IEOR4735 Final Project

Chun Yat Yeung (cy2623)
Jinyi Bai (jb4601)
Wai Jing Tsoi (wt2329)

December, 2022

Abstract

We price a quanto spot-rate product option using diffusion model and short-rate model.

Contents

1 Quanto Spot-Rate Product Option

Label time points $0 = t = T_0 < T_1 < ... < T_i = T < T_{i+1}$, where $T_i = T$ is the contract valuation date and T_{i+1} is the contract settlement date, with time interval $\tau_j = T_j - T_{j-1}$. Choose the last time interval $\tau_{i+1} \equiv \Delta$ to be 3 months. Observing at time t, define the following market variables

- Nikkei-225 spot price, S_t^f;
- JPY-USD FX rate, X_t;
- Nikkei-225 spot price quantoed from JPY to USD, $S_t \equiv S_t^f \cdot 1$;
- LIBOR forward rate between T_{j-1} and T_j, $L_t(T_{j-1}, T_j) \equiv L_t^j$.

Note that for simplicity, index subscript, say, i refers to time T_i.

Consider the following contract payoff at time $T_{i+1} = T + \Delta$

$$\Pi_{i+1} = \left[\left(\frac{S_i}{S_0} - k \right) \cdot \left(k' - \frac{L_i^{i+1}}{L_0^{i+1}} \right) \right]^+ \tag{1}$$

where k, k' are respectively strikes for spot and rate. At inception, the contract values

$$\Pi_0 = Ee^{-\int_0^{T_{i+1}} r_u du} \left[\left(\frac{S_i}{S_0} - k \right) \cdot \left(k' - \frac{L_i^{i+1}}{L_0^{i+1}} \right) \right]^+ \tag{2}$$

under the risk-neutral measure $\mathbb{Q}$. To the first order, Π_0 is a function of spot volatility, rate volatility and spot-rate correlation.

For now, ignore the quanto component and model the spot directly under the domestic risk-neutral measure.

2 T-Forward Measure

The risk-neutral measure $\mathbb{Q}$ adopts cash account $B(t)$ as numéraire while the T-forward measure $\mathbb{Q}^T$ adopts bond price $P(t, T)$ expiring time T as numéraire. The change of measure reads

$$\frac{d\mathbb{Q}^T}{d\mathbb{Q}} = \frac{P(t, T)/P(0, T)}{B(t)/B(0)}.$$

Under the T_{i+1}-forward measure $\mathbb{Q}^{i+1}$, discount factor drops out, thus

$$\Pi_0 = P(0, T_{i+1}) E^{i+1} \left[\left(\frac{S_i}{S_0} - k \right) \cdot \left(k' - \frac{L_i^{i+1}}{L_0^{i+1}} \right) \right]^+ . \tag{3}$$

As L_t^{i+1} is deflated by $P(t, T_{i+1})$, or mathematically,

$$L_t^{i+1} = \frac{1}{\tau_{i+1}} \left(\frac{P(t, T_i)}{P(t, T_{i+1})} - 1 \right),$$

L_t^{i+1} is a $\mathbb{Q}^{i+1}$-martingale.

3 Diffusion Model

In this approach, we stick with the minimal modeling assumptions to gain intuition on the dependence of contract price Π_0 on market parameters, such as volatilities and correlations. Later on, these are relaxed.

Define the stochastic exponential $\mathcal{E}$, a martingale, which for a generic stochastic process Y_t,

$$\mathcal{E}(Y_t) = \exp(Y_t - \langle Y_t \rangle /2).$$

We model under $\mathbb{Q}^{i+1}$ for rate. Assume $L_t^{i+1} \equiv L_t(T, T+\Delta)$ follows a Brownian diffusion with constant volatility σ^{i+1}

$$dL_t^{i+1} = \sigma^{i+1} L_t^{i+1} dW_t^{i+1}, \tag{4}$$

so at time T_i,

$$\frac{L_i^{i+1}}{L_0^{i+1}} = \mathcal{E}(\sigma^{i+1} W_i^{i+1}),$$

where W^{i+1} is a $\mathbb{Q}^{i+1}$-Brownian motion. Under this assumption, a caplet prices according to the Black-76 formula, and σ^{i+1} may be directly read off the market. More discussion on calibration aspects later.

But the spot component values at time T_i, so we model the T_i-forward spot as a $\mathbb{Q}^i$-martingale. Assume again a Brownian diffusion with constant volatility σ

$$d\left(\frac{S_t}{P(t,T_i)}\right) = \left(\frac{S_t}{P(t,T_i)}\right) \sigma dZ_t^i, \tag{5}$$

so at time T_i,

$$\frac{S_i}{S_0} = \frac{1}{P(0,T_i)} \mathcal{E}(\sigma Z_i^i),$$

where Z^i is a $\mathbb{Q}^i$-Brownian motion.

Now, we require a joint modeling of W^{i+1} and Z^i, by expressing Z^i in terms of W^{i+1} and $W^{i+1,\perp}$. Consider the change of measure from $\mathbb{Q}^{i+1}$ to $\mathbb{Q}^i$

$$Q_i^{i+1}(t) \equiv \frac{d\mathbb{Q}^i}{d\mathbb{Q}^{i+1}} = \frac{P(t,T_i)/P(0,T_i)}{P(t,T_{i+1})/P(0,T_{i+1})} = \frac{P(0,T_{i+1})}{P(0,T_i)}(1 + \tau_{i+1} L_t^{i+1}),$$

with differential

$$dQ_i^{i+1}(t) = \frac{P(0,T_{i+1})}{P(0,T_i)} \tau_{i+1} dL_t^{i+1} = \frac{P(0,T_{i+1})}{P(0,T_i)} \tau_{i+1} \sigma^{i+1} L_t^{i+1} dW_t^{i+1} = Q_i^{i+1}(t) \frac{\tau_{i+1} L_t^{i+1}}{1 + \tau_{i+1} L_t^{i+1}} \sigma^{i+1} dW_t^{i+1}. \tag{6}$$

By Girsanov theorem,

$$dW_t^i = dW_t^{i+1} - \frac{\tau_{i+1} L_t^{i+1}}{1 + \tau_{i+1} L_t^{i+1}} \sigma^{i+1} dt$$

and for an independent Brownian motion $W^{i,\perp}$,

$$dW_t^{i,\perp} = dW_t^{i+1,\perp}.$$

Assume spot-rate correlation ρ. Then,

$$dZ_t^i = \rho dW_t^i + \sqrt{1-\rho^2} dW_t^{i,\perp} = \rho dW_t^{i+1} + \sqrt{1-\rho^2} dW_t^{i+1,\perp} - \frac{\tau_{i+1} L_t^{i+1}}{1 + \tau_{i+1} L_t^{i+1}} \rho \sigma^{i+1} dt. \tag{7}$$

Integrating,

$$Z_i^i = \rho W_i^{i+1} + \sqrt{1-\rho^2} W_i^{i+1,\perp} - \rho\sigma^{i+1} \int_0^{T_i} \frac{\tau_{i+1} L_u^{i+1}}{1+\tau_{i+1} L_u^{i+1}} du,$$

which unfortunately depends on the path of L_t^{i+1}. By Itô product rule, one may rewrite the time integral as a stochastic integral.

The contract thus admits no closed-form price.

Remark. *If instead we choose to model the T_{i+1}-forward spot as a $\mathbb{Q}^{i+1}$-martingale (diffusing with Z^{i+1} etc.), the spot component in contract payoff then carries a factor involving $P(T_i, T_{i+1})$, making the payoff even more coupled. So we choose to model the spot under $\mathbb{Q}^i$.*

3.1 Small Volatility of Rate Approximation

Consider the path integral in Z_i^i. Assuming a small volatility of rate σ^{i+1} and a small initial forward rate L_0^{i+1},

$$\begin{aligned}
&\int_0^{T_i} \frac{\tau_{i+1} L_u^{i+1}}{1+\tau_{i+1} L_u^{i+1}} du \\
\approx &\int_0^{T_i} \tau_{i+1} L_u^{i+1} du \\
= &\,\tau_{i+1} \int_0^{T_i} d(u L_u^{i+1}) - u dL_u^{i+1} \\
= &\,\tau_{i+1} \left(T_i L_i^{i+1} - \int_0^{T_i} u \sigma^{i+1} L_u^{i+1} dW_u^{i+1} \right) \\
\approx &\,\tau_{i+1} T_i L_i^{i+1} \left(1 - \sigma^{i+1} \int_0^{T_i} \frac{u}{T_i} dW_u^{i+1} \right) \\
\approx &\,\tau_{i+1} T_i L_0^{i+1} \left(1 - \sigma^{i+1} \int_0^{T_i} \frac{u}{T_i} dW_u^{i+1} \right).
\end{aligned}$$

As $\int_0^{T_i} \frac{u}{T_i} dW_u^{i+1}$ is an Itô integral, it is normally distributed, following $N(0, T_i/3)$. But it correlates with W_i^{i+1} via covariance

$$E^{i+1} W_i^{i+1} \int_0^{T_i} \frac{u}{T_i} dW_u^{i+1} = \frac{T_i}{2},$$

and correlation

$$\bar{\rho} = \frac{T_i/2}{\sqrt{T_i}\sqrt{T_i/3}} = \frac{\sqrt{3}}{2}.$$

Thus, relabeling, Z_i^i is approximated by three independent Brownian motions $W_i^{1,2,3}$

$$\begin{aligned}
Z_i^i &= \rho W_i^1 + \sqrt{1-\rho^2} W_i^2 - \rho\sigma^{i+1}\tau_{i+1} T_i L_0^{i+1} \left(1 - \frac{\sigma^{i+1}}{\sqrt{3}} \frac{\sqrt{3} W_i^1 + W_i^3}{2} \right) \\
&= \rho \left(1 + \frac{(\sigma^{i+1})^2 T_i \tau_{i+1} L_0^{i+1}}{2} \right) W_i^1 + \sqrt{1-\rho^2} W_i^2 + \frac{(\sigma^{i+1})^2 T_i \tau_{i+1} L_0^{i+1}}{2\sqrt{3}} W_i^3 - \rho\sigma^{i+1}\tau_{i+1} T_i L_0^{i+1}.
\end{aligned}$$

Denote dimensionless constant $\alpha \equiv (\sigma^{i+1})^2 T_i \tau_{i+1} L_0^{i+1}$. Then,

$$\begin{aligned}
Z_i^i &= \rho \left(1 + \frac{\alpha}{2} \right) W_i^1 + \sqrt{1-\rho^2} W_i^2 + \frac{\alpha}{2\sqrt{3}} W_i^3 - \rho \frac{\alpha}{\sigma^{i+1}} \\
\langle Z_i^i \rangle &= \left[\rho^2 \left(1 + \frac{\alpha}{2} \right)^2 + (1-\rho^2) + \frac{\alpha^2}{12} \right] T_i = \left[1 + \rho^2 \alpha + \left(\frac{1}{3} + \rho^2 \right) \frac{\alpha^2}{4} \right] T_i.
\end{aligned}$$

Recall that spot and rate evolve as

$$\frac{S_i}{S_0} = \frac{1}{P(0,T_i)}\mathcal{E}(\sigma Z_i^i) \approx \frac{1}{P(0,T_i)} \exp\left(-\frac{\left[1+\rho^2\alpha + (1/3+\rho^2)\alpha^2/4\right]\sigma^2 T_i}{2} + \sigma Z_i^i\right)$$
$$\frac{L_i^{i+1}}{L_0^{i+1}} = \mathcal{E}(\sigma^{i+1} W_i^1) = \exp\left(-\frac{(\sigma^{i+1})^2 T_i}{2} + \sigma^{i+1} W_i^1\right).$$

To price the contract, a triple integral over $W_i^{1,2,3}$ is involved. To further simplify, approximate $\alpha \approx 0$, so

$$\frac{S_i}{S_0} \approx \frac{1}{P(0,T_i)} \exp\left(-\frac{\sigma^2 T_i}{2} + \sigma\left(\rho W_i^1 + \sqrt{1-\rho^2} W_i^2\right)\right)$$
$$\frac{L_i^{i+1}}{L_0^{i+1}} = \exp\left(-\frac{(\sigma^{i+1})^2 T_i}{2} + \sigma^{i+1} W_i^1\right).$$

We have the contract price, using iterated expectations,

$$\begin{aligned}
\Pi_0 &= P(0,T_{i+1}) E^{i+1}\left[\left(\frac{S_i}{S_0} - k\right)\cdot\left(k' - \frac{L_i^{i+1}}{L_0^{i+1}}\right)\right]^+ \\
&\approx \frac{1}{1+\tau_{i+1}L_0^{i+1}} E^{i+1}\left[\left(\mathcal{E}\left(\sigma\left(\rho W_i^1 + \sqrt{1-\rho^2}W_i^2\right)\right) - kP(0,T_i)\right)\cdot\left(k' - \mathcal{E}\left(\sigma^{i+1}W_i^1\right)\right)\right]^+ \\
&\approx \frac{1}{1+\tau_{i+1}L_0^{i+1}} E^{i+1}\left[\left(k' - \mathcal{E}\left(\sigma^{i+1}W_i^1\right)\right)^+ \cdot E_{W^1}^{i+1}\left(\mathcal{E}\left(\sigma\left(\rho W_i^1 + \sqrt{1-\rho^2}W_i^2\right)\right) - kP(0,T_i)\right)^+ + \right. \\
&\quad \left. \left(\mathcal{E}\left(\sigma^{i+1}W_i^1\right) - k'\right)^+ \cdot E_{W^1}^{i+1}\left(kP(0,T_i) - \mathcal{E}\left(\sigma\left(\rho W_i^1 + \sqrt{1-\rho^2}W_i^2\right)\right)\right)^+\right]
\end{aligned}$$

By Black-Scholes formula, for the conditional expectations, we have

$$E_{W^1}^{i+1}\left(\mathcal{E}\left(\sigma\left(\rho W_i^1 + \sqrt{1-\rho^2}W_i^2\right)\right) - kP(0,T_i)\right)^+ = \mathcal{E}(\rho\sigma W_i^1)N(d_1) - kP(0,T_i)N(d_2)$$
$$E_{W^1}^{i+1}\left(kP(0,T_i) - \mathcal{E}\left(\sigma\left(\rho W_i^1 + \sqrt{1-\rho^2}W_i^2\right)\right)\right)^+ = kP(0,T_i)N(-d_2) - \mathcal{E}(\rho\sigma W_i^1)N(-d_1)$$

where

$$d_{1,2} = \frac{\log \mathcal{E}(\rho\sigma W_i^1)/kP(0,T_i)}{\sqrt{1-\rho^2}\sigma\sqrt{T_i}} \pm \frac{\sqrt{1-\rho^2}\sigma\sqrt{T_i}}{2}.$$

Finally, contract price reads

$$\begin{aligned}
\Pi_0(k,k';\rho,\sigma,\sigma^{i+1}) \approx \frac{1}{1+\tau_{i+1}L_0^{i+1}} E^{i+1}&\left[\left(k' - \mathcal{E}\left(\sigma^{i+1}W_i^1\right)\right)^+ \cdot \left(\mathcal{E}(\rho\sigma W_i^1)N(d_1) - kP(0,T_i)N(d_2)\right) + \right. \\
&\left.\left(\mathcal{E}\left(\sigma^{i+1}W_i^1\right) - k'\right)^+ \cdot \left(kP(0,T_i)N(-d_2) - \mathcal{E}(\rho\sigma W_i^1)N(-d_1)\right)\right].
\end{aligned} \tag{8}$$

In fact, the expectation can be explicitly evaluated, but the computation is extremely involved. So instead, we resort to Monte-Carlo simulation. Note that by iterated expectations, we have reduced the bivariate expectation to a univariate expectation, enabling efficient simulation.

3.2 Quanto Component

We want the diffusion dynamics of the quantoed spot $S_t \equiv S_t^f \cdot 1$ under the domestic risk-neutral measure $\mathbb{Q}^d$. Suppose that under the foreign measure,

$$\frac{dS_t^f}{S_t^f} = r_t^f dt + \sigma^f dZ_t^f.$$

Consider FX rate X_t s.t. the comboed spot $S_t^d \equiv S_t^f \cdot X_t$ evolves as

$$\begin{aligned}\frac{dS_t^d}{S_t^d} &= r_t^d dt + \sigma^d dZ_t^d \\ \frac{dX_t}{X_t} &= (r_t^d - r_t^f)dt + \sigma^X dW_t^d\end{aligned}$$

with $dZ_t^d dW_t^d = dZ_t^f dW_t^d = \rho^X dt$ and $(\sigma^d)^2 = (\sigma^f)^2 + (\sigma^X)^2 + 2\rho^X \sigma^f \sigma^X$.

The drift of X_t is chosen s.t. $S_t^f X_t / B_t^d$ is a $\mathbb{Q}^d$-martingale.

Now, establish the change of measure from $\mathbb{Q}^d$ to $\mathbb{Q}^f$

$$\frac{d\mathbb{Q}^f}{d\mathbb{Q}^d} = \frac{X_t B_t^f / X_0 B_0^f}{B_t^d / B_0^d} = \mathcal{E}(\sigma^X W_t^d) = \mathcal{E}\left(\sigma^X \left(\rho^X Z_t^d + \sqrt{1-(\rho^X)^2} Z_t^{d,\perp}\right)\right),$$

thus

$$dZ_t^f = dZ_t^d - \rho^X \sigma^X dt.$$

The dynamics of the quantoed spot under $\mathbb{Q}^d$ is

$$\frac{dS_t}{S_t} = (r_t^f - \rho^X \sigma^X \sigma^f)dt + \sigma^f dZ_t^d = \left[r_t^d - (r_t^d - r_t^f + \rho^X \sigma^X \sigma^f)\right] dt + \sigma^f dZ_t^d, \tag{9}$$

so $e^{\int_0^t (r_u^d - r_u^f + \rho^X \sigma^X \sigma^f)du} S_t / B_t^d$ is a $\mathbb{Q}^d$-martingale, and $e^{\int_0^t (r_u^d - r_u^f + \rho^X \sigma^X \sigma^f)du} S_t / P(t, T_i)$ is a $\mathbb{Q}^i$-martingale.

Our pricing formulas above still apply, except with transformation

$$S_i \to e^{\int_0^{T_i} (r_u^d - r_u^f + \rho^X \sigma^X \sigma^f)du} S_i. \tag{10}$$

Consider the factor $e^{\int_0^{T_i} (r_u^d - r_u^f + \rho^X \sigma^X \sigma^f)du}$. Now, unfortunately path dependence of rate kicks in and we get a few more parameters to calibrate e.g. ρ^X, σ^X and two short-rate processes to model. As these are sub-dominant stochastic components relative to S_i and L_i^{i+1}, in this approach, to avoid over-complicating things, we will just ignore this factor. In other words, we assume $e^{\int_0^{T_i} (r_u^d - r_u^f + \rho^X \sigma^X \sigma^f)du} S_i$, as a whole, follows a Brownian diffusion, possibly with some volatility adjustment $\Delta\sigma$.

3.3 Calibration

A caplet is a call option on LIBOR spot rate, which prices as

$$\begin{aligned}C_0^L(R, T_i) &= \tau_{i+1} E e^{-\int_0^{T_{i+1}} r_u du} (L_i^{i+1} - R)^+ \\ &= \tau_{i+1} P(0, T_{i+1}) E^{i+1} (L_i^{i+1} - R)^+.\end{aligned}$$

Market quoting convention assumes that L_t^{i+1} follows a Brownian diffusion, thus by Black-76 formula,

$$C_0^L(R, T_i) = \tau_{i+1} P(0, T_{i+1}) (L_0^{i+1} N(d_1) - R N(d_2))$$

where

$$d_{1,2} = \frac{\log L_0^{i+1}/R}{\sigma^{i+1}\sqrt{T_i}} \pm \frac{\sigma^{i+1}\sqrt{T_i}}{2}.$$

Write in terms of relative strike $R = kL_i^{i+1}$, then

$$C_0^L(k, T_i) = \tau_{i+1} P(0, T_{i+1}) L_0^{i+1} (N(d_1) - kN(d_2)) \tag{11}$$

where

$$d_{1,2} = -\frac{\log k}{\sigma^{i+1}\sqrt{T_i}} \pm \frac{\sigma^{i+1}\sqrt{T_i}}{2}.$$

By matching to a caplet/floorlet struck at k' expring T_i, we back out the Black volatility σ^{i+1}.

Similarly, for the quantoed spot, we have the Nikkei-225 call price quoted in Black-Scholes

$$C_0^S(k, T_i) = S_0(N(d_1) - kP(0, T_i)N(d_2)) \tag{12}$$

where

$$d_{1,2} = -\frac{\log k}{\sigma\sqrt{T_i}} \pm \frac{\sigma\sqrt{T_i}}{2}.$$

By matching to a call/put struck at k expring T_i, we back out the implied volatility σ.

Both σ^{i+1}, σ can be directly read off market as LIBOR caplets and Nikkei calls are liquid.

However, there is no liquid derivative from which we can imply the spot-rate correlation ρ, under measure $\mathbb{Q}$. But ρ is preserved going from real-world measure $\mathbb{P}$ to risk-neutral measure $\mathbb{Q}$. So we can estimate the rough order of ρ from historical time series of Nikkei and LIBOR returns, and project the number into future.

3.4 Monte-Carlo Simulation

We price the contract with the small volatility of rate approximation. The price reads

$$\begin{aligned}\Pi_0(k, k'; \rho, \sigma, \sigma^{i+1}) \approx \frac{1}{1 + \tau_{i+1} L_0^{i+1}} E^{i+1} \Big[&\left(k' - \mathcal{E}\left(\sigma^{i+1} W_i^1\right)\right)^+ \cdot \left(\mathcal{E}(\rho\sigma W_i^1) N(d_1) - kP(0, T_i) N(d_2)\right) + \\ &\left(\mathcal{E}\left(\sigma^{i+1} W_i^1\right) - k'\right)^+ \cdot \left(kP(0, T_i) N(-d_2) - \mathcal{E}(\rho\sigma W_i^1) N(-d_1)\right) \Big]\end{aligned}$$

where

$$d_{1,2} = \frac{\log \mathcal{E}(\rho\sigma W_i^1)/kP(0, T_i)}{\sqrt{1-\rho^2}\sigma\sqrt{T_i}} \pm \frac{\sqrt{1-\rho^2}\sigma\sqrt{T_i}}{2}.$$

Further simplify by absorbing $P(0, T_i)$ into k and omitting the discount factor $1/(1 + \tau_{i+1} L_0^{i+1})$, which only scales the price. Effectively, $P(0, T_i) = 1$ and $L_0^{i+1} = 0$.

Simulation is straightforward by repeatedly sampling W_i^1, stubstituing and averaging. As only one random variable is involved, it can be vectorized in a simple manner.

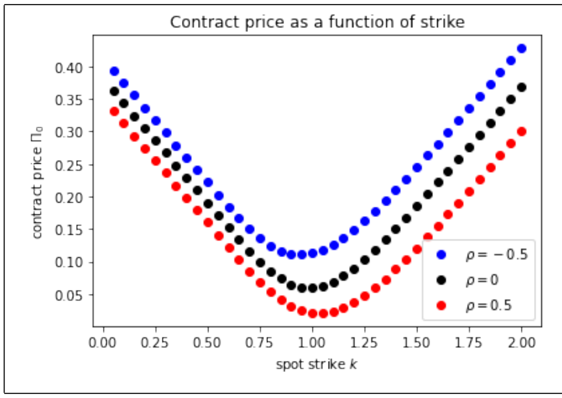
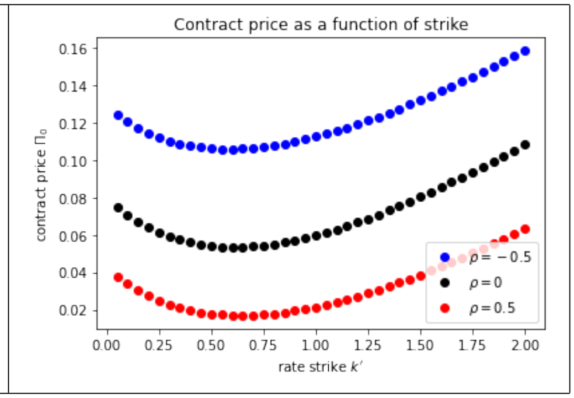
The pricing routine is provided in section 6.1.

3.5 Parameter Sensitivity

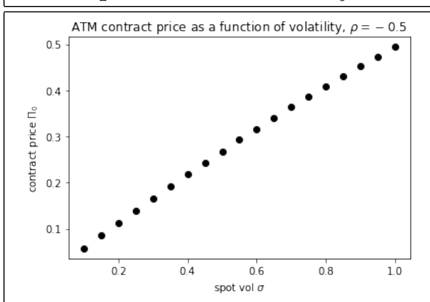
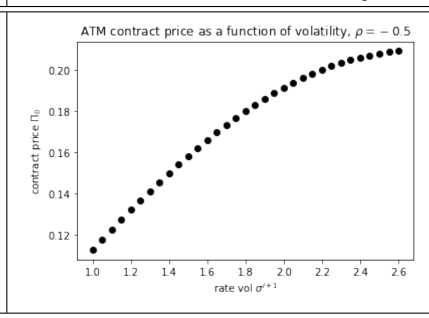
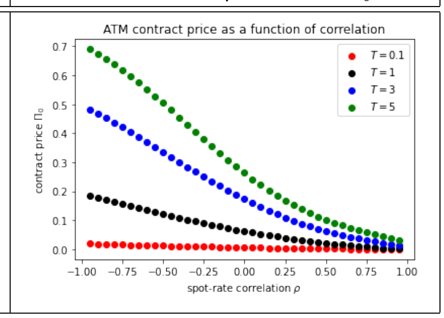
For the contract terms, fix $T_i = T = 1, \Delta = 0.25$, while for the diffusion dynamics, fix $\sigma^{i+1} = 100\%, \sigma = 20\%$.

We visualize the contract price as a function of both contractual and market parameters.

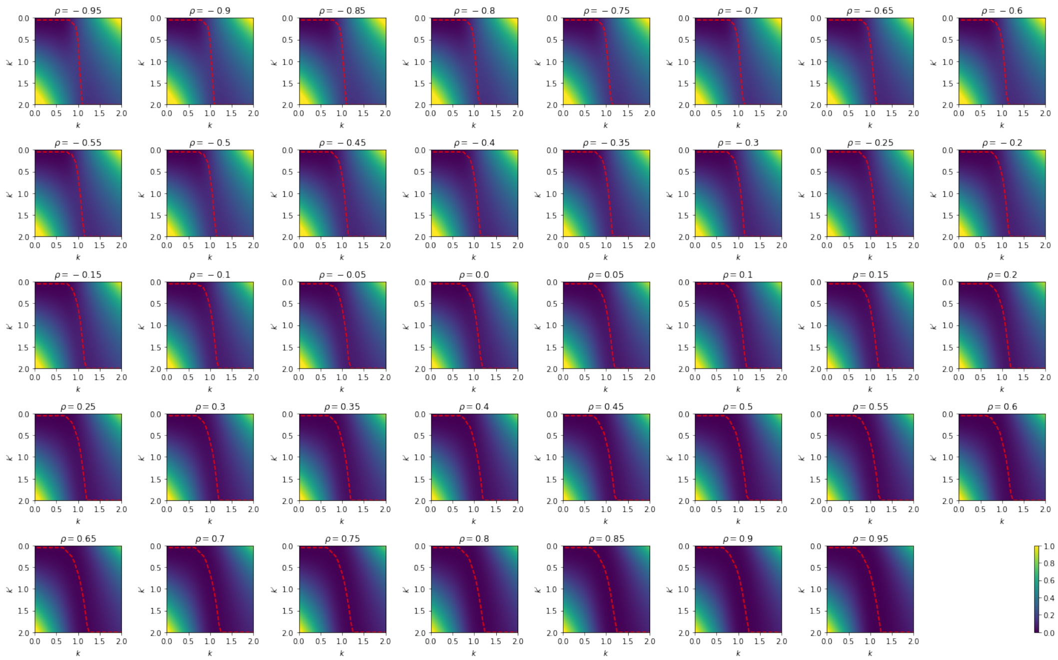
For strike sensitivity, we fix one of the two spot/rate strikes at-the-money and vary the other strike. The variation of contract price exhibits a V-shape, and more negative spot-rate correlation makes the contract worth more.

<table>
<tr><th>Spot-strike k sensitivity</th><th>Rate-strike k' sensitivity</th></tr>
<tr><td>
</td><td>
</td></tr>
</table>

For vol sensitivity, the contract price increases monotonically with spot/rate vol, and decreases monotonically with spot-rate correlation.

<table>
<tr><th>Spot-vol σ sensitivity</th><th>Rate-vol σ^{i+1} sensitivity</th><th>Correlation ρ sensitivity</th></tr>
<tr><td>
</td><td>
</td><td>
</td></tr>
</table>

We visialize the three-dimensional variation of contract price with spot/rate strikes and spot-rate correlation using a series of heatmaps. The red dashed paths travel across local troughs i.e. the most out-of-the-money paths.



3.6 Accuracy of Approximation

The exact pricing formula reads

$$\Pi_0 = P(0,T_{i+1})E^{i+1}\left[\left(\frac{S_i}{S_0}-k\right)\cdot\left(k'-\frac{L_i^{i+1}}{L_0^{i+1}}\right)\right]^+,$$

where under our diffusion model,

$$\frac{L_i^{i+1}}{L_0^{i+1}} = \mathcal{E}(\sigma^{i+1}W_i^1)$$

and

$$\frac{S_i}{S_0} = \frac{1}{P(0,T_i)}\mathcal{E}(\sigma Z_i)$$

with

$$Z_i = \rho W_i^1 + \sqrt{1-\rho^2}W_i^2 - \rho\sigma^{i+1}\int_0^{T_i}\frac{\tau_{i+1}L_u^{i+1}}{1+\tau_{i+1}L_u^{i+1}}du.$$

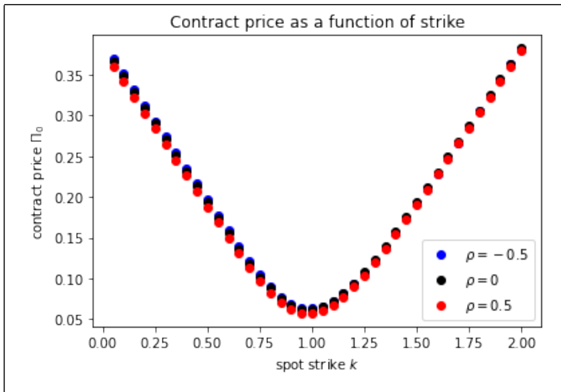
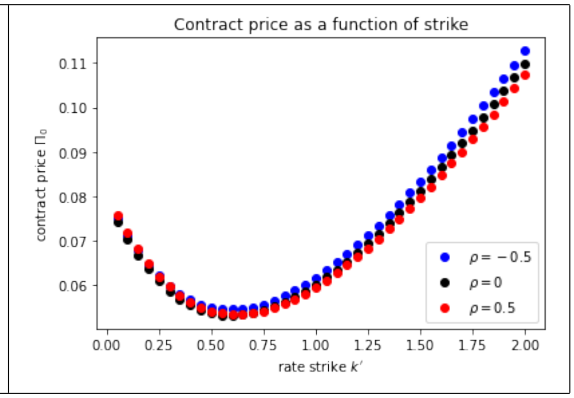
Combining, we have

$$\Pi_0 = \frac{1}{1+\tau_{i+1}L_0^{i+1}}E^{i+1}\left[\left(\mathcal{E}(\sigma Z_i)-kP(0,T_i)\right)\cdot\left(k'-\mathcal{E}(\sigma^{i+1}W_i^1)\right)\right]^+. \tag{13}$$

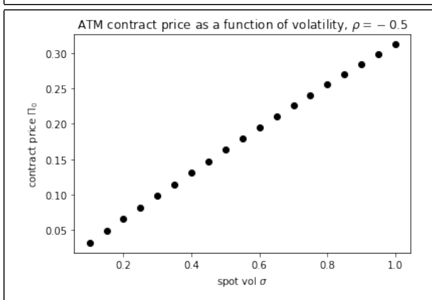
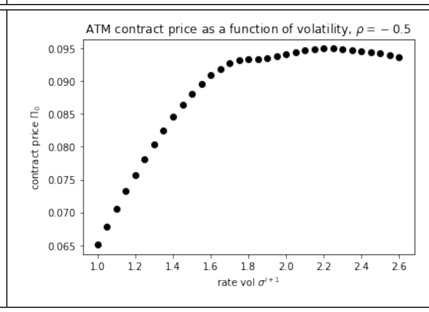
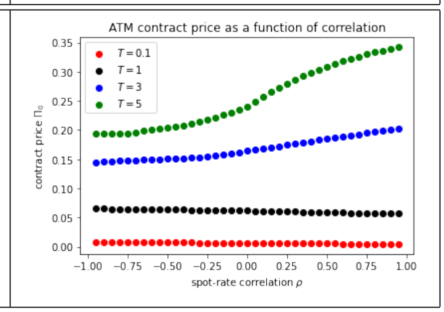
Again, we absorb $P(0,T_i)$ into k and omit the discount factor $1/(1+\tau_{i+1}L_0^{i+1})$, and check the accuracy of our small volatility of rate approximation.

We repeat the same sensitivity analysis as above.

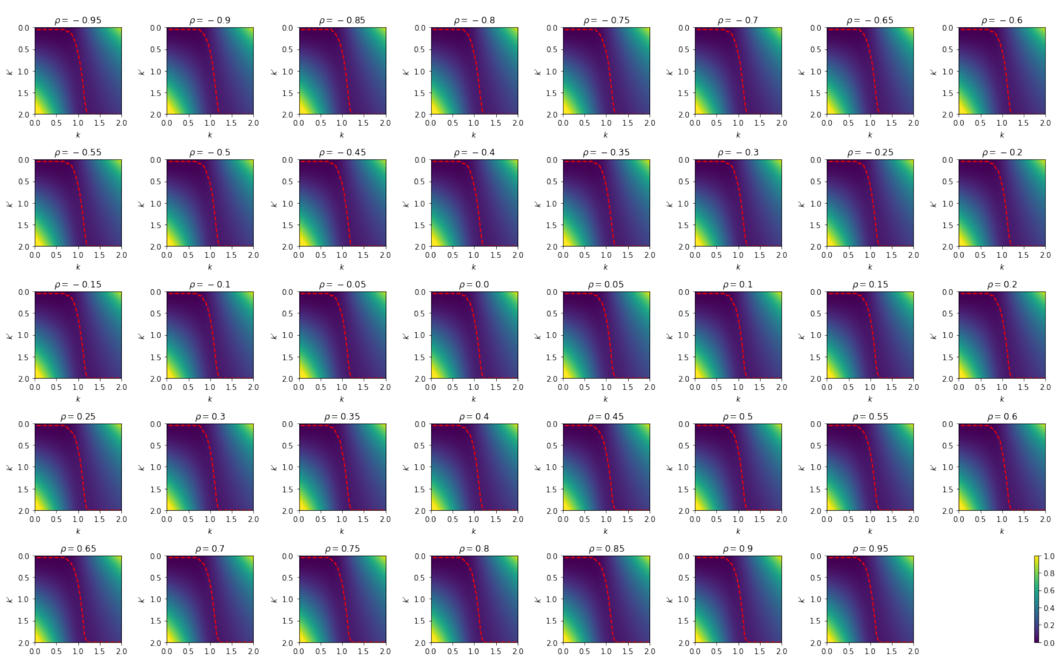
For strike sensitivity, we fix one of the two spot/rate strikes at-the-money and vary the other strike. Still, the variation of contract price exhibits a V-shape, but unlike the small volatility of rate approximation, at-the-money contract price does not respond sensitively to spot-rate correlation.

Spot-strike k sensitivity	Rate-strike k' sensitivity
	

For vol sensitivity, the contract price increases roughly monotonically with spot/rate vol. For small expiry, the price decreases monotonically with spot-rate correlation, but otherwise for large expiry.

Spot-vol σ sensitivity	Rate-vol σ^{i+1} sensitivity	Correlation ρ sensitivity
		

The three-dimensional variation of contract price with spot/rate strikes and spot-rate correlation is again visualized using a series of heatmaps.



We offer intuitive explanations of these observations in section 3.8.

3.7 Spot-Rate Density

If one has the terminal density, any European payoff can be priced. Here, we present the joint density of S_i/S_0 and L_i^{i+1}/L_0^{i+1} to understand the importance of spot-rate correlation ρ. Under our diffusion model, we have

$$\frac{L_i^{i+1}}{L_0^{i+1}} = \mathcal{E}(\sigma^{i+1}W_i^{i+1})$$
$$\frac{S_i}{S_0} = \frac{1}{P(0,T_i)}\mathcal{E}(\sigma Z_i^i).$$

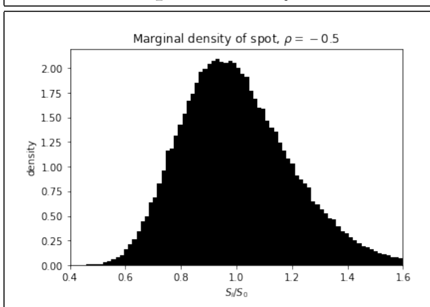
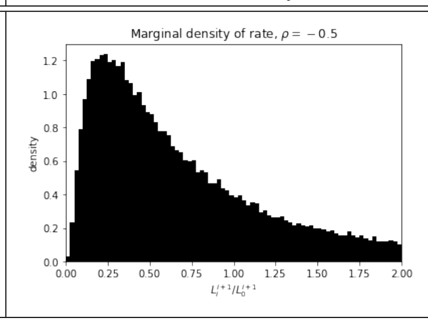
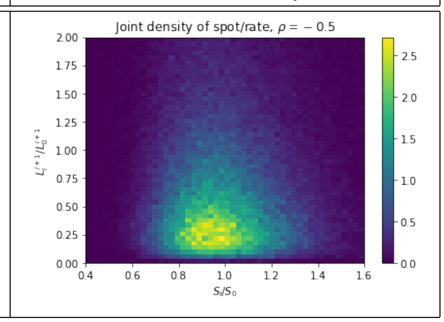
Recall the path-integral of Z_i^i

$$Z_i^i = \rho W_i^{i+1} + \sqrt{1-\rho^2}W_i^{i+1,\perp} - \rho\sigma^{i+1}\int_0^{T_i}\frac{\tau_{i+1}L_u^{i+1}}{1+\tau_{i+1}L_u^{i+1}}du.$$

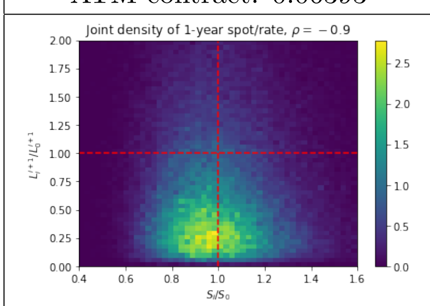
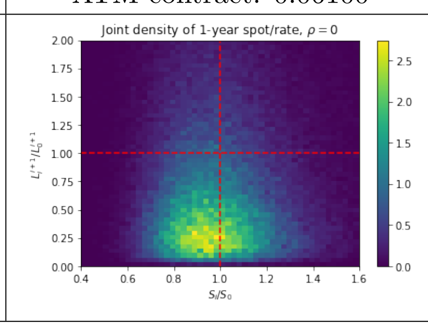
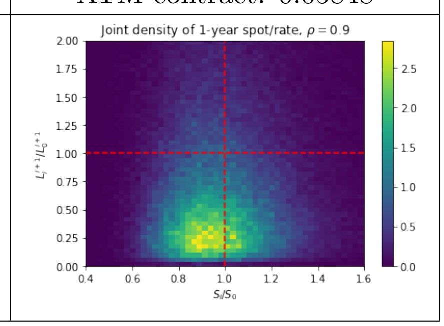
Therefore, to construct the density, we require the path $\{W_t^{i+1}\}_{0\le t\le T_i}$ and an independent normal random variable $W_i^{i+1,\perp}$, repeating for some large number of simulations.

We visualize as the correlation ρ varies, how the density ends up going into/out of the money.

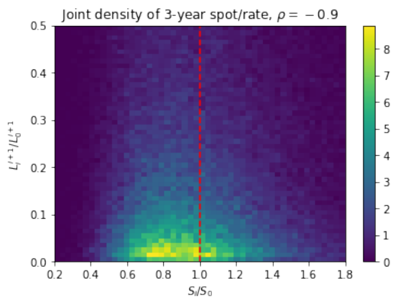
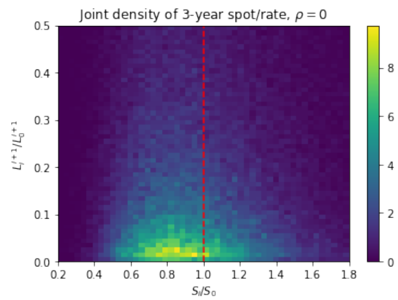
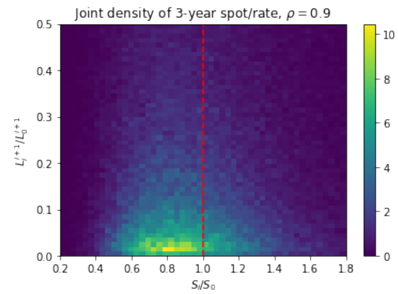
Fixing spot-rate correlation $\rho=-0.5$, we show the spot/rate densities and the joint density. Rate density is precisely log-normal while spot density is no longer log-normal due to compensation term in Z_i^i. Note that the two densities average precisely to unity, as both spot and rate are stochastic exponential.

<table>
<tr><th>Spot density</th><th>Rate density</th><th>Joint density</th></tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</table>

For short expiry $T=1$, as correlation rises, in-the-money (ITM) probability decreases and accordingly, contract price decreases. ITM refers to the top left and bottom right domains, cut by red dashed lines.

<table>
<tr><th>$\rho=-0.9$</th><th>$\rho=0$</th><th>$\rho=0.9$</th></tr>
<tr><td>ITM probability: 0.49678
ATM contract: 0.06393</td><td>ITM probability: 0.48517
ATM contract: 0.06100</td><td>ITM probability: 0.47080
ATM contract: 0.05848</td></tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</table>

For long expiry $T = 3$, as correlation rises, in-the-money probability decreases but the spot/rate market variables can become more extreme. The net effect is that contract values more.

$\rho = -0.9$	$\rho = 0$	$\rho = 0.9$
ITM probability: 0.46688 ATM contract: 0.14737	ITM probability: 0.45869 ATM contract: 0.16751	ITM probability: 0.45169 ATM contract: 0.20632
		

3.8 Discussion

From the sensitivity plots, we observe that

- Our small volatility of rate approximation formula is able to capture some of the qualitative features of contract price against various parameters, and yield a correct order of magnitude of price. As its analytic formula is derivable (but we did not do it as it involves a few pages of tedious, digressing calculations), one may adopt the approximation formula as a control variate in Monte-Carlo.
- For strike sensitivity, regardless of spot-rate correlation ρ, contract price appears roughly as a V-shape plotting against strikes, when the other strike is struck at-the-money. For spot, the contract is most "out-of-the-money" (cheapest) when struck at 1; for rate, most "out-of-the-money" when struck at 0.6. The V-shape may be explained by the following heuristic argument. Consider the contract payoff in the form $(S \cdot L)^+$ where $S, L \in \mathbb{R}$. When one of the two strikes is very small or large (the other struck at-the-money), the sign of one of S, L becomes deterministic, causing the product $S \cdot L$ less likely to end up in a negative domain, so the contract is worth more. Mathematically, consider uncorrelated spot and rate. Assume deterministic spot so that $S_i = 1$ and normally distributed rate so that $1 + Z \equiv L_i^{i+1}/L_0^{i+1} \sim N(1, 1)$. With rate struck at-the-money, contract pays $((1 - k) \cdot Z)^+ = |1 - k| \cdot Z^+$, which is a strict V-shape over k.
- For correlation sensitivity, contract price decreases monotonically with spot-rate correlation when expiry is short, but increases monotonically when expiry is long. For short expiry, when correlation is high, both spot and rate either end up small or large. Thus, S, L carry opposite signs, expiring out-of-the-money, so the contract is worth less. For long expiry, when correlation is high, although probability going into the money gets lower, S, L can be more extreme in value due to correlation, with the net effect pricing the contract up.
- For volatility sensitivity, contract price increases almost monotonically with both spot and rate volatility. Higher volatility implies higher chance to end up in-the-money, thus the contract is worth more time value. But with extremely large rate volatility, drift term in Z_i^i due to measure change is likely to end up more negative. Now, with $S \downarrow L \uparrow$ the contract is more likely to go out-of-the-money thus cheaper.

4 Short-Rate Model

In the diffusion model, we assume the forward LIBOR L_i^{i+1} evolves as a Brownian diffusion. Here, we model the short-rate r_t with Hull-White under the risk-neutral measure $\mathbb{Q}$, and the spot S_t with Heston under the T_i-forward measure $\mathbb{Q}^i$.

Assume the short-rate r_t takes a time-deterministic drift α_t and volatility σ_t s.t. under $\mathbb{Q}$,

$$\begin{aligned} r_t &= \alpha_t + x_t \\ dx_t &= -a x_t dt + \sigma_t dW_t, \quad x_0 = 0. \end{aligned} \tag{14}$$

Solving,

$$x_T = x_t e^{-a(T-t)} + \int_t^T e^{-a(T-u)} \sigma_u dW_u.$$

Integrating,

$$\int_t^T x_u du = B(t,T) x_t + \int_t^T B(u,T) \sigma_u dZ_u$$

where $B(t,T) \equiv \left(1 - e^{-a(T-t)}\right)/a$.

By virtue of Itô integral, conditional expectation reads

$$E_t e^{-\int_t^T x_u du} = \exp\left(-B(t,T)x_t + \int_t^T B(u,T)^2 \sigma_u^2 du\right) \equiv A(t,T) e^{-B(t,T)x_t}.$$

A bond expiring T has price at time t

$$P(t,T) = A(t,T) e^{-\int_t^T \alpha_u du} e^{-B(t,T)x_t},$$

which can be written as

$$P(t,T) = \frac{A(0,t)A(t,T)}{A(0,T)} \frac{P(0,T)}{P(0,t)} e^{-B(t,T)x_t}.$$

The LIBOR forward rate expressed in terms of bond prices reads

$$L_t^{i+1} = \frac{1}{\tau_{i+1}} \left(\frac{P(t,T_i)}{P(t,T_{i+1})} - 1 \right),$$

where we have

$$\frac{P(t,T_i)}{P(t,T_{i+1})} = \frac{P(0,T_i)}{P(0,T_{i+1})} \frac{A(0,T_{i+1})}{A(0,T_i)A(T_i,T_{i+1})} e^{(B(t,T_{i+1})-B(t,T_i))x_t}.$$

Under $\mathbb{Q}^{i+1}$, L_t^{i+1} deflated by $P(t,T_{i+1})$ must be a martingale, so

$$dL_t^{i+1} = \left(L_t^{i+1} + \frac{1}{\tau_{i+1}}\right)\left(B(t,T_{i+1}) - B(t,T_i)\right) \cdot \sigma_t dW_t^{i+1}. \tag{15}$$

More rigorously, one will need to model the OIS rate and LIBOR-OIS basis, but this introduces many more moving parts quickly complicating the model. Here, for interpretability, we assume that the short-rate model directly implies the LIBOR forward rate dynamics.

The T_i-forward spot, quanto-transformed according to equation (10), is modeled as a $\mathbb{Q}^i$-martingale. Assume a Brownian diffusion with stochastic variance v_t

$$\begin{aligned} d\left(\frac{S_t}{P(t,T_i)}\right) &= \left(\frac{S_t}{P(t,T_i)}\right)\sqrt{v_t}dZ_t^i \\ dv_t &= \lambda(\bar{v}-v_t)dt + \eta\sqrt{v_t}dX_t^i \end{aligned} \tag{16}$$

with

$$dZ_t^i dX_t^i = \rho^v dt.$$

Assume spot-rate correlation ρ. Then, we have from the change of measure $d\mathbb{Q}^i/d\mathbb{Q}^{i+1}$ following equation (6) that

$$dZ_t^i = \rho dW_t^{i+1} + \sqrt{1-\rho^2}dW_t^{i+1,\perp} - (B(t,T_{i+1})-B(t,T_i))\cdot\rho\sigma_t dt, \tag{17}$$

or integrating up to T_i,

$$Z_i^i = \rho W_i^{i+1} + \sqrt{1-\rho^2}W_i^{i+1,\perp} - \rho\int_0^{T_i}(B(u,T_{i+1})-B(u,T_i))\cdot\sigma_u du,$$

where the compensation term is deterministic.

For the Hull-White rate dynamics, we calibrate a, α_t, σ_t to bond prices and rate options over an array of expiries as we require the term-structures α_t, σ_t; for the Heston spot dynamics, we calibrate $\lambda, \bar{v}, \eta, \rho^v$ to vanilla options expiring T_i. Below we sketch the calibration procedures.

4.1 Calibration of Heston

Consider a call option $C(K,T)$ on spot, modeled under the T-forward measure so effectively interest rate and dividend yield are zero. Define log-strike $k=\log(K/S_0)$ and log-spot $Y_T=\log(S_T/S_0)$. Lewis equation states that under characteristic function $\phi_T(u)\equiv Ee^{iuY_T}$, call price

$$C(K,T) = S_0 - \frac{\sqrt{S_0K}}{2\pi}\int_{-\infty}^{\infty}\frac{du}{u^2+\frac{1}{4}}e^{-iuk}\phi_T\left(u-\frac{i}{2}\right),$$

where the latter Fourier-integral can be computed with fast-Fourier transform (FFT), detailed in the seminal work of Carr and Madan (1999).

Heston model admits the characteristic function $\phi_T(u)=\exp\left(\mathcal{C}_T(u)\bar{\boldsymbol{v}}+\mathcal{D}_T(u)\boldsymbol{v}\right)$ where

$$\begin{aligned} \mathcal{C}_T(u) &= \lambda\left(r_-T - \frac{2}{\eta^2}\log\left(\frac{1-ge^{-dT}}{1-g}\right)\right) \\ \mathcal{D}_T(u) &= r_-\frac{1-e^{-dT}}{1-ge^{-dT}} \end{aligned}$$

in which

$$\begin{aligned} \alpha &= -\frac{u^2}{2}-\frac{iu}{2} \\ \beta &= \boldsymbol{\lambda}-\boldsymbol{\rho^v\eta}iu \\ \gamma &= \frac{\eta^2}{2} \\ d &= \sqrt{\beta^2-4\alpha\gamma} \\ r_\pm &= \frac{\beta\pm d}{2\gamma} \\ g &= \frac{r_+}{r_-} \end{aligned}$$

following Gatheral's Volatility Surface (2004). Heston model parameters are highlighted in bold.

The Heston-implied price/volatility surface is calibrated to the market-observed price/volatility surface, by means of least square appropriately weighted, say using bid-ask spreads or vegas. Strictly speaking, we have to calibrate to domestic options on the quanto-transformed spot (as $\mathbb{Q}$ is domestic) but no such instrument exists. Therefore as a proxy, we calibrate to the foreign options on foreign spot; in other words, Nikkei-225 options dominated in JPY. One may either calibrate to the whole surface or specifically to the T_i-slice – our code does the former. We do not describe the calibration technicalities here because our main focus is pricing.

4.2 Calibration of Hull-White

A bond expiring T has initial price

$$P(0,T) \equiv e^{-y_T T} = A(0,T)e^{-\int_0^T \alpha_u du}$$

where y_T is the bond yield. Then,

$$\begin{aligned} y_T T &= -\log A(0,T) + \int_0^T \alpha_u du \\ \partial_T(y_T T) &= -\partial_T \log A(0,T) + \alpha_T. \end{aligned}$$

Now, recall

$$\begin{aligned} A(t,T) &\equiv \exp\left(\int_t^T B(u,T)^2 \sigma_u^2 du\right) \\ B(t,T) &\equiv \frac{1-e^{-a(T-t)}}{a}, \end{aligned}$$

so

$$\partial_T(y_T T) = \alpha_T - \int_0^T 2B(u,T)\partial_T B(u,T)\sigma_u^2 du.$$

Assume a flat rate volatility term structure $\sigma_t = \sigma$. We have

$$\partial_T(y_T T) = \alpha_T - \sigma^2 B(0,T)^2.$$

In this flat-vol Hull-White model, we are to calibrate (a, σ, α_t). To match the degrees of freedom, we need (1) the entire yield curve, (2) two rate options, say at-the-money caplets expiring T_i and T_{i+1}. In theory, if we relax the flat-vol assumption, Hull-White may be matched to all at-the-money caplets/floorlets. But here for simplicity, we specialize to flat-vol where analytic formulas may be obtained.

We derived that the shifted LIBOR forward rate $\tilde{L}_t^{i+1} \equiv L_t^{i+1} + 1/\tau_{i+1}$ is a martingale

$$d\tilde{L}_t^{i+1} = \tilde{L}_t^{i+1}(B(t,T_{i+1}) - B(t,T_i)) \cdot \sigma dW_t^{i+1}.$$

As derived in calibration section under diffusion model, a caplet prices according to

$$C_0^L(R,T_i) = \tau_{i+1}P(0,T_{i+1})E^{i+1}(L_i^{i+1} - R)^+ \equiv \tau_{i+1}P(0,T_{i+1})E^{i+1}(\tilde{L}_i^{i+1} - \tilde{R})^+$$

where $\tilde{R} \equiv R + 1/\tau_{i+1}$. Quoting in Black volatility σ^{i+1} (under Black diffusion), at-the-money we have

$$C_0^L(L_0^{i+1}, T_i) = \tau_{i+1}P(0,T_{i+1})\tilde{L}_0^{i+1}(2N(d_1) - 1)$$

where $d_1 = \sigma^{i+1}\sqrt{T_i}/2$. For small d_1, $2N(d_1) - 1 \approx \sqrt{2/\pi}d_1$, so

$$C_0^L(L_0^{i+1}, T_i) \approx \frac{\tau_{i+1}P(0, T_{i+1})}{\sqrt{2\pi}}\tilde{L}_0^{i+1}\sigma^{i+1}\sqrt{T_i}.$$

Suppose that we observe Black volatility $\hat{\sigma}^{i+1}$ in the market, then by quoting convention

$$\frac{\tau_{i+1}P(0, T_{i+1})}{\sqrt{2\pi}}\tilde{L}_0^{i+1}\sigma^{i+1}\sqrt{T_i} \equiv \frac{\tau_{i+1}P(0, T_{i+1})}{\sqrt{2\pi}}L_0^{i+1}\hat{\sigma}^{i+1}\sqrt{T_i}$$

thus

$$\sigma^{i+1} = \frac{L_0^{i+1}}{\tilde{L}_0^{i+1}}\hat{\sigma}^{i+1}.$$

Under our flat-vol Hull-White, $\tilde{L}_t^{i+1}$ diffuses with a time-deterministic volatility, so we have for the total implied variance

$$w_i(a, \sigma) \equiv (\sigma^{i+1})^2 T_i = \int_0^{T_i} (B(u, T_{i+1}) - B(u, T_i))^2 \sigma^2 du = \frac{\sigma^2}{2}B(0, \tau_{i+1})^2 B(0, 2T_i)$$

after simplification. Note that a-dependence hides inside $B(0, \cdot)$. But i is arbitrary so we have a system of equations for $i, i+1$:

$$\begin{cases} w_i(a,\sigma) \equiv \left(\frac{L_0^{i+1}}{\tilde{L}_0^{i+1}}\hat{\sigma}^{i+1}\right)^2 T_i = \frac{\sigma^2}{2}B(0,\tau_{i+1})^2 B(0, 2T_i) \\ w_{i+1}(a,\sigma) \equiv \left(\frac{L_0^{i+2}}{\tilde{L}_0^{i+2}}\hat{\sigma}^{i+2}\right)^2 T_{i+1} = \frac{\sigma^2}{2}B(0,\tau_{i+2})^2 B(0, 2T_{i+1}) \end{cases}$$

for the two degrees of freedom a, σ, with w_i, w_{i+1} read off market. The system has to be numerically solved.

After calibrating a, σ, we calibrate α_T via

$$\alpha_T = \partial_T(y_T T) + \sigma^2 B(0, T)^2.$$

4.3 Monte-Carlo Simulation

To simulate, we evolve LIBOR forward rate L_t^{i+1}, or equivalently $\tilde{L}_t^{i+1} \equiv L_t^{i+1} + 1/\tau_{i+1}$, and T_i-forward spot S_t according to equations (15) and (16)

$$\begin{aligned} d\tilde{L}_t^{i+1} &= (B(t, T_{i+1}) - B(t, T_i)) \cdot \sigma_t \tilde{L}_t^{i+1} dW_t^{i+1} \\ dS_t &= \sqrt{v_t} S_t dZ_t^i \\ dv_t &= \lambda(\bar{v} - v_t)dt + \eta\sqrt{v_t}dX_t^i \end{aligned}$$

where correlations

$$dZ_t^i = \rho dW_t^{i+1} + \rho^v dX_t^i + \sqrt{1 - \rho^2 - (\rho^v)^2}dW_t^{i+1,\perp} - (B(t, T_{i+1}) - B(t, T_i)) \cdot \rho\sigma_t dt$$

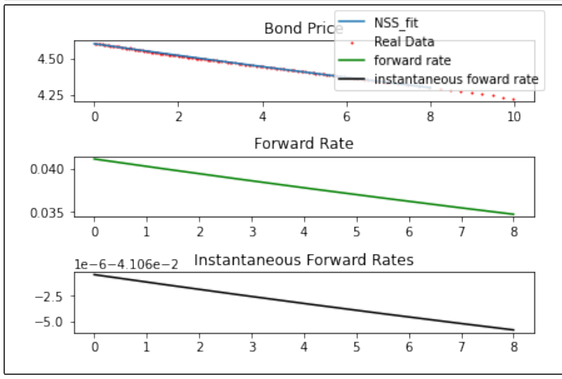
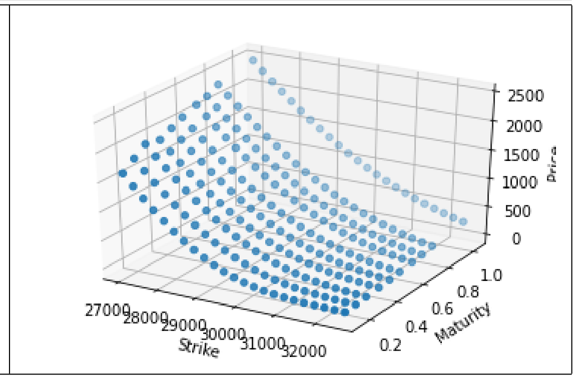
and dW_t^{i+1}, dX_t^i are assumed uncorrelated, i.e. volatility of Nikkei-225 does not correlate with the LIBOR foward rate.

Discretization is extremely straightforward, except for variance we have to adopt some truncation scheme so that it remains positive.

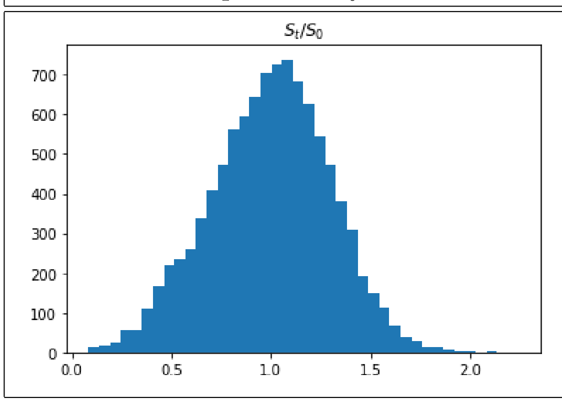
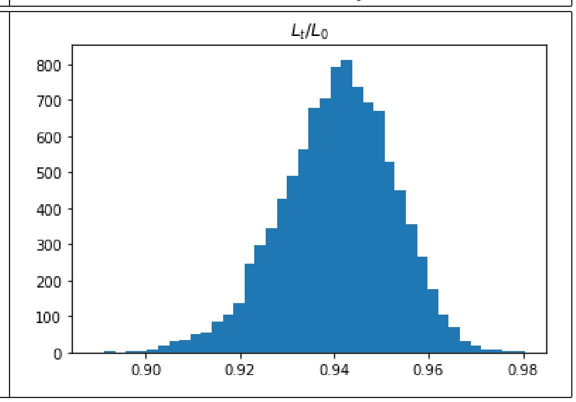
The calibration and pricing routine are provided in section 6.2.

4.4 Parameter Sensitivity

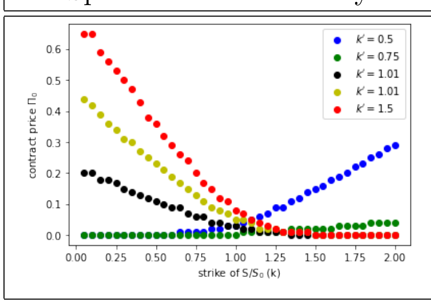
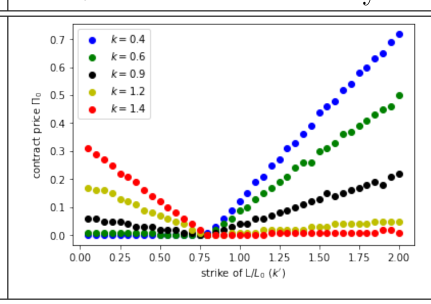
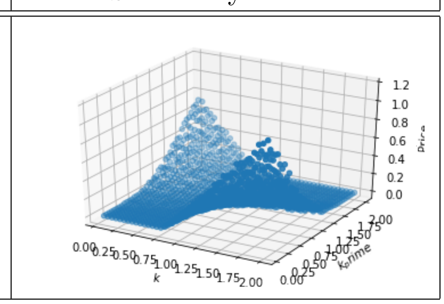
We study in particular the case spot-rate correlation $\rho = 0$. Calibration to US treasury yields using the Nelson-Siegel-Svensson parametrization and Nikkei-225 calls using least square optimization yield the following graphs.

Yield calibration	Option calibration
	

Simulation under the market-calibrated parameters gives the following spot/rate densities at expiry T_i. Notice that they are more or less Gaussian-shaped, contrary to the diffusion model where rate density is strictly log-normal.

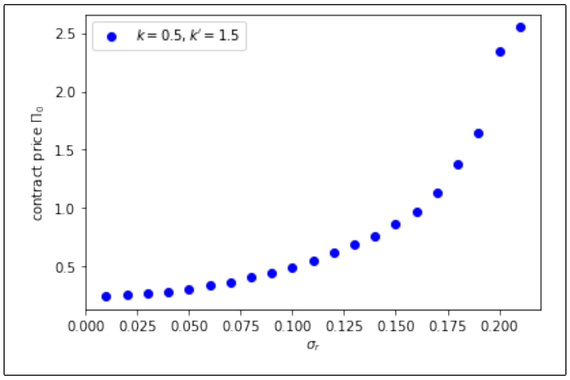
Spot density	Rate density
	

Sensitivities wrt. strikes k, k' are shown below. The sensitivity surface is consistent with the diffusion model – see for example the heatmap for $\rho = 0$.

Spot-strike k sensitivity	Rate-strike k' sensitivity	Sensitivity surface
		

Finally, we show the sensitivity wrt. rate volatility, which monotonically increases over the domain $[0, 0.2]$.

Rate-vol sensitivity



4.5 Discussion

In this section, real market price is used in calibration process. Although the price will not be exactly same, some similarities exist in both models.

- For strike sensitivity, the result of in this section is consistent with Diffusion model's: the contract is most expensive when k and k' are far away, and the closer the two strikes are, the cheaper the contract price is.
- For volatility sensitivity, contract price increases monotonically with volatility, which also matches the result in Diffusion model.
- When fix k or k', the curve of the contract price v.s. k' or k is similar to call, put option or a straight line around zero. For example, fix strike_prime at around simulation mean of $\frac{L}{L_0}$, then the latter part, $(k' - \frac{L}{L_0})$ will be around zero. Hence, the payoff curve is a line around zero. When k' is fixed less than simulation mean, $(k' - \frac{L}{L_0})$ will be negative. Thus, if k is really small, the whole thing, $(\frac{S}{S_0} - k)(k' - \frac{L}{L_0})$, will become negative and the price of the contract will be zero. However, when k is large enough, the first part of payoff function will be negative, so $(\frac{S}{S_0} - k)(k' - \frac{L}{L_0})$ will be positive. Then, the price curve will look like a call option curve.

5 Conclusion

We studied the pricing, calibration and sensitivities of a quanto spot-rate product option under diffusion and short-rate model. In the modeling, the option is priced under the T_{i+1}-forward measure, under which LIBOR forward rate L_t^{i+1} is a $\mathbb{Q}^{i+1}$-martingale, and T_i-forward spot $S_t/P(t, T_i)$ is modeled as a $\mathbb{Q}^i$-martingale. From here, one may motivate more sophisticated models of their choice. Future study may investigate the model-dependence of the contract, e.g. comparing the two models, say under mild parameters, does price significantly vary?

6 Codes

Here we present the pricing routines of the two models proposed, coded in python. For code snippets and data to replicate the plots in this report, see the python notebook attached. The data are obtained from Bloomberg, snapshot as of late November, 2022, included in the `data` folder.

6.1 Diffusion Model

```
import numpy as np
```

```python
import matplotlib.pyplot as plt
from scipy.special import ndtr
from tqdm import tqdm

def stoc_exp(sig, T, Z=None, n=1):
    '''
    @ n: vector size
    @ sig: volatility
    @ T: evaluation time
    stochastic exponential of sig*Bt, a martingale
    '''
    if Z is None:
        return np.exp(-sig**2*T/2+sig*np.sqrt(T)*np.random.normal(size=n))
    else:
        return np.exp(-sig**2*T/2+sig*np.sqrt(T)*Z)

def call_bs(F, K, sig, T, D=1):
    '''
    @ D: discount factor
    @ F: fwd price
    @ K: strike
    @ sig: volatility
    @ T: expiry
    Black-76 formula for call price
    '''
    d1 = np.log(F/K)/(sig*np.sqrt(T))+(sig*np.sqrt(T))/2
    d2 = d1-sig*np.sqrt(T)
    return D*(F*ndtr(d1)-K*ndtr(d2))

def put_bs(F, K, sig, T, D=1):
    '''
    @ D: discount factor
    @ F: fwd price
    @ K: strike
    @ sig: volatility
    @ T: expiry
    Black-76 formula for put price
    '''
    d1 = np.log(F/K)/(sig*np.sqrt(T))+(sig*np.sqrt(T))/2
    d2 = d1-sig*np.sqrt(T)
    return D*(K*ndtr(-d2)-F*ndtr(-d1))

_pricer_diff_vor_cache = dict()
_pricer_diff_vor_cacheall = dict()

def pricer_diff_vor(k1, k2, rho, sig1, sig2, T, tau=0.25, L0=0, P0=1, N=1e4, cache=
       False, cacheAll=False):
    '''
    @ k1: spot strike
    @ k2: rate strike
    @ rho: spot-rate correlation
    @ sig1: spot vol
    @ sig2: rate vol
    @ T: expiry
    @ tau: arrear interval
    @ L0: LIBOR fwd rate from T to T+tau
    @ P0: bond price/discount factor expiring T
    @ N: number of simulations
    @ cache: use cached random variables or not
    @ cacheAll: use all cached variables or not (for pricing over strike grid)
    contract pricer under diffusion model using small vol-of-rate approx
```

```
    '''
    N = int(N)
    if cacheAll:
        global _pricer_diff_vor_cacheall
        if not _pricer_diff_vor_cacheall or \
        (rho,sig1,sig2,T,N) != _pricer_diff_vor_cacheall['(rho,sig1,sig2,T,N)']:
            Z = np.random.normal(size=N)
            S = stoc_exp(rho*sig1,T,Z)
            L = stoc_exp(sig2,T,Z)
            _pricer_diff_vor_cacheall = {
                '(rho,sig1,sig2,T,N)': (rho,sig1,sig2,T,N),
                'Z': Z,
                'S': S,
                'L': L,
            }
        Z = _pricer_diff_vor_cacheall['Z']
        S = _pricer_diff_vor_cacheall['S']
        L = _pricer_diff_vor_cacheall['L']
    else:
        if cache:
            global _pricer_diff_vor_cache
            if not _pricer_diff_vor_cache or \
            N != _pricer_diff_vor_cache['N']:
                Z = np.random.normal(size=N)
                _pricer_diff_vor_cache = {
                    'N': N,
                    'Z': Z,
                }
            Z = _pricer_diff_vor_cache['Z']
        else:
            Z = np.random.normal(size=N)
        S = stoc_exp(rho*sig1,T,Z)
        L = stoc_exp(sig2,T,Z)
    Pi = np.mean(call_bs(S,k1*P0,np.sqrt(1-rho**2)*sig1,T)*np.maximum(k2-L,0)+\
                 put_bs(S,k1*P0,np.sqrt(1-rho**2)*sig1,T)*np.maximum(L-k2,0))/(1+tau*L0
    )
    return Pi

pricer_diff_vor_vec = np.vectorize(pricer_diff_vor)

_pricer_diff_exact_cache = dict()
_pricer_diff_exact_cacheall = dict()

def pricer_diff_exact(k1, k2, rho, sig1, sig2, T, tau=0.25, L0=0, P0=1, N=1e4, dt
    =0.004, disc=True, cache=False, cacheAll=False):
    '''
    @ k1: spot strike
    @ k2: rate strike
    @ rho: spot-rate correlation
    @ sig1: spot vol
    @ sig2: rate vol
    @ T: expiry
    @ tau: arrear interval
    @ L0: LIBOR fwd rate from T to T+tau
    @ P0: bond price expiring T
    @ N: number of simulations
    @ dt: time interval of simulation paths
    @ disc: include discount factor or not
    @ cache: use cached random variables or not
    @ cacheAll: use all cached variables or not (for pricing over strike grid)
    contract pricer under diffusion model using exact computation
```

```
    '''
    N = int(N)
    M = int(T/dt)
    if cacheAll:
        global _pricer_diff_exact_cacheall
        if not _pricer_diff_exact_cacheall or \
        (rho,sig1,sig2,T,tau,M,N) != _pricer_diff_exact_cacheall['(rho,sig1,sig2,T,tau,
    M,N)']:
            Z = np.random.normal(size=(M,N))
            W = np.random.normal(size=N)
            Lt = L0*np.cumprod(stoc_exp(sig2,dt,Z),axis=0)
            D = np.exp(-rho*sig2*np.sum(Lt/(1+tau*Lt),axis=0)*dt)
            S = D*stoc_exp(sig1,T,rho*Z[-1]+np.sqrt(1-rho**2)*W)
            L = Lt[-1]/L0
            _pricer_diff_exact_cacheall = {
                '(rho,sig1,sig2,T,tau,M,N)': (rho,sig1,sig2,T,tau,M,N),
                'Z': Z,
                'W': W,
                'Lt': Lt,
                'D': D,
                'S': S,
                'L': L,
            }
        Z = _pricer_diff_exact_cacheall['Z']
        W = _pricer_diff_exact_cacheall['W']
        Lt = _pricer_diff_exact_cacheall['Lt']
        D = _pricer_diff_exact_cacheall['D']
        S = _pricer_diff_exact_cacheall['S']
        L = _pricer_diff_exact_cacheall['L']
    else:
        if cache:
            global _pricer_diff_exact_cache
            if not _pricer_diff_exact_cache or \
            (M,N) != _pricer_diff_exact_cache['(M,N)']:
                Z = np.random.normal(size=(M,N))
                W = np.random.normal(size=N)
                _pricer_diff_exact_cache = {
                    '(M,N)': (M,N),
                    'Z': Z,
                    'W': W,
                }
            Z = _pricer_diff_exact_cache['Z']
            W = _pricer_diff_exact_cache['W']
        else:
            Z = np.random.normal(size=(M,N))
            W = np.random.normal(size=N)
        Lt = L0*np.cumprod(stoc_exp(sig2,dt,Z),axis=0)
        D = np.exp(-rho*sig2*np.sum(Lt/(1+tau*Lt),axis=0)*dt)
        S = D*stoc_exp(sig1,T,rho*Z[-1]+np.sqrt(1-rho**2)*W)
        L = Lt[-1]/L0
    Pi = np.mean(np.maximum((S-k1)*(k2-L),0))
    if disc: Pi /= 1+tau*L0
    return Pi

pricer_diff_exact_vec = np.vectorize(pricer_diff_exact)
```

6.2 Short-Rate Model

6.2.1 Calibration of Heston

We reference [1] for the calibration.

```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from scipy.integrate import quad
from scipy.optimize import minimize
from nelson_siegel_svensson import NelsonSiegelSvenssonCurve
from nelson_siegel_svensson.calibrate import calibrate_nss_ols

def charifunc(phi, rho, lbd, t, r, S0, v0, k, theta, sigma):

    a = k*theta
    b = k+lbd
    rpi = rho*sigma*phi*1j
    c = np.sqrt((rho*sigma*phi*1j - b)**2 + (phi*1j+phi**2)*sigma**2)
    d = (b-rpi+c)/(b-rpi-c)

    return np.exp(r*phi*1j*t) * S0**(phi*1j) * ((1-d*np.exp(c*t))/(1-d))**(-2*a/sigma**2) * np.exp(a*t*(b-rpi+c)/sigma**2 + v0*(b-rpi+c)*((1-np.exp(c*t))/(1-d*np.exp(c*t)))/sigma**2)

def heston_price(S0, K, v, k, theta, sigma, rho, lbd, t, r):
    args = (rho, lbd, t, r, S0, v, k, theta, sigma)
    price = 0
    up_limit_ = 100
    n = 10000
    dt = up_limit_/n

    for i in range(1,n):
        phi = dt * (2*i + 1)/2
        numerator = np.exp(r*t)*charifunc(phi-1j,*args) - K * charifunc(phi,*args)
        denominator = 1j*phi*K**(1j*phi)
        price += dt * numerator/denominator

    return np.real((S0 - K*np.exp(-r*t))/2 + price/np.pi)
```

```python
## Get the parameteric yield curve from Nelson Siegel Svensson model
## We use JPY Libor rate to approximate risk-free interest rate
yield_maturities = np.array([1/12, 3/12, 6/12])
yields = np.array([-0.06005,-0.04713,0.03097]).astype(float)/100
curve_fit, status = calibrate_nss_ols(yield_maturities,yields)

y = curve_fit
t = np.linspace(0, 10, 10000)

## Plot the yield curve
plt.plot(t, y(t),label = 'NSS_fit')
plt.legend()
plt.show()
```

```python
from scipy.linalg.special_matrices import dft
from locale import D_FMT
## Get the NIKKEI 225 call option data from Bloomberg
## Read the data from the drive
from google.colab import drive
import pandas as pd
drive.mount('/content/gdrive')
df = pd.read_csv('gdrive/MyDrive/NKY.csv')
df['rate'] = df['maturity'].apply(curve_fit)
```

```python
## Get the settlement call price for Nikkei option
## Get the settlement spot price of Nikkei 225 as spot price
```

```python
S0 = df['spot'].to_numpy('float')
r = df['rate'].to_numpy('float')
K = df['Strike'].to_numpy('float')
t = df['maturity'].to_numpy('float')
MP = df['Price'].to_numpy('float')

## Implement min_squared_error to get the parameters
cali_vars = {"v": {"x0": 0.05, "bnd": [1e-3,0.5]},
           "k": {"x0": 1.5, "bnd": [1e-3,2]},
           "theta": {"x0": 0.05, "bnd": [1e-3,0.5]},
           "sigma": {"x0": 0.5, "bnd": [1e-2,1]},
           "rho": {"x0": -0.5, "bnd": [-1,1]},
           "lbd": {"x0": 0.5, "bnd": [-1,1]}}

x0 = [cali_var["x0"] for key, cali_var in cali_vars.items()]
bnds = [cali_var["bnd"] for key, cali_var in cali_vars.items()]

def Squared_err(x):
    v, k, theta, sigma, rho, lbd = [cali_var for cali_var in x]
    err = np.sum((MP-heston_price(S0, K, v, k, theta, sigma, rho, lbd, t, r))**2 /len(P
    ))
    return err

result = minimize(Squared_err, x0, tol = 1e-4, method='SLSQP', options={'maxiter': 1e4
    }, bounds=bnds)
v, k, theta, sigma, rho, lbd = [cali_var for cali_var in result.x]
```

```python
price = heston_price(S0, K, v, k, theta, sigma, rho, lbd, t, r)
df['mod'] = price

from mpl_toolkits import mplot3d
fig = plt.figure()
ax = plt.axes(projection='3d')

# Data for a three-dimensional line
zdata = df['mod']
xdata = df['Strike']
ydata = df['maturity']
ax.scatter3D(xdata, ydata, zdata, 'green')
ax.set_xlabel('Strike')
ax.set_ylabel('Maturity')
ax.set_zlabel('Price')
plt.show()
```

6.2.2 Calibration of Hull-White

We reference [2] for the calibration.

```python
## We use United States Treasury Strip Coupon Price (Zero Coupon Bond) to calibrate
    HUll White Model
## Data is pulled from Bloomberg
from scipy.linalg.special_matrices import dft
from locale import D_FMT
## Read the data from the drive
from google.colab import drive
import pandas as pd
drive.mount('/content/gdrive')
bp = pd.read_excel('gdrive/MyDrive/US_Zero.xlsx')

df = pd.DataFrame(index = bp.index)

```

```
df['price'] = bp['Ask Price']
df['Maturity'] = (pd.to_datetime(bp['Maturity']) - pd.to_datetime('2022-11-23')).dt.
    days / 365
df = df[df['Maturity'] < 10]
T = df['Maturity'].to_numpy('float')
P = df['price'].to_numpy('float') / 100
```

```
from scipy.optimize import minimize
import matplotlib.pyplot as plt
from scipy.misc import derivative
import matplotlib.pyplot as plt
from scipy.optimize import curve_fit
from scipy.integrate import quad
import numpy as np
## Fit the bond price curve
def func(x, a, b, c):
    return a * np.exp(-b * x) + c

popt, pcov = curve_fit(func, df['Maturity'], np.log(df['price']))

#curve_fit, status = calibrate_nss_ols(np.array(df['Maturity']),np.array(np.log(df['
    price'])))
def function(x):
    y = func(x,*popt)
    return y

def deriv(x):
    return derivative(function, x)

y = curve_fit
t = np.linspace(0, 8, 10000)

figure, axis = plt.subplots(3, 1)
## For P(0,T)
axis[0].plot(t, func(t, *popt),label = 'NSS_fit')
axis[0].scatter(df['Maturity'],np.log(df['price']), c='red',s = 1,label = 'Real Data')
axis[0].set_title("Bond Price")

# For forward rate curve
axis[1].plot(t, -deriv(t), color='green', label='forward rate')
axis[1].set_title("Forward Rate")


# For market-implied instantaneous forward rates
axis[2].plot(t, deriv(-deriv(t)), color='black', label='instantaneous foward rate')
axis[2].set_title("Instantaneous Forward Rates")

figure.tight_layout()
figure.legend()
plt.show()
```

```
from scipy.integrate import quad

def theta_cal(T,a,sigma):
    return deriv(-deriv(T)) + a * (-deriv(T)) + sigma**2 / (2*a) * (1-np.exp(-2*a*T))

def integrand(x, T,a, sigma):
    B = (1- np.exp(-a*x))/a
    theta = deriv(-deriv(x)) + a * (-deriv(x)) + sigma**2 / (2*a) * (1-np.exp(-2*a*x))

    return theta * B

```

```python
def bond_price(T,a,sigma,r):
    B = (1- np.exp(-a*T))/a
    I, error = quad(integrand, 0, T, args=(T,a,sigma))
    A = (np.exp(-I) - (sigma**2 / (2*(a**2)))) * (B - T) - (sigma**2)/(4*a) * B**2
    price = np.exp(A - B*r)
    return price

params = params = {"a": {"x0": 2, "bnd": [0.001,5]},
          "sigma": {"x0": 0.02, "bnd": [0.001,1]},
          "r_0": {"x0": 0.02, "bnd": [0.001,1]}}

x0 = [param["x0"] for key, param in params.items()]
bnds = [param["bnd"] for key, param in params.items()]

def Squared_err(x):
    a, sigma, r_0= [param for param in x]
    err = 0
    for i in range(len(P)):
        err += ((P[i]-bond_price(T[i],a, sigma, r_0))**2 /len(P))

    return err

result = minimize(Squared_err, x0, tol = 1e-4, method='SLSQP', options={'maxiter': 1e3
    }, bounds=bnds)
a, sigma, r_0 = [param for param in result.x]
print(result)
```

```python
bp_lst = []
for i in range(len(df)):
    bp_lst.append(bond_price(T[i],a, sigma,r_0) * 100)
df['mod'] = bp_lst
plt.scatter(df['Maturity'],df['price'],color = 'black',label = 'Real Data')
plt.scatter(df['Maturity'],df['mod'],color = 'red',label = 'Model Data')
plt.legend()
plt.show()
```

6.2.3 Monte-Carlo Pricing

Under Hull-White model,

$$dr_t = [\theta_t - \alpha r_t]dt + \sigma dW_t.$$

We then simulate $P(T, T+\Delta) = E[e^{-\int_T^{T+\Delta} r_s ds}], p(0,T), P(0,T+\Delta)$ to calculate $L(T,T,T+\Delta) = \frac{1}{\Delta}\frac{1-P(T,T+\Delta)}{P(T,T+\Delta)}$ and $L(0,T,T+\Delta) = \frac{1}{\Delta}\frac{P(0,T)-P(0,T+\Delta)}{P(0,T+\Delta)}$.

Under Euler Discretization, the short rate path will be discretized into constant-increment time steps of Δt, with the updated rate, r_{i+1}, given as an explicit function of r_i:

$$r_{i+1} = r_i + [\theta_{i+1} - \alpha r_i]\Delta t + \sigma \Delta W_{i+1} = r_i + [\theta_{i+1} - \alpha r_i]\Delta t + \sqrt{\Delta t}\epsilon_r$$

where $\epsilon_r \sim N(0,1)$.

Heston's Stochastic Volatility Model under risk-neutral measure reads

$$\begin{aligned}
dS_t &= rS_t dt + \sqrt{v_t} S_t dW_{1,t}^{\mathbb{Q}} \\
dv_t &= k^{\mathbb{Q}}(\Theta^{\mathbb{Q}} - v_t)dt + \sigma\sqrt{v_t}dW_{2,t}^{\mathbb{Q}} \\
\rho dt &= dW_{1,t}^{\mathbb{Q}} dW_{2,t}^{\mathbb{Q}}
\end{aligned}$$

where λ is the risk premium and $k^{\mathbb{Q}} = k + \lambda, \theta^{\mathbb{Q}} = \frac{k\theta}{k+\lambda}$.

Thus, we need to first simulate volatility path and then simulate price.

Under Euler Discretization, the volatility path will be discretized into constant-increment time steps of Δt, with the updated volatility, v_{i+1}, given as an explicit function of v_i:

$$v_{i+1} = v_i + k^{\mathbb{Q}}(\theta^{\mathbb{Q}} - v_i)\Delta t + \sigma\sqrt{v_i}\Delta W_{i+1}^{\mathbb{Q},v} = v_i + k^{\mathbb{Q}}(\theta^{\mathbb{Q}} - v_i)\Delta t + \sigma\sqrt{v_i}\sqrt{\Delta t}\epsilon_v$$

where $\epsilon_v \sim N(0,1)$.

Given v_{i+1}, asset price path can be simulated as

$$S_{i+1} = S_i \exp((r - \frac{1}{2}v_i^+)\Delta t + \sqrt{v_i^+}\Delta W_{i+1}^{\mathbb{Q},S}) = S_i \exp((r - \frac{1}{2}v_i^+)\Delta t + \sqrt{v_i^+\Delta t}\epsilon_s)$$

where $\epsilon \sim N(0,1)$ and $\epsilon_S = \rho\epsilon_v + \sqrt{1-\rho^2}\epsilon$.

```python
## Fit the bond price curve
def func(x, a, b, c):
    return a * np.exp(-b * x) + c

#curve_fit, status = calibrate_nss_ols(np.array(df['Maturity']),np.array(np.log(df['
    price'])))
def function(x):
    y = func(x,*popt)
    return y

def deriv(x):
    return derivative(function, x)

## Simulation for short rate
def theta_cal(T,a,sigma):
  return deriv(-deriv(T)) + a * (-deriv(T)) + sigma**2 / (2*a) * (1-np.exp(-2*a*T))

def simulation(strike,strike_prime,T = 3,delta = 0.25,sigma_r = 0.06907028,sigma_S =
    0.49977804):
  dt = 1/252 ## One banking date
  N = 10000 ## Number of simulation
  #T = 3
  #delta = 0.25

  ## Parameters from calibration in Hull-White
  r_0 = 0.03235033
  a = 0.00880248
  #sigma_r = 0.06907028
  p_tT, p_0t, p_0T = 0,0,0

  ## Parameters from calibration in Heston
  v_0, k, theta, rho, lbd = 0.03993293, 1.69993713, 0.04011673,  -0.70141614,
    0.40182189
  S_0 = 28162.83 ## Closed price of Nikkei 225 Index at 11/25
  k_Q = k + lbd
  theta_Q = (k * theta) / (k + lbd)
  rf = -0.001 ## Japan 1 Year Government Bond Interest Rate


  ## Simulate for Bond price
  r = np.zeros((int(N/100),int((T+delta)/dt)+1,N)) ## Initialize short rate matrix
  r[:,0,:] += r_0

  for j in range(1,int((T+delta)/dt)+1):
      eps_r = np.random.normal(0,1,(int(N/100),N))
      ## Simulate short rate
```

```
        r[:,j,:] = r[:,j-1,:] + (theta_cal(j*dt,a,sigma_r) - a*r[:,j-1,:]) * dt + sigma_r
     * np.sqrt(dt) * eps_r
  p_tT = np.mean(np.exp(-1 * (np.sum(r[:,int(T/dt):],axis = 1))*dt),axis = 0)
  p_0t = np.mean(np.exp(-1 * (np.sum(r[:,:int(T/dt)+1],axis = 1))*dt),axis = 0)
  p_0T = np.mean(np.exp(-1 * (np.sum(r,axis = 1))*dt),axis = 0)

  ## Simulate libor rate
  L_tt = (1-p_tT) / (delta * p_tT)
  L_0t = (p_0t - p_0T) / (delta * p_0T)

  ## Simulate stock price

  v = np.full(N,v_0)
  S = np.full(N,S_0)

  for j in range(1,int((T+delta)/dt)+1):

    eps_v = np.random.normal(0,1,N)
    eps = np.random.normal(0,1,N)
    eps_S = rho * eps_v + np.sqrt(1-rho**2 ) * eps
    S = S * np.exp((rf - 1/2 * v)*dt + np.sqrt(v * dt) * eps_S)
    v = v + k_Q*(theta_Q - v) * dt + sigma_S * np.sqrt(v * dt) * eps_v
    v[v < 0] = 0 ## Full Truncate Scheme


  ## Simulate option price
  C = (S/S_0-strike) * (strike_prime - L_tt/L_0t)
  C[C < 0] = 0

  return  round(np.mean(C),2)
```

7 Statement of Contributions

All works in this project, including mathematical formulation, model development and report writing, are equally split between us. We thank Prof. Alireza for his teaching and valuable insights over the semester.

References

[1] S. W. Jeng, A. Kiliçman (2021). SPX Calibration of Option Approximations under Rough Heston Model. Mathematics, 9(21), 2675.

[2] S. Gurrieri, M. Nakabayashi, T. Wong (2009). Calibration methods of hull-white model. Available at SSRN 1514192.