1 The Kou jump-diffusion model

By [?], the risk-neutral dynamics for log asset price $\log S_t$ under the DEJD model can be expressed as

$$d\log S_t = \left(r - d - \lambda\bar{\mu} - \frac{1}{2}\sigma^2\right)dt + \sigma dW_t^{\mathbb{Q}} + dZ_t.$$

Here, $W_t^{\mathbb{Q}}$ is a standard Brownian motion; Z_t is a compound Poisson process with the representation $Z_t = \sum_{i=1}^{N_t} Y_i$, where N_t is a Poisson process with intensity λ and $\{Y_i\}_{i=1}^{\infty}$ is a sequence of i.i.d. random variables following the double exponential distribution with the density

$$f_Y(y) := p_Y\eta_1 e^{-\eta_1 y}1_{\{y\geq 0\}} + (1-p_Y)\eta_2 e^{\eta_2 y}1_{\{y<0\}}, \tag{1}$$

for some constants $p_Y \in [0,1]$, $\eta_1 > 1$, and $\eta_2 > 0$; $\bar{\mu}$ is given by $\bar{\mu} = \mathbb{E}^{\mathbb{Q}}[e^{Y_1}] - 1$.

2 Density tail calculation

We need to calculate the tail for the density of $X_t = \log S_t$. However, we find that existing transform inversion algorithms are not easy to use. Here, I propose a route such that the load for transform inversionis minimized. Consider

$$X_t = X_0 + ct + \sigma W_t^{\mathbb{Q}} + Z_t.$$

We have

$$\begin{aligned}
\mathbb{Q}(X_t \leq x) &= \mathbb{Q}(X_0 + ct + \sigma W_t^{\mathbb{Q}} + Z_t \leq x) \\
&= \int \mathbb{Q}(X_0 + ct + \sigma W_t^{\mathbb{Q}} + Z_t \leq x | W_t^{\mathbb{Q}} = w)\mathbb{Q}(W_t^{\mathbb{Q}} \in dw) \\
&= \int \mathbb{Q}(Z_t \leq x - [X_0 + ct + \sigma w] | W_t^{\mathbb{Q}} = w)\mathbb{Q}(W_t^{\mathbb{Q}} \in dw).
\end{aligned}$$

Here, we have

$$\begin{aligned}
\mathbb{Q}(Z_t &\leq x - [X_0 + ct + \sigma w] | W_t^{\mathbb{Q}} = w) \\
&= \mathbb{Q}(\textstyle\sum_{i=1}^{N_t} Y_i \leq x - [X_0 + ct + \sigma w] | W_t^{\mathbb{Q}} = w) \\
&= \sum_{n=0}^{\infty} \mathbb{Q}(\textstyle\sum_{i=1}^{n} Y_i \leq x - [X_0 + ct + \sigma w] | W_t^{\mathbb{Q}} = w, N_t = n)\mathbb{Q}(N_t = n) \\
&= \sum_{n=0}^{\infty} \mathbb{Q}(\textstyle\sum_{i=1}^{n} Y_i \leq x - [X_0 + ct + \sigma w])\mathbb{Q}(N_t = n).
\end{aligned}$$

So, we have

$$\begin{aligned}
\mathbb{Q}(X_t \leq x) &= \int \sum_{n=0}^{\infty} \mathbb{Q}(\textstyle\sum_{i=1}^{n} Y_i \leq x - [X_0 + ct + \sigma w])\mathbb{Q}(N_t = n)\mathbb{Q}(W_t^{\mathbb{Q}} \in dw) \\
&= \sum_{n=0}^{\infty} \left[\int \mathbb{Q}(\textstyle\sum_{i=1}^{n} Y_i \leq x - [X_0 + ct + \sigma w])\mathbb{Q}(W_t^{\mathbb{Q}} \in dw)\right]\mathbb{Q}(N_t = n).
\end{aligned}$$

Then, we have the following formal expression of the density:

$$\frac{d}{dx}\mathbb{Q}(X_t \le x) = \sum_{n=0}^{\infty} \left[\int \frac{d}{dx}\mathbb{Q}(\sum_{i=1}^{n} Y_i \le x - [X_0 + ct + \sigma w])\mathbb{Q}(W_t^{\mathbb{Q}} \in dw) \right] \mathbb{Q}(N_t = n).$$

Using transform inversion to calculate the density

$$\frac{d}{dx}\mathbb{Q}(\sum_{i=1}^{n} Y_i \le x - [X_0 + ct + \sigma w])$$

is relatively easy and is now a very "clear" task. This should be fast and accurate! Test it first. Then, calculate

$$\sum_{n=0}^{N} \left[\int \frac{d}{dx}\mathbb{Q}(\sum_{i=1}^{n} Y_i \le x - [X_0 + ct + \sigma w])\mathbb{Q}(W_t^{\mathbb{Q}} \in dw) \right] \mathbb{Q}(N_t = n)$$

Needless to say, you need to plug in $\mathbb{Q}(W_t^{\mathbb{Q}} \in dw)$ and then calculate the numerical integration and also plug in $\mathbb{Q}(N_t = n)$. In numerical test, increase N until the value of the above series almost does not change!

I sense that this is a better way to calculate the density, since we have disentangled the structure!

3 Algorithm Construction

In this section, we introduce the procedures of our algorithm for calculating density of Kou model.

Input time parameter t, starting location X_0, drift parameter c, diffusion parameter σ, density parameter x, Poisson intensity λ, double exponential distribution related parameters p_Y, eta_1, eta_2 and truncation level N.

Procedure 1 Given constant $n \le N$ and w, we calculate

$$\frac{d}{dx}\mathbb{Q}(\sum_{i=1}^{n} Y_i \le x - [X_0 + ct + \sigma w]).$$

Here we apply characteristic function method, i.e, we first calculate the characteristic function of Y and then apply inverse transform formula to obtain the density of $\sum_{i=1}^{n} Y_i$. We note in particular that simply use involution method is infeasible when n is large. The related computational efforts is unbearable.

Procedure 2 Calculate integral

$$\int \frac{d}{dx}\mathbb{Q}(\sum_{i=1}^{n} Y_i \le x - [X_0 + ct + \sigma w])\mathbb{Q}(W_t^{\mathbb{Q}} \in dw).$$

Here $\mathbb{Q}(W_t^{\mathbb{Q}} \in dw) = \frac{1}{\sqrt{2\pi t}} e^{-\frac{w^2}{2t}}$.

Procedure 3 Calculate

$$\sum_{n=0}^{N}\left[\int \frac{d}{dx}\mathbb{Q}(\sum\nolimits_{i=1}^{n} Y_i \leq x - [X_0 + ct + \sigma w])\mathbb{Q}(W_t^{\mathbb{Q}} \in dw)\right]\mathbb{Q}(N_t = n)$$

by summing n over $0, 1, \cdots, N$ and output the calculation result.

Procedure 4 Gradually increase N until the output is stable. Output both probability density function (or cumulative distribution function) and truncation level N.

Parameter Choice Following Section 3 of Kou 2002, 'A Jump-Diffusion Model for Option Pricing', we temporarily choose parameters $t = 10$, $\sigma = 0.2$, $c = 0.15$, $\lambda = 0.04$, $p_Y = 0.3$ and plot its density in Figure 6.

4 Report

After running numerical experiments, we found that, under our current parameter choices, letting truncation level $N = 9$ is sufficient to guarantee accuracy while costing only bearable computational efforts. Indeed, choosing a larger N will only lead to negligible improvement in accuracy, but wasting much more computation time at cost, as shown by experiment data, i.e, see Figure 7. This finding supports our N-choice.

In the rest part of this section, we present figures to illustrate the effects of this algorithm. In Figures from 1 to 7, we show the probability density function of $X = X_t$ when N increases from 0 to 9, respectively. Furthermore, we directly simulate X for 100,000 times and draw empirical probability density function in Figure 8. This figure serves as a justification for our algorithm, i.e, the shape and scale are approximately the same as Figure 7 with $N = 9$.

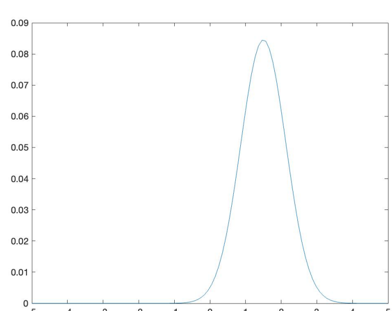


Figure 1: Probability Density Function of X when n = 0

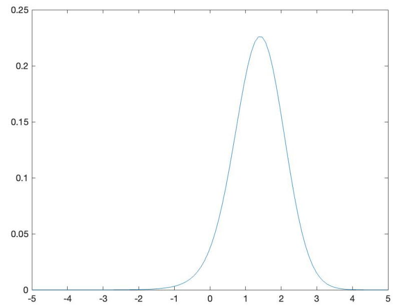


Figure 2: Probability Density Function of X when N = 1

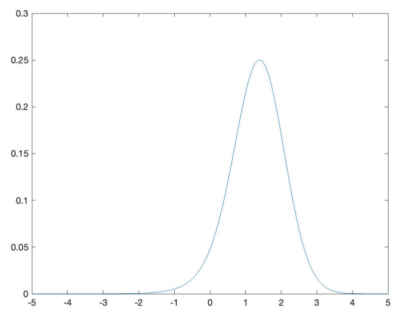


Figure 3: Probability Density Function of X when N = 2

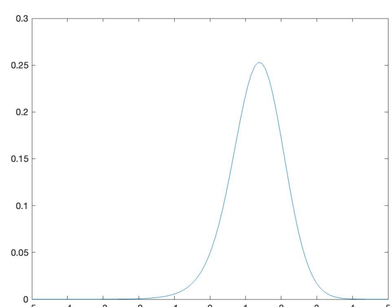


Figure 4: Probability Density Function of X when N = 3

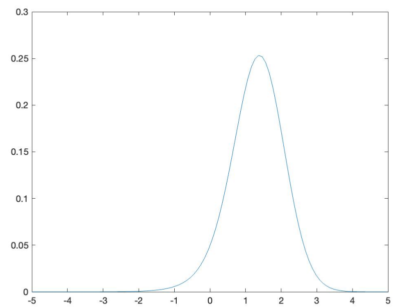


Figure 5: Probability Density Function of X when N = 4

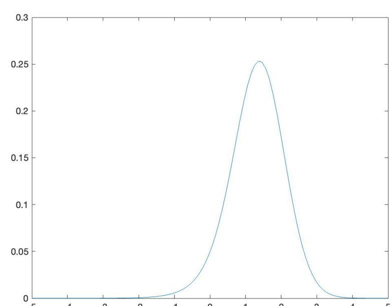
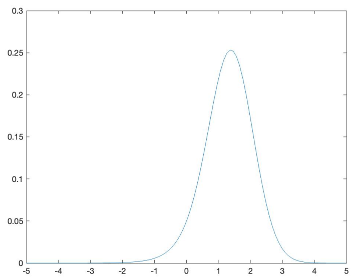
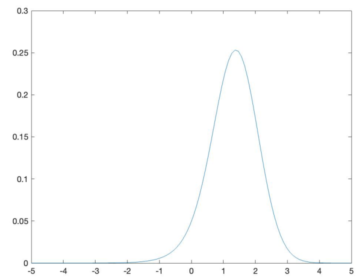


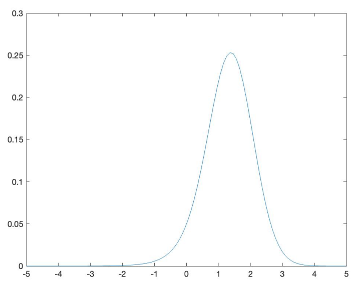
Figure 6: Probability Density Function of X when N = 5



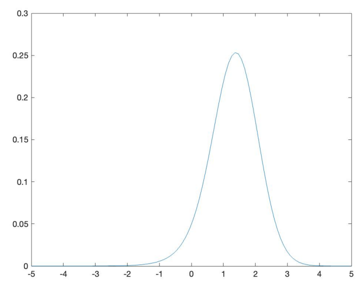
(a) N = 6



(b) N = 7



(c) N = 8



(d) N = 9

Figure 7: Probability Density Function of X when N = 6, 7, 8, 9

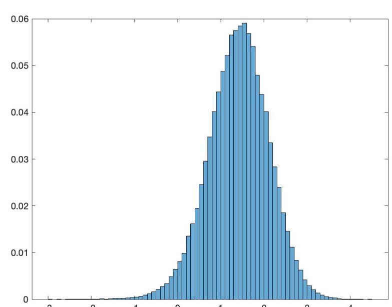


Figure 8: Simulation of X

5 Results Reproduction

To reproduce these results, you may follow the procedures below.

1. Running the script 'model.m' to load all the parameters of this model.
2. Running the script 'cal_data.m' to partly finish the inner calculation.
3. Running the script 'running.m' to obtain the results when N equals from 0 to 9, and obtain results from N0 to N9. Then running the command 'plot(x, N0)' can obtain the function images. It may take hours to finish this part of calculation under transform inversion method.
4. Running the script 'Simulation_X' and then obtain the distribution of X by simulation method.