1. Black Scholes Model and its variant

Both classic Black-Scholes Model (BSM) and the Black 76 Model (B76) are included in *option_tools.py*. BSM prices options relative to the spot price, while B76 prices options relative to forward price. The functions in *option_tools.py* automatically use the B76 model if a forward price is given.

Black-Scholes Model:

	Call	Put
Fair Value (V)	$Se^{-qT}N(d_1) - Ke^{-rT}N(d_2)$	$Ke^{-rT}N(-d_2) - Se^{-qT}N(-d_1)$
Delta ($\partial V/\partial S$)	$e^{-qT}N(d_1)$	$-e^{-qT}N(-d_1)$
Gamma ($\partial^2 V/\partial S^2$)	$e^{-qT}\frac{N'(d_1)}{S\sigma\sqrt{T}}$	
Vega ($\partial V/\partial \sigma$)	$Se^{-qT}N'(d_1)\sqrt{T}$	
Theta ($\partial V/\partial T$)	$-e^{-qT}\frac{SN'(d_1)\sigma}{2\sqrt{T}} - rKe^{-rT}N(d_2) + qSe^{-qT}N(d_1)$	$-e^{-qT}\frac{SN'(d_1)\sigma}{2\sqrt{T}} + rKe^{-rT}N(-d_2) - qSe^{-qT}N(-d_1)$
Rho (r)	$Kre^{-rT}N(d_2)$	$-Kre^{-rT}N(-d_2)$

Where

$$d_1 = \frac{\ln\left(\frac{S}{K}\right) + (r - q + \frac{1}{2}\sigma^2)T}{\sigma\sqrt{T}}$$

$$d_2 = d_1 - \sigma\sqrt{T}$$

Black 76 Model:

	Call	Put
Fair Value (V)	$Fe^{-rT}N(d_1) - Ke^{-rT}N(d_2)$	$Ke^{-rT}N(-d_2) - Fe^{-rT}N(-d_1)$
Delta ($\partial V/\partial F$)	$e^{-rT}N(d_1)$	$-e^{-rT}N(-d_1)$
Gamma ($\partial^2 V/\partial F^2$)	$e^{-rT}\frac{N'(d_1)}{F\sigma\sqrt{T}}$	
Vega ($\partial V/\partial \sigma$)	$Fe^{-rT}N'(d_1)\sqrt{T}$	
Theta ($\partial V/\partial T$)	$-e^{-rT}\frac{FN'(d_1)\sigma}{2\sqrt{T}} - rKe^{-rT}N(d_2) + rFe^{-rT}N(d_1)$	$-e^{-rT}\frac{FN'(d_1)\sigma}{2\sqrt{T}} + rKe^{-rT}N(-d_2) - rFe^{-rT}N(-d_1)$
Rho (r)	$-re^{-rT}(FN(d_1) - KN(d_2))$	$-re^{-rT}(KN(-d_2) - FN(-d_1))$

Where

$$d_1 = \frac{\ln\left(\frac{F}{K}\right) + \frac{1}{2}\sigma^2 T}{\sigma\sqrt{T}}$$

$$d_2 = d_1 - \sigma\sqrt{T}$$

2. Implied forward price

Using put-call parity and the forward price formula, we can derive the implied forward price:

$$\begin{cases} C + Se^{-qT} = P + Ke^{-rT} \\ F = Se^{(r-q)T} \end{cases}$$

This gives us:

$$F = (C - P)e^{rT} - K$$

Therefore, for each T and K, we can use the market quotes to calculate the implied forward price for the underlying stock. We will use the implied forward price and the Black 76 Model for this project, rather than the spot price and the BSM. The reasons are as follows:

(a) The current market prices of the options (mid-price of the quotes) are almost never aligned with the spot prices in BSM as traders like to price options relative to the forward price.
(b) The difference between the put's implied volatility and the call's implied volatility can be very large if the quotes are too far away from the BSM fair values. Put-call parity dictates that if both options are priced using Black Scholes Model with the same parameters, their implied volatility should be the same. Using the implied forward price and B76 can ensure that both call and put use the same parameters.
(c) BSM needs dividend yield as an input, but dividend yield is hard to estimate from historical data. Instead, forward price has priced in the market's expectation of the dividend yield.
(d) The spot price at the moment when the option quotes were given was unknown.

3. Volatility surface interpolation

The volatility surface needs to be interpolated across strikes and tenors.

For strikes, we will use the cubic spline interpolation. Similar to most spline interpolation methods, cubic spline uses a set of coefficients on the cubic polynomials used to interpolate the data. The coefficients bend the line so that it passes through each of the data points without any erratic behaviour or breaks in continuity. We chose cubic spline interpolation because it preserves local monotony and convexity. The monotony and convexity of interpolated data within a small range around the original data points are generally preserved.

For tenors, we use linear interpolation on the total variance. Note that to avoid calendar arbitrage, volatility at t_1 and t_2 must satisfy:

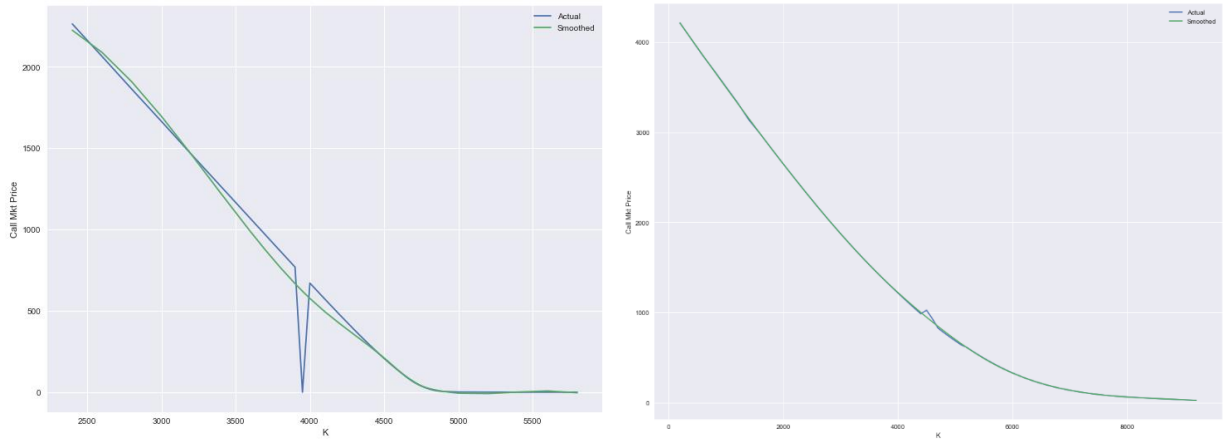
$$\sigma_{t_1}^2 t_1 + \sigma_{\Delta t}^2 \Delta t = \sigma_{t_2}^2 t_2$$

Where $t_1 + \Delta t = t_2$. This, intuitively, is similar to the interest rate term structure. We interpolate $\sigma_T^2 T$ linearly across the iso-strike lines. That is, on a iso-strike line, if we have (σ_{t_1}, t_1) and (σ_{t_2}, t_2), then the "forward volatility" σ_f between t_1 and t_2 will be $\sqrt{(\sigma_{t_2}^2 t_2 - \sigma_{t_1}^2 t_1)/(t_2 - t_1)}$. Thus, for any tenor t' between t_1 and t_2, the interpolated volatility should be $\sqrt{(\sigma_{t_1}^2 t_1 + \sigma_f^2 (t' - t_1))/t'}$.

4. Arbitrage-free smoothing

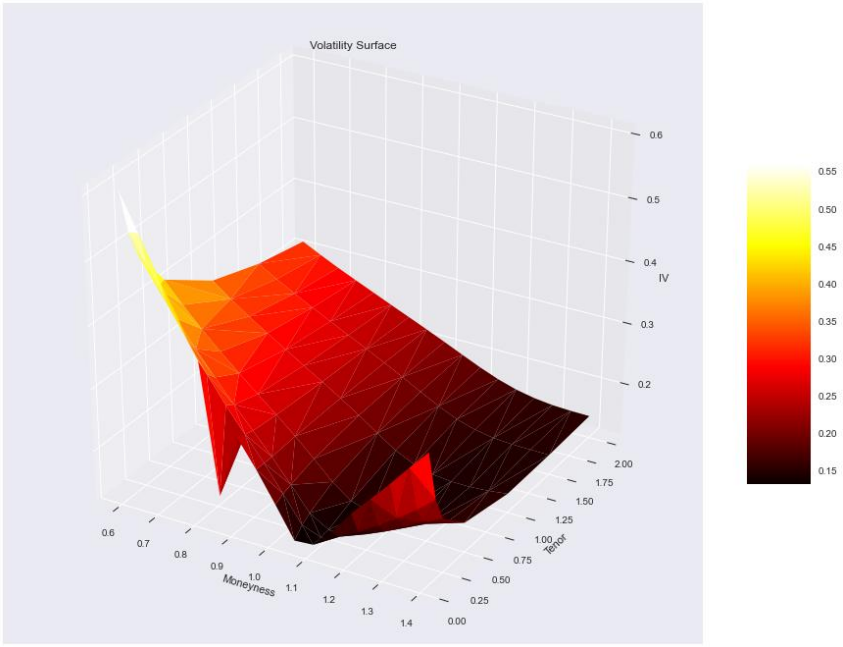
Occasionally, the option prices show have arbitrage opportunities that cannot be justified by carry costs. We will not examine calendar arbitrage for this project as calendar arbitrage opportunities are highly unlikely, especially since volatility also has term structure that moves with time. Arbitrages on strikes are much more common and easy to detect. The C-K curve must be monotonically decreasing and convex. If there is a "bump" or "dip" on the curve, then an arbitrage opportunity occurs. To eliminate the occasional arbitrages, we can replace the bumps and dips with smoothed local values. Fengler (2005) proposed a cubic spline smoother with constraints for this issue. The outcome of smoothed price curve is different from the original one in two aspects: (a) Arbitrage-free prices will be slightly changed and the difference does not have a significant impact on the volatility surface; (b) Arbitrage prices are replaced with cubic spline interpolated values that preserve monotony and convexity. That is, to sacrifice a small amount of accuracy in exchange for overall arbitrage-free surface.

The algorithm is not extremely complicated, but it can take some time to code. Due to time limits, I coded an unconstrainted cubic spline smoother just for illustrative purpose. See the following examples of actual C-K curve and the smoothed arbitrage-free (supposedly) curve based on Jan 14 2022 quotes:

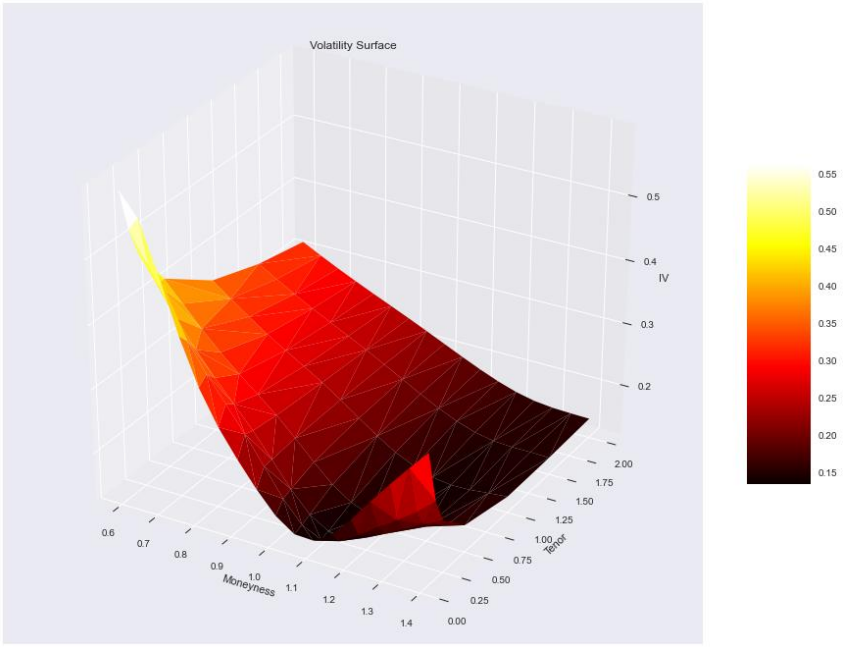


The left chart is two weeks to expiration and the right chart is 5 years. The unconstrainted cubic spline smoother does not work well on short maturities because deeply out-of-money option prices are close to zero at short maturities, and the smoother will produce some negative option prices.

To save some time, we will smooth the volatility surface rather than on the option prices. Using the 18 Jan 2022 quotes, the unsmoothed volatility surface looks like this:



The downward spike at 0.85 moneyness and 0.08 years indicate that some quotes around there are abnormal. A possible issue that the bid-ask spread is too large for an option that is supposed to be worth 0, and the mid-price is too far away from 0. Applying the cubic spline smoother along iso-tenor lines, we got a much smoother surface:



4. About the code

There are three scripts, option_tools.py, smoother_tools.py, and main.py.

option_tools.py is the primary package for option calculations, including Black-Scholes, implied volatility, volatility surface, binary tree, etc. There are detailed comments in option_tools.py, and hopefully they will be enough to explain everything.

smoother_tools.py contains smoothers, i.e. cubic spline smoothers. Note that although cubic spline interpolation is included in scipy.interpolate, there is no public package for cubic spline smoothing. Smoother_tools.py is preliminary collection of smoothers.

main.py calls the functions in option_tools.py to import data, calculate the volatility surface and greeks, and export the results. Run everything in main.py and it will produce charts of two unsmoothed volatility surfaces, charts of two smoothed volatility surfaces, a .xlsx file with two smoothed surfaces, and a .csv file with all option information including the greeks. Due to the high volume of the quotes, it roughly takes 3-5 minutes finish calculating everything.

Please pip install quandl before running main.py. Live yield curve data is obtained from quandl. Alternatively, I have downloaded the yield curve data and we can read_csv without installing quandl.