assumes that the asset follows the following dynamics:

$$dS_t = \mu S dt + \sqrt{v_t} S dW_t^1$$

where S_t is the asset value, W_t^1 is a standard Brownian motion, and the volatility term, v_t, follows a CIR, or square-root process:

$$d\sqrt{v_t} = -\beta\sqrt{v_t}dt + \sigma dW_t^2.$$

Next we consider the local volatility by using the Constant Elasticity of Variance (CEV) model .
This is given by:

$$dS_t = \mu S_t dt + \sigma S_t^{\gamma} dW_t$$

where γ and σ are non-negative, and all other variables are defined as in the usual Black-Scholes model. If γ= 1, we are back to the standard Black-Scholes model. For γ< 1, stock volatility increases as prices decrease – an effect called leverage, which is often observed in practice.

The CEV model describes a process which evolves according to the following stochastic differential equation:
$ds_t = \mu S_t dt + \sigma S_t^{\gamma} dW_t$ (7)
where S is the spot price, t is time, and μ is a parameter characterizing the drift, σ and ϒ are other parameters, and W is a Brownian motion.
The notation "dX" represents a differential, i.e. an infinitesimally small change in parameter X.

The constant parameters , σ, ϒ satisfy the conditions, σ ≥0,Y≥0 .The parameter ϒ controls the relationship between volatility and price, and is the central feature of the model. When ϒ < 1 we see the so-called leverage effect, commonly observed in equity markets, where the volatility of a stock increases as its price falls. Conversely, in commodity markets, we often observe ϒ > 1 , the so-called inverse leverage effect,[3][4] whereby the volatility of the price of a commodity tends to increase as its price increases.
If we rewrite our dynamics as:

$dst = \mu St dt + \sigma St(St,t) dWt$ (8)

If we do this, then we should have it that σ (St; t) = σ $St\gamma-1$. We are assuming that our volatility process does not explicitly depend on time; only on the underlying asset price. Implementing the CEV model in python, we first, define the parameters to be used alongside defining the stock prices to be used:

#Define additional parameters

localhost:8888/notebooks/Downloads/GroupWork_Submission_2/GroupWork-Submission-2.ipynb

jupyter GroupWork-Submission-2 Last Checkpoint: 29 minutes ago (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

Run Code

```python
#Call Option specific information
K = 100
T = 1
k_log = np.log(K)

#Approximation information
t_max = 30
N = 100
a = sigma **2 /2
```

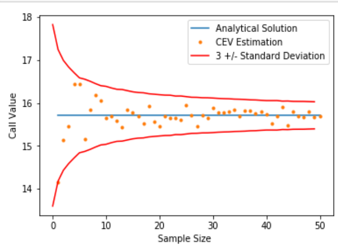
In [2]:
```python
# characteristic function Code

def b(u, kappa = kappa, rho = rho, sigma = sigma):
    return kappa - rho * sigma *1j * u

def c (u)for j in range(0,number_of_periods):
            Z = norm.rvs(size=i * 1000)
            if j ==0:
                sigma_cev = sigma * np.array([S0] * i * 1000) ** (gamma - 1)
                share_prices = np.array([S0] * i * 1000) * np.exp((r - (sigma_cev **2)/2) * (T/number_of_periods) + (sigma_cev *
            else:
                sigma_cev = sigma * share_prices ** (gamma - 1)
                share_prices = share_prices * np.exp((r - (sigma_cev **2)/2) * (T/number_of_periods) + (sigma_cev * np.sqrt(T/num
        call_estimates[i-1] = np.mean(discounted_call_payoff(share_prices, K, r, T))
        call_std_estimates[i-1]= np.std(discounted_call_payoff(share_prices, K, r, T)) / np.sqrt(i * 1000)
    return call_estimates, call_std_estimates
```

When gamma 0 = 1

In [6]:
```python
call_estimates_gamma_one, std_estimates_gamma_one = montecarlo_estimation(gamma = 1, number_of_periods = 12)
```

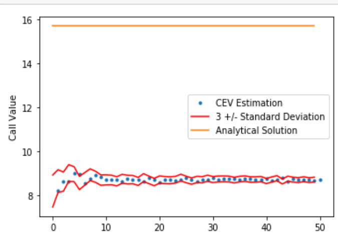
In [7]:
```python
d1 = (math.log(S0/K) + (r + sigma**2/2) * (T)) / (sigma * math.sqrt(T))
d2 = d1 - sigma*math.sqrt(T)

analytic_callprice = S0 * norm.cdf(d1) - K*math.exp(-r * (T))*norm.cdf(d2)
```

In [8]:
```python
plt.plot(list(range(1, 51)), [analytic_callprice] * 50, label = "Analytical Solution")
plt.plot(list(range(1, 51)), call_estimates_gamma_one, ".", label = "CEV Estimation")
plt.plot(analytic_callprice + 3*np.array(std_estimates_gamma_one), "r", label = "3 +/- Standard Deviation" )
plt.plot(analytic_callprice - 3*np.array(std_estimates_gamma_one), "r")
plt.xlabel("Sample Size ")
plt.ylabel("Call Value")
plt.legend()
plt.show()
```



In [9]:
```python
call_estimates, std_estimates = montecarlo_estimation(gamma = 0.75, number_of_periods = 12)
```

In [10]:
```python
plt.plot(list(range(1, 51)), call_estimates, ".", label = "CEV Estimation")
plt.plot(call_estimates + 3*np.array(std_estimates), "r", label = "3 +/- Standard Deviation")
plt.plot(call_estimates - 3*np.array(std_estimates), "r")
plt.plot([analytic_callprice] * 50, label = "Analytical Solution")
plt.xlabel("Sample Size")
plt.ylabel("Call Value")
plt.legend()
plt.show()
```



Bibliography

Albrecher, H., Mayer, P., Schoutens, W. and Tistaert, J. (2007). “The Little Heston Trap”, Wilmott (1): 83–92.

Bauer, R. (2012). Fast calibration in the Heston model. Vienna Institute of Technology.

Breeden, D. T. and Litzenberger, R. H. (1978). “Prices of State-contingent Claims Implicit in Option Prices”, Journal of Business, pp. 621–651.

Davis, M. H. (2011). “The Dupire Formula”, Finite Difference Methods Course material, Imperial College London.

Gatheral, J. (2011). The volatility surface: a practitioner's guide (Vol. 357). John Wiley & Sons.

Heston, S. L. (1993). “A Closed-form Solution for Options with Stochastic Volatility with Applications to Bond and Currency Options”, The Review of Financial Studies 6(2): 327–343.

Hsu, Y.-L., Lin, T. and Lee, C. (2008). “Constant Elasticity of Variance (cev) Option Pricing Model: Integration and Detailed Derivation”, Mathematics and Computers in Simulation 79(1): 60–71.

Rouah, F. D. (n.d.). “Heuristic Derivation of the Fokker-Planck Equation”, www. frouah. com 14: 15, (accessed on 21.09. 2015, at https://frouah.com/finance%20notes/Derivation%20of%20the%20FokkerPlanck%20equation.p df).

Rouah, F. D. (2013). The Heston Model and Its Extensions in Matlab and C. John Wiley & Sons.