Stochastic Jump Simulation Monte Carlo Simulation Using Euler Discretization

Qianfan Wu, Advisor: Prof. Gustavo Schwenkler
MS. Mathematical Finance, Boston University
Codes Available at: Github Link

This project explores the Monte Carlo simulation method of the Merton's Jump & Diffusion stochastic process, in which the security price jumps are modeled by a point process with Poisson intensity. A Euler Discretization Scheme proposed by Giesecke, Teng & Wei (2015) is applied to simulate a single MC path. The computational efficiency of this method, including MSE, bias, and variance is evaluated. We also derived the optimized Euler Bound Parameter ϵ for options with different maturities and strikes.

Suppose the price of a security S follows a Merton Process of the type:

$$dS_t = \mu S_{t-}dt + \sigma S_{t-}dB_t + S_{t-}dJ_t$$

For a standard Brownian Motion B and a point process of the type:

$$J_t = \sum_{j=1}^{N_t} \left(e^{X^j} - 1\right)$$

With i.i.d. random variables $X^j \sim N(a, b^2)$, and a Poisson process N with intensity λ. Let $Z_{K,M}$ be the estimator based on K Monte Carlo samples of a Euler Discretization with M Euler steps per unit of time. That is,

$$Z_{K,M} = \frac{1}{K}\sum_{k=1}^{K} 1_{\left\{\widehat{N}_T^{(k)} < \kappa_J \Delta^{-\epsilon}\right\}} \left(\hat{S}_T^{(k)} - X\right)_+$$

where $(\hat{S}_T^{(k)}), 1 \le k \le K$ are independent samples of an Euler discretization of S with Euler step size $\Delta = \frac{1}{M}$, $(\widehat{N}_T^{(k)}), 1 \le k \le K$ are the resulting samples of the number of jumps until maturity, and $\kappa_J, \epsilon > 0$ are constants.

From the definition of Merton model and the properties of Poison process, we could derive that conditional on N_t, the stock price S_t is lognormally distributed. More precisely:

$$S_t|_{N_t=n} \sim LN(\log(S_o + \left(\mu - \frac{\sigma^2}{2}\right)t + an, \sigma^2 t + b^2 n)$$

This expression can be used to compute proxies for the true option prices via exact simulation with a large number of Monte Carlo samples.

We would like to evaluate the accuracy and computational efficiency of the Euler discretization estimator for pricing European options. Suppose we wished to price a European call option on security S. The maturity of the option is $T \in \{0.5, 1, 3\}$ $(years)$ and the strike price is $X \in \{1500,2000,2500\}$ ($). For this problem, we would like to choose:

$$S_0(initial\ price) = 2000;$$
$$K(\#of\ MC\ Samples) = 1000;$$
$$\mu = r = 0\ (Merton\ Parameters);$$
$$\sigma = 0.17\ (Volatility);$$
$$\lambda = 2\ (Jump\ Intensity\ );$$
$$a =\ -0.05, b = 0.03\ (Lognormal\ Parameters);$$
$$M = 100\ (Euler\ Steps);$$
$$\kappa_J = 2\ (Euler\ Parameter)$$

We plotted the mean squared error, the estimation bias, and the variance of the estimator $Z_{K,M}$ as a function of ϵ $(Euler\ Bound\ Parameter)$. The true option market prices are given in the following table. We also obtained the value of ϵ that minimizes the mean squared error of the MC estimator for different

True Option Price ($):

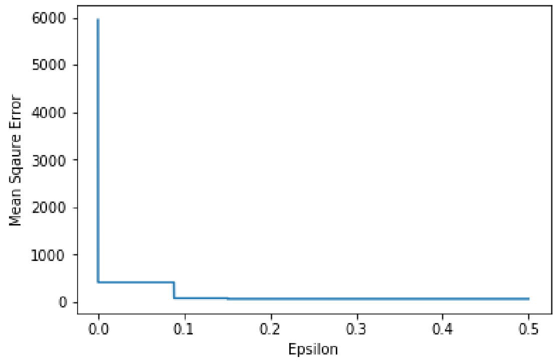
X(Strike), T(Maturity)	**T = 0.5**	**T = 1**	**T = 3**
X = 1500	409.17	341.34	191.81
X = 2000	62.53	68.67	56.29
X = 2500	1.81	6.48	14.54

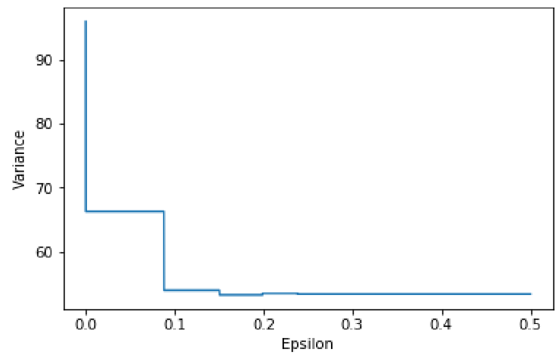
Given the above framework, there are $1000\ (K) * 100\ (M) = 100{,}000$ of Euler Discretization Monte Carlo runs, for each given ϵ. In the MC settings, the values of ϵ is sampled from the range $[0{,}0.5]$,with step size 0.0001.

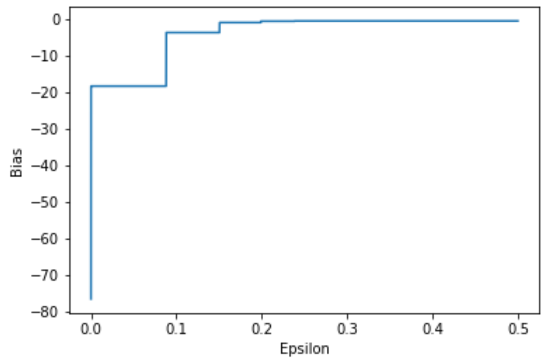
The number of MC estimators is relatively small due to computational power limit. The result should be much more stable and robust if I have more computational budget and if I could run 10000 or more estimators.

T = 0.5, X = 1500:

When Epsilon = 0.2721, it minimizes MSE

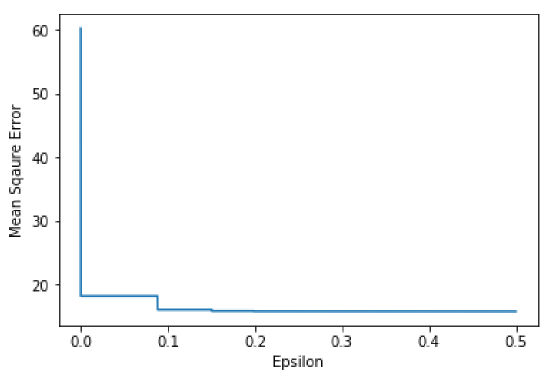


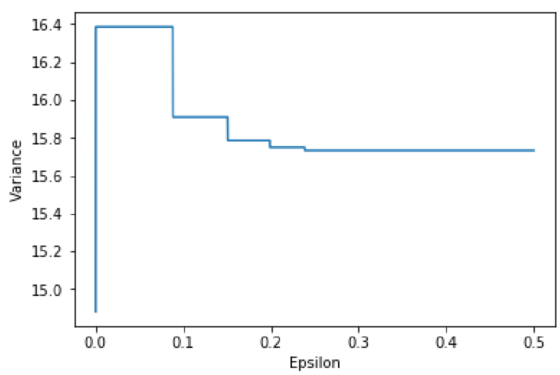


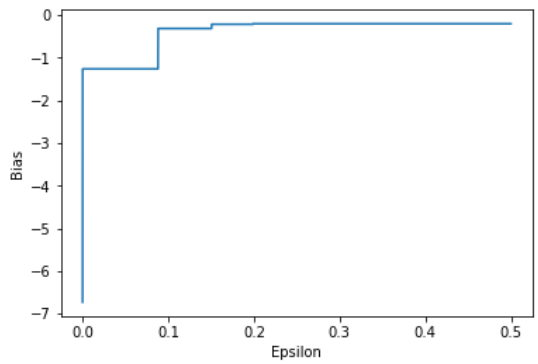


T = 0.5, X = 2000:

When epsilon = 0.2386, it minimizes MSE.

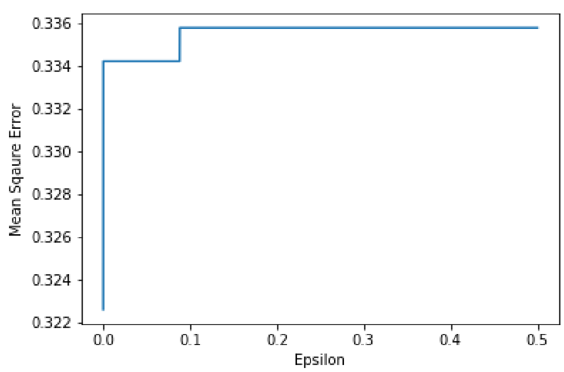


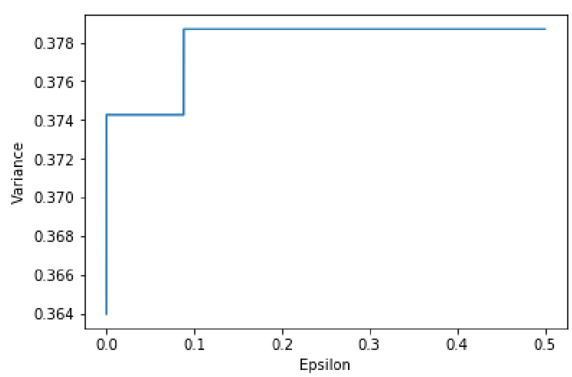


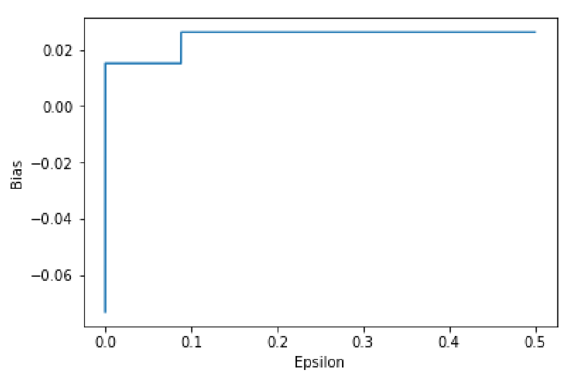


T = 0.5, X = 2500:

When epsilon = 0, it minimizes MSE.

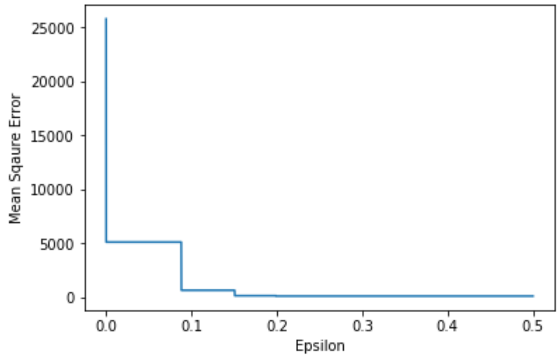


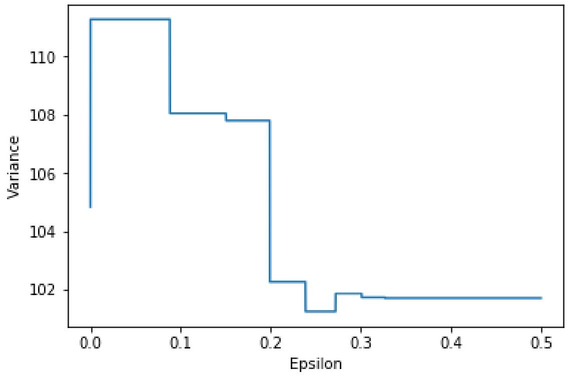




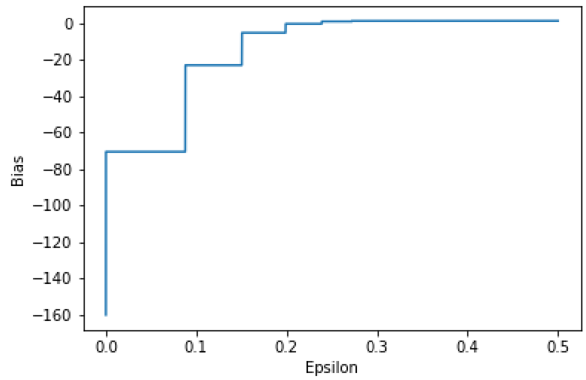
T = 1, X = 1500:

When epsilon = 0.2386, it minimizes MSE.



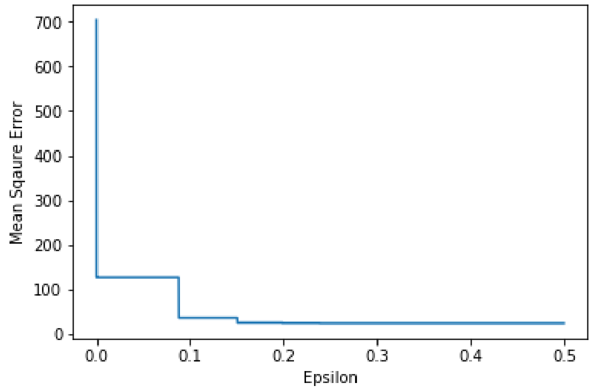


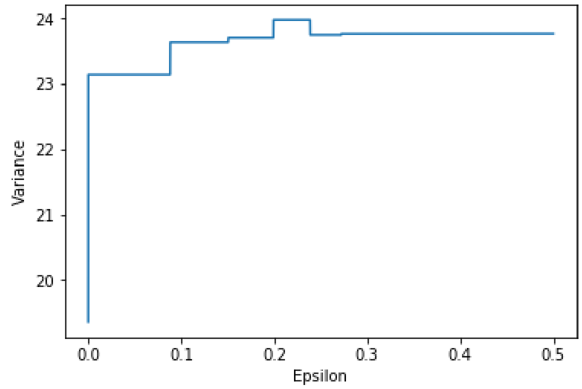
When epsilon = 0.2386, it minimizes MSE.



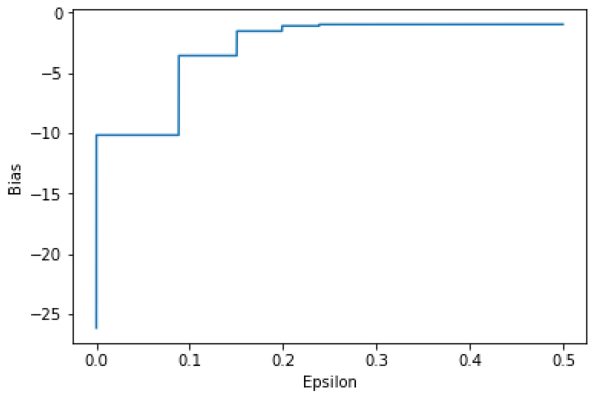
T =1, X = 2000:

When epsilon = 0.2721, it minimizes MSE.



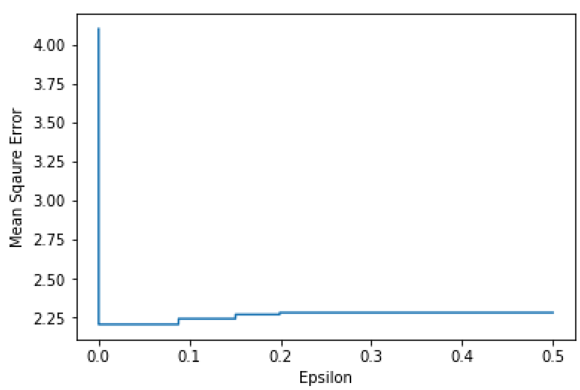


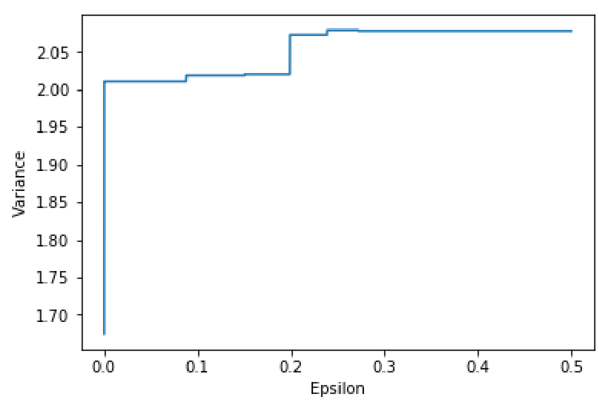
When epsilon = 0.2721, it minimizes MSE.

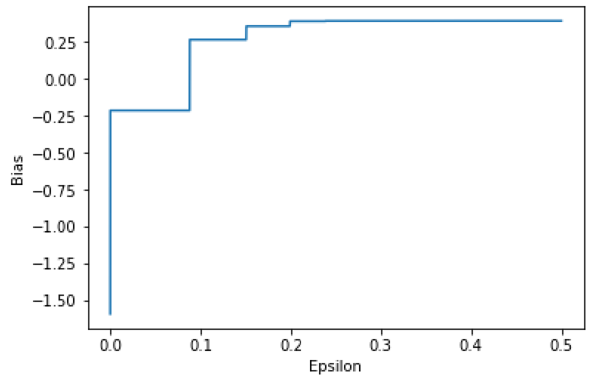


T = 1, X = 2500:

When epsilon = 0.001, it minimizes MSE.

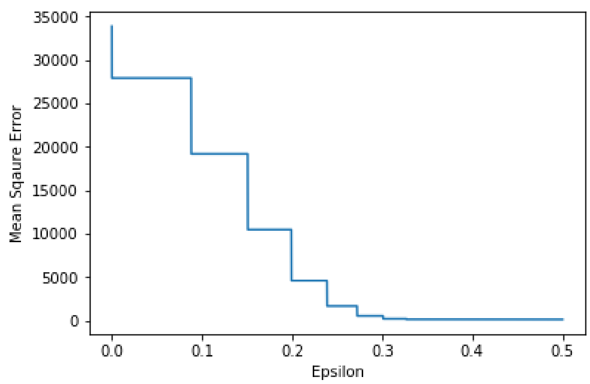


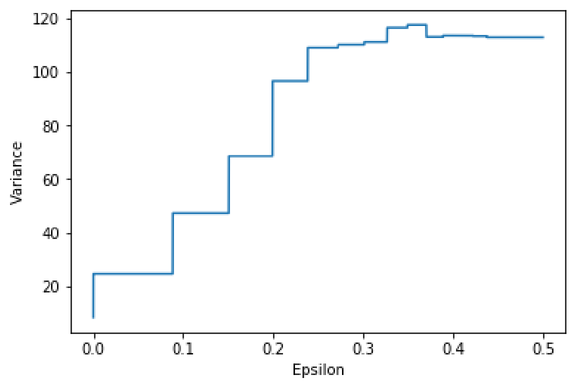




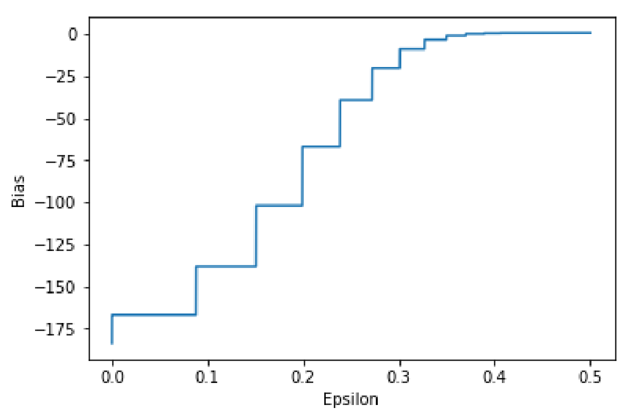
T = 3, X = 1500:

hen epsilon = 0.3702, it minimizes MSE.



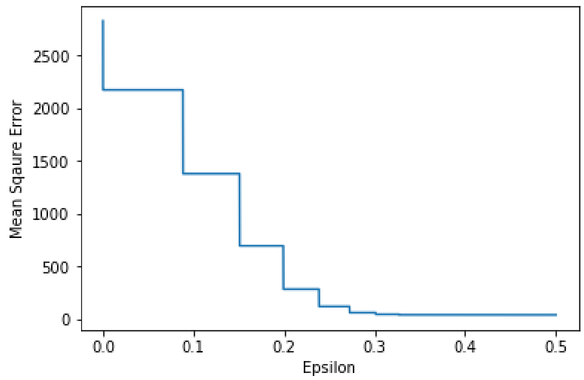


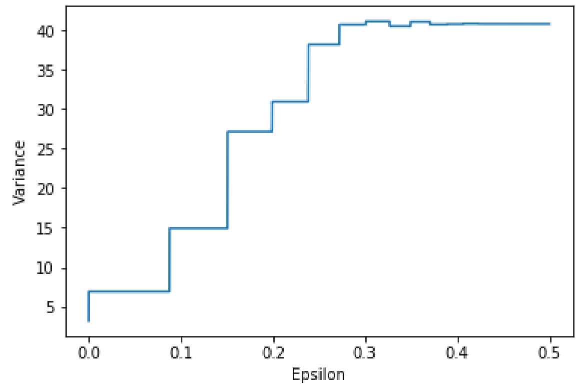
When epsilon = 0.3702, it minimizes MSE.

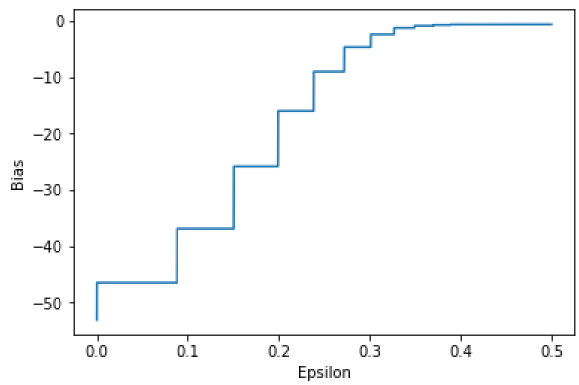


T = 3, X = 2000:

When epsilon = 0.3891, it minimizes MSE.

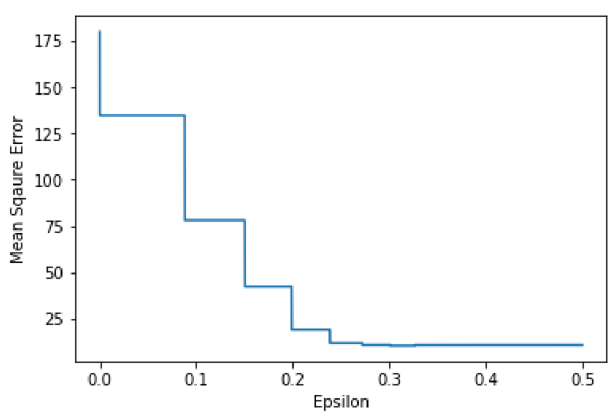


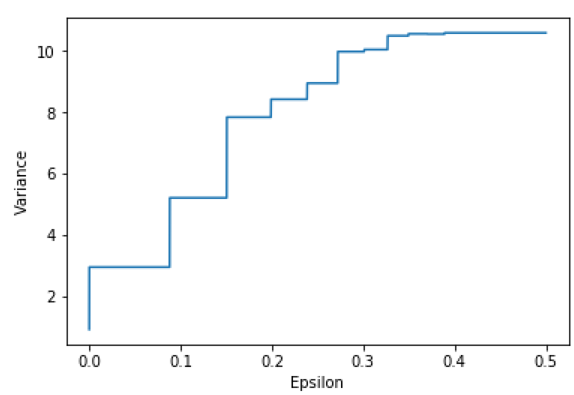


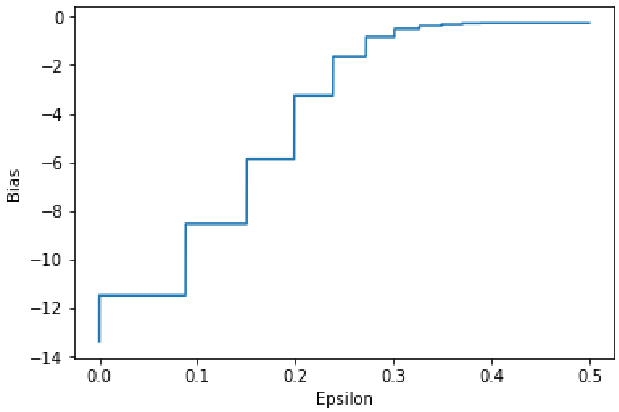


T = 3, X = 2500:

When epsilon = 0.3011, it minimizes MSE.







Bias
Epsilon
0
−2
−4
−6
−8
−10
−12
−14
0.0
0.1
0.2
0.3
0.4
0.5