E.3 No-Arbitrage Price Surface

In this section, we describe how to calibrate an arbitrage-free price surface on each business day from the market option prices. We want to create a parametrization of the price surface that is **arbitrage-free**. We use use SABR model to match the market volatility smiles and we use an arbitrage-free interpolation based on Local Volatility Function (LVF) model to interpolate the SABR model value between expiries.

In this thesis, we choose the SABR model over LVF models in matching market smiles and computing the value of options with the strikes unobserved in the market. This is because LVF models that were initially proposed by Dupire et al. [68], Derman and Kani [60] and Rubinstein [158] and put into highly efficient pricing engines by Andersen and Brotherton-Ratcliffe [5] and Dempster and Richards [59] amongst others, heavily rely on an arbitrage-free BS implied volatility surface as the input. If there are arbitrage violations, the convergence of the algorithm solving the underlying generalized Black Scholes partial differential equation will be obstructed [75]. Unfortunately, an arbitrage-free BS implied volatility surface as the input is not guranteed in practice. For instance, we in this thesis observe market bid and ask quotes of European options. The mid prices are used in calibrating pricing functions. The input of BS implied volatilities from mid prices are not guaranteed to be arbitrage-free. Therefore, we need to remove the arbitrage of the BS implied volatilities in the input data either manually or use arbitrage-free smoonthing algorithm as in [75] before using it as the input in LVF pricing. Another problem as indicated by Hagan et al. [99] is that the dynamics of the market smile predicted by LVF is opposite to the market behavior. The contradiction between model and market lead to unstable sensitivities (delta, vega) computation. Hedging using the delta from LVF may perform worse than hedging using BS implied delta [99]. [1]

Surface calibration models based on LVF that do not reply on the assumption that the input BS implied volatility surface is arbitrage-free exists. For example, volatility interpolation algorithm from Andreasen and Huge [6] can create an arbitrage-free surface without assuming the input is arbitrage-free. However, the resulting model cannot interpolate and extrapolate in strikes as we will discuss in more details in later section. Therefore, in this thesis, we only use the volatility interpolation algorithm [6] to interpolate the model value from SABR between market available expiries.

E.3.0.1 Arbitrage-Free Surface From SABR Model

In this section, we discuss how to use SABR model to create an arbitrage-free surface calibrated to match market available prices. Although SABR model is computationally efficient and can match the market volatility smile well, it is not arbitrage-free. The formula (2.2.11) is an approximation, obtained from an asymptotic series expansion. Its accuracy degrades if the option

[1] Note that we also use SABR-Bartlett delta as the comparing hedging method, we will need to calibrate the SABR model for each expiry anyway.

strikes move away from the option at-the-money (ATM) strike. Therefore, we also discuss how to fix the arbitrage issue of SABR.

Following [85], we check whether an implied volatility surface is free of *calendar arbitrage*, and *butterfly arbitrage*, which we describe below. Assume that we have a collection of European call option prices $\{C(T,K)\}_{T,K}$ for a range of strikes, K, and expiries, T, with the Black-Scholes implied volatility surface $\{\sigma^{imp}(T,K)\}_{T,K}$. We also suppose that interest rates are deterministic with $D(t,T) = e^{-r(T-t)}$ for the discount factor, where t is trading date and T is the maturity date.

1. Butterfly Arbitrage: At time t, given a collection of call option prices $\{C(T,K)\}_{T,K}$, using Dupire's method [68], one can write the option value with an implied probability density $p(\cdot;T,S_t)$ such that

$$C(T,K) = D(t,T)\int_{(0,\infty)}(S-K)_+ p(S;T,S_t)dS.$$

   We say that the surface $\{\sigma^{imp}(T,K)\}_{T,K}$, where $\sigma^{imp}(T,K)$ is the Black-Scholes implied volatility, is free of Butterfly Arbitrage if its implied probability density $p(\cdot;T,S_t)$ is a valid density, i.e., $p(S;T,S_t) \geq 0$ for all $S > 0$ and $\int_0^\infty p(S;T,S_t)dS = 1$. Additionally, the condition $p(S;T,S_t) \geq 0$ for all $S > 0$ is equivalent to require $\frac{\partial^2 C(T,K)}{\partial K^2} \geq 0$ for all $K > 0$ since Breeden and Litzenberger [24] show that:

$$\left.\frac{\partial^2 C(T,K)}{\partial K^2}\right|_{K=x} = D(t,T)p(x;T,S_t)$$

2. Calendar Arbitrage: Given a surface $\sigma^{imp}(T,K)$, we consider, at the time t, the corresponding total variance (TV) surface defined by

$$w(\tau,k) = \sigma^{imp}(T,K)^2\tau$$

   where $\tau = T-t$ and k is parameterized by log-moneyness, i.e. $k := \log(K/F(t;T))$, and $F(t;T) = S_t e^{(r-q)(T-t)}$ is the at-the-money forward (ATMF) price for S_t. Let $t = T_0 < T_1 < \cdots < T_M$ be a set of expiries. We say that the surface $\{\sigma^{imp}(T,K)\}_{T,K}$ is free of Calendar Arbitrage

   - if $\frac{\partial w(\tau,k)}{\partial T} \geq 0$ for all $k \in \mathbb{R}, T > 0$ for continuous time data
   - if $w(\tau_i,k) \leq w(\tau_{i+1},k)$ for all $k \in \mathbb{R}, T_i < T_{i+1}$ for discrete time data

Furthermore, given a grid of strikes: $0 = K_0 < K_1 < \cdots < K_N$, and a grid of expiries $t = T_0 < T_1 < \cdots < T_M$. The corresponding discrete criteria [33] for a grid of option prices to be free of arbitrage are set as the following with $j = 1,\ldots,N-1$:

1. No butterfly spread arbitrage condition:

$$C(T_j,K_{i-1})-C(T_j,K_i)>\frac{K_i-K_{i-1}}{K_{i+1}-K_i}\left(C(T_j,K_i)-C(T_j,K_{i+1})\right) \tag{E.1}$$

2. No calendar spread arbitrage:

$$C(T_j,K_i)\geq C(T_{j-1},K_i) \tag{E.2}$$

In this thesis, we will use SABR model to obtain the option price for an strike K unobserved in the market when T is an expiry listed in the exchange.

We denote the grid of market observed expiries on a business date t to be

$$\mathbf{T}_t^{mkt}=\{T_0<T_1<\cdots<T_M\}$$

We set the first expiry be t: $T_0=t$. Note that for the first expiry $T_0=t$, the market option value at t is just the option payoff at t which is $\max\{S_t-K,0\}$ (call) or $\max\{K-S_t,0\}$ (put). For an expiry T_i, let us further define:

$$\mathbf{K}^{mkt}(t,T)=\{K_{t,T,1}^{mkt},\ldots,K_{t,T,N_K}^{mkt}\}$$

to be the grid of strikes K observed in market on date t for which we have market option value for the expiry T

To calibrate an option value function with the market prices under SABR model, given T_i and fix $\beta=1$, we solve

$$\min_{\alpha,\nu,\rho}\sum_{K\in\mathbf{K}^{mkt}(t,T_i)}\left(V_{SABR}(S_t,t,T_i,K,r,q;\alpha,\beta,\nu,\rho)-V_{t,T_i,K}^{mkt}\right)^2$$

where $V_{SABR}(S,t,T,K,r,q;\alpha,\beta,\nu,\rho)$ is option pricing function described in section 2.2.3 with the approximation formula (2.2.11). Hagan et al. [99] suggest that β can be chosen from prior beliefs about which assumption on the distribution of S_T is appropriate (e.g., $\beta=0$ implies a normal distribution of S_T conditioned on a realization of the volatility while $\beta=1$ implies a lognormal distribution of S_T conditioned on a realization of the volatility). In this thesis, we fix $\beta=1$ in the calibration process since we are dealing equity options. If we are dealing with interest rate derivatives, setting $\beta=0$ or $\beta=0.5$ may be more appropriate. In practice, the choice of β has little effect on the resulting shape of the volatility curve produced by the SABR model. Hagan et al. [99] suggest that the choice of β is not crucial in matching the market volatility smile. Furthermore, Bartlett [16] suggests that the choice of β is also not crucial when we use the bartlett delta as in equation (2.3.18) as the delta position from SABR model.

We choose to calibrate a different pricing function model for each expiry. A different set of parameters is specified for each expiry, describing an instantaneous process. We choose this approach because the single implied volatility surface calibrated for all expiries and strikes is

unlikely to fit the actual surface very well. In addition, calibrating a single surface is harder and more time-consuming.

Note that we have three parameters to be calibrated so we need to observe at least 3 data points from market to successfully build a SABR model. Therefore, the number of strikes N_K is expected be larger than or equal to 3. [2]

Due to the fact that equation (2.2.11) being an appproximation, the implied probability density function:

$$\frac{1}{D(t,T)} \frac{\partial^2 C_{SABR}(T,K)}{\partial K^2}\bigg|_{K=x} = p(x;T,S_t) \tag{E.3}$$

where $C_{SABR}(T,K)$ is the SABR pricing function of a call option at strike K and expiry T computed using the equation (2.2.11), may become negative at very low or very high strikes. Therefore, we may observe butterfly spread arbitrage in SABR prices returned by the calibrated models. In addition, for each expiry, a separate set of SABR parameters is calibrated so that calender spread arbitrage can also exist. However, the existence of calendar arbitrage will be rare since the market option prices rarely contains calendar arbitrage and therefore the SABR model, which usually matches the market option data very well, rarely produces calendar arbitrage.

Given a grid of strikes K for which we aim to use SABR model to produce the option prices, if we found that the grid of prices returned by the SABR model has failed the **discrete** arbitrage conditions for butterfly arbitrage (E.1), we will introduce some adjustments. To fix the butterfly spread arbitrage, we implements a risk-neutral adjustment. This adjustment substitutes the two implied distribution tails by those of certain log-normal distributions. The following adjustment is inspired by [27]. Interested reader can refer to [27] for more details. Here we just briefly discuss the process of the adjustment.

Firstly, we introduce lower and upper strike limits, K_L and K_U within which the implicit probability density function (p.d.f) $p(x;T,S_t)$ from SABR model is assumed to be valid. The lower and upper strike limit can be the maximum and minimum strike K for which the discrete no butterfly spread arbitrage condition (E.1) holds. Brunner and Hafner [27] set the tail distributions as the mixture of two lognormal distributions:

$$\hat{g}(x) = \begin{cases} \lambda_L\, q(x;\mu_L^1,\sigma_L^1) + (1-\lambda_L)\, q(x;\mu_L^2,\sigma_L^2), & \text{if } 0 < x < K_L \\ p(x;T,S_t), & \text{if } K_L \le x \le K_U \\ \lambda_U\, q(x;\mu_U^1,\sigma_U^1) + (1-\lambda_U)\, q(x;\mu_U^2,\sigma_U^2), & \text{if } x > K_U \end{cases} \tag{E.4}$$

where $q(x;\mu,\sigma)$ is the p.d.f of a log-normal distribution:

$$q(x;\mu,\sigma) = \frac{1}{x\sigma\sqrt{2\pi}} e^{-\frac{(\ln(x)-\mu)^2}{2\sigma^2}}.$$

and $p(x;T,S_t)$ is the implied probability density from the calibrated SABR model. Here, the

[2] The $V_{t,T,K}^{mkt}$ is the mid-price of market observed best bid and best ask prices.

parameters to be determined are:

$$\{\lambda_L, \mu_L^1, \sigma_L^1, \mu_L^2, \sigma_L^2, \lambda_U, \mu_U^1, \sigma_U^1, \mu_U^2, \sigma_U^2\}.$$

In this thesis, we assume the adjusted p.d.f is of the following simplified form:

$$\hat{g}(x) = \begin{cases} \lambda_L \, q(x; \mu_L, \sigma_L), & \text{if } 0 < x < K_L \\ p(x; T, S_t), & \text{if } K_L \leq x \leq K_U \\ \lambda_U \, q(x; \mu_U, \sigma_U), & \text{if } x > K_U \end{cases} \tag{E.5}$$

We assume that the underlying price at expiry at T: S_T is distributed according the adjusted p.d.f $\hat{g}(x)$. We choose the equation (E.5) becasue it has a simpler solution than the equation (E.4), for which we need to solve an overdetermined non-linear system.

We require the following condition to be satisfied:

1. Intergrability constraint

$$\int_0^{K_L} \lambda_L q(x; \mu_L, \sigma_L) dx + \int_{K_L}^{K_U} p(x; T, S_t) dx + \int_{K_U}^{\infty} \lambda_U \, q(x; \mu_U, \sigma_U) dx = 1 \tag{E.6}$$

2. Martingale constraint

$$\int_0^{K_L} x\lambda_L q(x; \mu_L, \sigma_L) dx + \int_{K_L}^{K_U} x p(x; T, S_t) dx + \int_{K_U}^{\infty} x\lambda_U \, q(x; \mu_U, \sigma_U) dx = F(t, T) \tag{E.7}$$

   The intergrability constraint ensures that $\hat{g}(x)$ is a valid p.d.f. The martingale constraint ensures that $E[S_T] = F(t,T) = S_t e^{(r-q)(T-t)}$ under ther adjusted p.d.f.

Since they are six unknown parameters $\{\mu_L, \mu_U, \sigma_L, \sigma_U, \lambda_L, \lambda_U\}$, additional calibration conditions are imposed. Observing that the Black-Scholes [20] model implies that, under the risk neutral measurement, the prices of the underlying asset S_T at the maturity T are log-normal distribued:

$$\ln(S_T) \sim \mathcal{N}(\ln(S_t) + (r - q - \frac{\sigma^2}{2})(T-t), \sigma^2(T-t))$$

we set $\mu_L = \ln(S_t) + (r - q - \frac{\sigma_L^2}{2})(T-t)$ and $\mu_U = \ln(S_t) + (r - q - \frac{\sigma_U^2}{2})(T-t)$ such that we have only four parameters to be solved: $\{\sigma_L, \sigma_U, \lambda_L, \lambda_U\}$. Furthermore, with $\mu_L = \ln(S_t) + (r - q - \frac{\sigma_L^2}{2})(T-t)$ and $\mu_U = \ln(S_t) + (r - q - \frac{\sigma_U^2}{2})(T-t)$, one can easily verify that if we have $\{\sigma_L = \sigma_B(K_L), \sigma_U = \sigma_B(K_U), \lambda_L = 1, \lambda_U = 1\}$, intergrability constraint (E.6) and martingale constraint (E.7) will be satisified.

We use the adjusted p.d.f (E.5) for option pricing

$$C_{\hat{g}(x)}(T, K) = D(t, T) \int_{-\infty}^{\infty} (x - K)_+ \hat{g}(x) dx \tag{E.8}$$

For simplicity, we write $\sigma_B(K)$ to denote the implied Black's volatility given by the SABR formula (2.2.11) for a strike K since the other inputs in $\sigma_B(F,t,T,K;\alpha,\beta,\nu,\rho)$ as in formula (2.2.11) remain unchanged in the following discussion for an expiry T at a time t.

By setting:

$$\begin{aligned}
&\sigma_L = \sigma_B(K_L)\ ,\ \sigma_U = \sigma_B(K_U)\\
&\lambda_L = 1\ ,\ \lambda_U = 1\\
&\mu_L = \ln(S_t) + (r - q - \frac{\sigma_B(K_L)^2}{2})(T-t)\\
&\mu_U = \ln(S_t) + (r - q - \frac{\sigma_B(K_U)^2}{2})(T-t)
\end{aligned} \tag{E.9}$$

we can easily verify that (E.8) can be written as:

$$\overline{C}_{SABR}(T,K) \leftarrow C_{\hat{g}(x)}(T,K) = \begin{cases} C_{BS}(T,K;\sigma_B(K_L)) & \text{if } 0 < K < K_L \\ C_{SABR}(T,K) & \text{if } K_L \le K \le K_U \\ C_{BS}(T,K;\sigma_B(K_U)) & \text{if } K > K_U \end{cases}$$

Here we use the $\overline{C}_{SABR}(T,K)$ to indicate it is the SABR model value after the fix for the butterfly arbitrage. Since the adjusted p.d.f (E.5) with the setting (E.9) satisfies intergrability constraint (E.6) and martingale constraint (E.7) and will not be negative at tails of the distribution because that the two tails are from two log-normal distribuions, we can conclude that the adjusted $\overline{C}_{SABR}(T,K)$ will be free of butterfly arbitrage. In appendix B, we show in details that the adjusted p.d.f (E.5) with the parameters setting as in equations (E.9) will satisfy the intergrability constraint (E.6) and martingale constraint (E.7).

For calendar arbitrage, we will shift the price by the the following procedure to remove it. Suppose we have a grid of $\mathbf{K}_{grid} = \{K_0, K_1, \ldots, K_N\}$ and a grid of expiries $t = T_0 < T_1 < \cdots < T_M$. Then for each expiry $T_j, i = 1, \ldots, M$, we have:

$$shift_{T_j} = -min\left\{ min_{K \in \mathbf{K}_{grid}} \left[\overline{C}_{SABR}(T_j,K) - \overline{C}_{SABR}(T_{j-1},K) \right], 0 \right\}.$$

$$\widetilde{C_{SABR}}(T_j,K) \leftarrow \overline{C}_{SABR}(T_j,K) + shift_{T_j}$$

Note the shift will be zero if no calendar arbitrage is observed between T_j and T_{j-1}. Note that the constant shift will preserve the no butterfly arbitrage condition from $\overline{C}_{SABR}(T,K)$, one can easily see this by noting that no butterfly arbitrage implies that:

$$\overline{C}_{SABR}(T_j,K_{i-1}) - \overline{C}_{SABR}(T_j,K_i) > \frac{K_i - K_{i-1}}{K_{i+1} - K_i} \left(\overline{C}_{SABR}(T_j,K_i) - \overline{C}_{SABR}(T_j,K_{i+1}) \right)$$

Therefore, we still have

$$\begin{aligned}&[\overline{C}_{SABR}(T_j,K_{i-1})+shift_{T_j}]-[\overline{C}_{SABR}(T_j,K_i)+shift_{T_j}]\\&>\frac{K_i-K_{i-1}}{K_{i+1}-K_i}\left\{\left[\overline{C}_{SABR}(T_j,K_i)+shift_{T_j}\right]-\left[\overline{C}_{SABR}(T_j,K_{i+1})+shift_{T_j}\right]\right\}\end{aligned}$$

Lastly, there is a no call-spread arbitrage condition that requires the call option value decreasing in strikes:

$$C(T,K_{i+1})\leq C(T,K_i) \text{ when } K_{i+1}>K_i.$$

We comments that call-spread arbitrage violation is rare and no violations of this condition from the prices produced by SABR model are observed for all the computational results in this thesis. Additionally, by the call-put parity, if we have no butterfly arbitrage and no calendar arbitrage with call option prices, then we will have no butterfly arbitrage and no calendar arbitrage in the put option prices as well.

E.3.0.2 Alternative Methods to SABR Calibration

Given BS implied volatility surface computed from market option values as a discrete set of points $\{\sigma^{imp}_{BS,mkt}(T,K)\}_{T,K}$, a natural question is whether arbitrage exists. In most cases, one must

1. Fit a parameterization to the points $\{\sigma^{imp}_{BS,mkt}(T,K)\}_{T,K}$, typically by time-slice as what we do with SABR model, obtaining a parameterization $\widetilde{\sigma}^{imp}_{model}(T_i,K)$ for each T_i separately. For example, if we fit a SABR model, $\widetilde{\sigma}^{imp}_{model}(T_i,K)=\sigma_B(F,t,T_i,K;\alpha,\beta,\nu,\rho)$ and since except K all other parameters are fixed for an expiry T_j on a time t, it is a function of K only.

2. Compute $\{\widetilde{w}(\tau_i,k)\}_{i=1}^N$ on a fine grid of k using the expression $\widetilde{w}(\tau_i,k):=\widetilde{\sigma}^{imp}_{model}(T_i,K)^2\tau_i$, where $\tau_i=T_i-t$ and k is parameterized by log-moneyness, i.e. $k:=\log(K/F(t;T))$.

3. Check for the discrete no-arbitrage criteria (E.1) and (E.2) as in [33] for a given grid of strikes and a given grid of expiries.

To obtain such mappings, one typically fits a stochastic volatility model like SABR or Heston to slices of $\{\sigma^{imp}_{BS,mkt}(T,K)\}_{T,K}$ or considers a generalized model such as Stochastic Volatility Inspired (SVI) parameterizations.

The Stochastic Inspired Volatility model (SVI) [83] was used internally at Merrill Lynch and publicly disclosed by Jim Gatheral in 2004. SVI is a simple five-parameters model. In 2012, the Surface SVI (SSVI) [85] is proposed by Gatheral and Jacquier to extend SVI model to be a model that can fit the whole surface instead of just one volatility smile. The SSVI is parameterized in a way that a SSVI slice at a given maturity T is a SVI slice with only 3 parameters. This restriction leads to explicit sufficient conditions for the absence of arbitrage, while allowing

enough flexibility for calibration. The SSVI model is recently extended in [102] and [53]. If the input market data used for calibration contains arbitrage, the calibrated surface from SSVI [85] or its extension [102, 53] can typically be viewed as the surface that is as close as possible to the original market data, while staying arbitrage-free. In this section we briefly review a recent variant dd-eSSVI (data-driven extended SSVI) method [53] as an example.

E.3.0.3 dd-eSSVI Parameterization

Following the work of [53], we introduce the following SSVI parameterization for a surface's Total Variance (TV), $w(\tau,k)$ as

$$w(\tau,k) = \frac{\hat{\theta}_\tau}{2}\left(1+\hat{\rho}_\tau\hat{\psi}_\tau k + \sqrt{(\hat{\psi}_\tau k+\hat{\rho}_\tau)^2+(1-\hat{\rho}_\tau^2)}\right) \tag{E.10}$$

In this parameterization we have that

- $\hat{\theta}_\tau$ is the ATM Forward TV which can be extracted from market directly.

  $$\hat{\theta}_\tau = w(\tau,0) = \sigma^{imp}_{BS,mkt}(T,K_{ATMF})^2\cdot\tau$$

  where $K_{ATMF} = F(t,T)$

- $\hat{\rho}_\tau$ controls the slope of the skew

- $\hat{\psi}_\tau$ controls the curvature, which is usually defined as a function of $\hat{\theta}_\tau$: $\hat{\psi}_\tau(\hat{\theta}_\tau)$

An important feature of this parameterizations is that it provides easy way to impose sufficient conditions on the parameters $(\hat{\theta}_\tau,\hat{\rho}_\tau,\hat{\psi}_\tau)$ so that there is no butterfly arbitrage for a given slice, and no calendar arbitrage between two time slices. Interested reader can refer to [53, 85] for the detailed conditions on those parameters.

There are many different approaches for calibrating SSVI models. For example, one can fit SSVI model to market data without imposing any constraints on the parameters and then check if any arbitrage exists by imposing the arbitrage conditions on the calibrated parameters. If arbitrage conditions are violated, one can adjust the parameters so that the sufficient conditions on parameters are satisfied. One can also use an arbitrage-free calibration [53] by imposing the sufficient conditions on the parameters into the calibration process. Interested reader can refer to [102, 53] for more details on how to calibrate the SSVI efficiently.

Since the major goal in this thesis is not to compare arbitrage-free surface calibration methods, we leave the exploration of the alternative methods to calibrate the arbitrage-free surface and comparing its impact on the data-driven risk hedging model as the future work of our study.

E.3.0.4 Volatility Interpolation Between Expiries

In the previous section, for each T_i, we calibrate a separate set of SABR parameters and we then use the calibrated SABR parameters to compute option price and the associated implied volatiliy for a predetermined grid of strikes for each T_i. We correct for butterfly arbitrage and calendar arbitrage if we detect any of them in the option values produced from the SABR models. However, after the SABR calibration, we only obtain the parametrization of option values for expiries listed in the market. Our next goal is get the parametrization of option values for expiries that are not available in the market. In this section, we discuss how to interpolate the volatility between different expiries. Note that even if we use SSVI instead of SABR model, this step of volatility interpolation between expiries is still needed since SSVI model and SABR model are both calibrated to match the volatility smile of market for each market expiry only. Under SSVI models, one usually interpolates the SSVI parameters $\{\hat{\theta}_\tau, \hat{\rho}_\tau, \hat{\psi}_\tau\}$ between different expiries available in market [53].

Andreasen and Huge [6] have introduced an efficient and arbitrage-free volatility interpolation method based on an one step finite difference implicit Euler scheme applied to a local volatility parametrization. In this thesis, we use the volatility interpolation approach to compute option price for an arbitrary expiry T unobserved in the market.

The volatility interpolation method is based on the Dupire's equation [68]. The Dupire's equation enables us to deduce the volatility function in a local volatility model from put and call options in the market. Under a risk-neutral measure, we assume:

$$\frac{dS_t}{S_t} = (r-q)\,dt + \sigma(t,S_t)dZ_t$$

where r is the risk-free interest rate and q is the dividend yield. Let $C(T,K)$ be the call option pricing function, Dupire's equation states:

$$\frac{\partial C(T,K)}{\partial T} = \frac{1}{2}\sigma^2(T,K)K^2\frac{\partial^2 C(T,K)}{\partial K^2} - (r-q)K\frac{\partial C(T,K)}{\partial K} - qC(T,K)$$

Define the normalized call price in terms of discounting factor $D(t,T)$ and forward price $F(t,T)$ and the normalized strike $\widehat{K}$ as:

$$\begin{aligned} D(t,T) &= e^{-r(T-t)} \\ F(t,T) &= S_t e^{(r-q)(T-t)} \\ \widehat{K} &= \frac{K}{F(t,T)} \\ \widehat{C}(T,\widehat{K}) &= \frac{C(T,\widehat{K}F(t,T))}{D(t,T)F(t,T)} = \frac{C(T,K)}{D(t,T)F(t,T)} \end{aligned}$$

Dupire's equation can be simplified as [6][3]:

$$\frac{\partial \widehat{C}(T,\widehat{K})}{\partial T} = \frac{1}{2}\widehat{\sigma}^2(T,\widehat{K})\widehat{K}^2\frac{\partial^2 \widehat{C}(T,\widehat{K})}{\partial \widehat{K}^2}, \;\; \widehat{\sigma}(T,\widehat{K}) = \sigma(T,K)$$

Therefore, we can sequentially solve the finite difference discretization of the Dupire's forward equation using the fully implicit method. Observing that $T_0 = t$, on the trading date t, the option value expiring at t is just option payoff, we have the the initial condition $\widehat{C}(T_0,\widehat{K}) = max(1-\widehat{K},0)$. Furthermore, when $K = 0$, we arrive at the lower boundary condition $\widehat{C}(T,0) = 1$. When the largest strike $K_{max} \gg S_t$, we assume the upper boundary condition $\widehat{C}(T,\widehat{K}_{max}) = 0$ is true.

Assume we are given a grid of expiries available in market $t = T_0 < T_1 < \cdots < T_M$ and a grid of normalized strike: $0 = \widehat{K}_0 < \widehat{K}_1 < \cdots < \widehat{K}_N = \widehat{K}_{max}$, Andreasen and Huge [6] assume $\widehat{\sigma}(T,\widehat{K})$ to be a piecewise constant functions for a given T_i.

$$\widehat{\sigma}(T_i,\widehat{K}) = \begin{cases} \widehat{\sigma}_{T_i,\widehat{K}_0}, & \text{if } \widehat{K} \le \widehat{K}_0 \\ \vdots & \vdots \\ \widehat{\sigma}_{T_i,\widehat{K}_j}, & \text{if } \widehat{K}_{j-1} < \widehat{K} \le \widehat{K}_j \\ \vdots & \vdots \\ \widehat{\sigma}_{T_i,\widehat{K}_N}, & \text{if } \widehat{K} > \widehat{K}_N \end{cases} \tag{E.11}$$

The authors further assume:

$$\left.\frac{\partial^2 \widehat{C}(T,\widehat{K})}{\partial \widehat{K}^2}\right|_{\widehat{K}=0} = \left.\frac{\partial^2 \widehat{C}(T,\widehat{K})}{\partial \widehat{K}^2}\right|_{\widehat{K}=\widehat{K}_N} = 0 \tag{E.12}$$

From (E.3), one can see the second partial derivative of call prices with regards to strike K is the implied density:

$$\frac{1}{D(t,T)}\left.\frac{\partial^2 C(T,K)}{\partial K^2}\right|_{K=x} = p(x;T,S_t)$$

Therefore, the boundary conditions (E.12) essentially assume that the probablity density at the low strike boundary ($K = 0$) and high strike boundary ($K = K_N$) is zero, which is a reasonable assumption.

[3] We use the call option as the example but the analysis also holds for put options. More specifically, let $P(T,K)$ be the function of put option price, the Dupire's equation for put option is:

$$\frac{\partial \widehat{P}(T,\widehat{K})}{\partial T} = \frac{1}{2}\widehat{\sigma}^2(T,\widehat{K})\widehat{K}^2\frac{\partial^2 \widehat{P}(T,\widehat{K})}{\partial \widehat{K}^2}, \;\; \widehat{\sigma}(T,\widehat{K}) = \sigma(T,K).$$

The fully implicit finite difference method in matrix form is:

$$\begin{bmatrix} \widehat{C}(T_i,\widehat{K}_0) \\ \widehat{C}(T_i,\widehat{K}_1) \\ \widehat{C}(T_i,\widehat{K}_2) \\ \vdots \\ \widehat{C}(T_i,\widehat{K}_N) \end{bmatrix} = (I - \mathbb{S}\mathbb{D}(T_{i+1}-T_i)) \begin{bmatrix} \widehat{C}(T_{i+1},\widehat{K}_0) \\ \widehat{C}(T_{i+1},\widehat{K}_1) \\ \widehat{C}(T_{i+1},\widehat{K}_2) \\ \vdots \\ \widehat{C}(T_{i+1},\widehat{K}_N) \end{bmatrix} \tag{E.13}$$

where I is the identity matrix, $\mathbb{S}$ is a diagonal matrix parameterized by $\widehat{\sigma}(T_i,.)$ as in equation (E.11), $\mathbb{D}$ is proportional to the discrete second order difference matrix and $(T_{i+1}-T_i)$ is a scaler. Specifically:

$$\mathbb{S} = \begin{bmatrix} \widehat{\sigma}^2(T_i,\widehat{K}_0) & & \\ \vdots & \ddots & \vdots \\ & & \widehat{\sigma}^2(T_i,\widehat{K}_N) \end{bmatrix} \tag{E.14}$$

$$\mathbb{D} = \begin{bmatrix} 0 & 0 & & & & \\ l_1 & -l_1-u_1 & u_1 & & & \\ & l_2 & -l_2-u_2 & u_2 & & \\ & & \ddots & \ddots & \ddots & \\ & & & l_{N-1} & -l_{N-1}-u_{N-1} & u_{N-1} \\ & & & & 0 & 0 \end{bmatrix} \tag{E.15}$$

where

$$l_j = \frac{1}{\widehat{K}_{j+1}-\widehat{K}_{j-1}} \frac{1}{\widehat{K}_j - \widehat{K}_{j-1}}$$

$$u_j = \frac{1}{\widehat{K}_{j+1}-\widehat{K}_{j-1}} \frac{1}{\widehat{K}_{j+1} - \widehat{K}_j}$$

Denote $\mathbb{M}(T_{i+1},T_i,\widehat{\sigma}(T_i,.)) = (I - \mathbb{S}\mathbb{D}(T_{i+1}-T_i))$. We can see that $\mathbb{M}(T_{i+1},T_i,\widehat{\sigma}(T_i,.))$ is a tri-diagonal matrix parametrized by $\widehat{\sigma}(T_i,.)$, T_{i+1}, and T_i.

Given the price vector at T_i as the input:

$$\left[\widehat{C}(T_i,\widehat{K}_0), \quad \widehat{C}(T_i,\widehat{K}_1), \quad \widehat{C}(T_i,\widehat{K}_2), \quad \cdots \quad \widehat{C}(T_i,\widehat{K}_N)\right]$$

and the matrix $\mathbb{M}(T_{i+1},T_i,\widehat{\sigma}(T_i,.))$, we can compute the price vector at T_{i+1}:

$$\left[\widehat{C}(T_{i+1},\widehat{K}_0), \quad \widehat{C}(T_{i+1},\widehat{K}_1), \quad \widehat{C}(T_{i+1},\widehat{K}_2), \quad \cdots \quad \widehat{C}(T_{i+1},\widehat{K}_N)\right].$$

We want the price vector at T_{i+1} produced by the above LVF to match the price vector we computed using SABR model on T_{i+1}. In other words, we will try to find the $\widehat{\sigma}(T_i,.)$, which has the form as in equation (E.11). Andreasen and Huge [6] suggest one can obtain $\widehat{\sigma}^*(T_i,.)$ by solving

the following non-linear least square problem:

$$\inf_{\widehat{\sigma}(T_i,.)} \sum_{j=0}^{N} \left(\frac{\widehat{C}(T_{i+1},\widehat{K}_j) - \widehat{C}_{SABR}(T_{i+1},\widehat{K}_j)}{Vega_B(T_{i+1},\widehat{K}_j)}\right)^2 \tag{E.16}$$

where we set:

$$\widehat{C}_{SABR}(T_{i+1},\widehat{K}_j) = \frac{\widetilde{C_{SABR}}(T_{i+1},K_j)}{D(t,T_{i+1})F(t,T_{i+1})},$$

$Vega_B(T_{i+1},\widehat{K}_j)$ is the vega computed using SABR model calibrated to market prices at T_{i+1} and $\widetilde{C_{SABR}}(T,K)$ is the arbitrage-free SABR model value we produce in previous section.

Note that for the intial case $T_0 = t$, $\widehat{C}(T_0,\widehat{K}) = max(1-\widehat{K},0)$ is given as the payoff. Given $\widehat{C}(T_0,\widehat{K})$, we firstly solve (E.16) for the $\widehat{\sigma}^*(T_0,.)$. After obtaining $\widehat{\sigma}^*(T_0,.)$, we can then solve the forward system (E.16) to get

$$\begin{bmatrix} \widehat{C}(T_i,\widehat{K}_0) \\ \widehat{C}(T_i,\widehat{K}_1) \\ \widehat{C}(T_i,\widehat{K}_2) \\ \vdots \\ \widehat{C}(T_i,\widehat{K}_N) \end{bmatrix}$$

sequentially for $i = 1,2,\ldots,T_{M-1}$, where M is the number of expiries in the grid.

After we solved for $\widehat{\sigma}^*(T_i,.), i = 0,1,\ldots,T_{M-1}$, for $T \in (T_i,T_{i+1}]$, we can fill in the gaps by:

$$\begin{bmatrix} \widehat{C}(T,\widehat{K}_0) \\ \widehat{C}(T,\widehat{K}_1) \\ \widehat{C}(T,\widehat{K}_2) \\ \vdots \\ \widehat{C}(T,\widehat{K}_{max}) \end{bmatrix} = \mathbb{M}^{-1}(T,T_i,\widehat{\sigma}^*(T_i,.)) \begin{bmatrix} \widehat{C}(T_i,\widehat{K}_0) \\ \widehat{C}(T_i,\widehat{K}_1) \\ \widehat{C}(T_i,\widehat{K}_2) \\ \vdots \\ \widehat{C}(T_i,\widehat{K}_{max}) \end{bmatrix} \tag{E.17}$$

where $\mathbb{M}(T,T_i,\widehat{\sigma}^*(T_i,.)) = I - \mathbb{SD}(T-T_i)$ is now a tri-diagonal matrix parametrized by $\sigma^*(T_i,.)$, T and T_i. Note that $\sigma^*(T_i,.)$ is known after the calibration (E.16). We then recover the call price by:

$$C(T,K) = \widehat{C}(T,\widehat{K})D(t,T)F(t,T)$$

Interpolation based on the above procedure can guarantee the option prices computed is arbitrage-free. Detailed proofs are can be found in Appendix C.

In this thesis, we only use the above volatility interpolation algorithm to interpolate the option value produced by SABR model for expiry $T \in (T_i,T_{i+1})$, where T_i and T_{i+1} are expiries listed in the market. A natural question the reader may ask is that why we cannot apply above algorithm with purely market prices? In other words, we solve the below problem instead of problem

(E.16):

$$\inf_{\widehat{\sigma}(T_i,.)} \sum_j \left(\frac{\widehat{C}(T_{i+1},\widehat{K}_j) - \widehat{C}_{mkt}(T_{i+1},\widehat{K}_j)}{Vega_B(T_{i+1},\widehat{K}_j)}\right)^2 .$$

In reality, it is hard to find a grid of normalized strike: $0 = \widehat{K}_0 < \widehat{K}_1 < \cdots < \widehat{K}_N$, for which, $\widehat{C}_{mkt}(T_i,\widehat{K}_j)$, $i = 0,1,\ldots,N$ all exists. Especially, if we want our grid of strike to cover both in-the-money option and out-of-the-money option. That is why we use SABR model which can produce option value for any K for an expiry T_i in market after the calibration. In this way, we can use any grid of strikes as we want.

Following the discussion in section E.3, we summarize the SABR smile calibration and the corresponding fix for the butterfly arbitrage and the calendar arbitrage in Algorithm 2 and Algorithm 3. We summarize the LVF volatility interpolation in Algorithm 4. With the help from SABR model and LVF volatility interpolation, we essentially obtain a parametrization of the option value at each trading date t: $\{V^t_{model}(T,K)\}_{T,K}$. Here the expiry T can be any value that is later than t and before the maximum expiry $T^{mkt}_{t,max}$ observed in the market on date t. The strike K price can be any value. We summarize the process of constructing the arbitrage-free options values for a given grid of strikes and a given grid of expiries in Algorithm 5.

E.3.0.5 Construction of Training, Testing and Validation Data Sets

We provide more details on training, testing and validation data sets.

Step 1: We test on the real market expiries. The set of all testing expiries is defined as:

$$\mathbf{T}_{AllTest} = \{T^{mkt} | \text{2000-01-01} \leq T^{mkt} \leq \text{2015-08-31}, T^{mkt} \text{ is a market expiry date } \}$$

Namely, we test on all the market observed expiry date T, which are between 2000-01-01 to 2015-08-31. Note that, we have included two crisis periods: the burst of dot-com bubble period (2000 to 2002) and subprime mortgage crisis period (2007 to 2008). We assume we are on the sell-side of the option trading.

Step 2: For a testing expiry date $T^{test} \in \mathbf{T}_{AllTest}$, we construct the testing set below:

$$TestSet = \{Scenario(T^{test},K) | \forall K \in \mathbf{K}^{mkt}_{grid}(t_0,T^{test})\}$$

where $\mathbf{K}^{mkt}_{grid}(t_0,T^{test})$ is the grid of market strikes for expiry T^{test} that can be observed directly from market on the initial date t_0. And t_0 is 100 business away from T^{test}. In other word, the total hedging horizon is $N_H = 100$ business days. We build a sequence of models to hedge those testing scenarios with the same expiry date T^{test} and different strikes $K \in \mathbf{K}^{mkt}_{grid}(t_0,T^{test})$. The testing scenario: $Scenario(T^{test},K)$ is constructed using procedure described in Algorithm 7.

Algorithm 2: Function For SABR Calibration

Function `SABRCalibration(`t, T, $\mathbf{K}_{grid}$`)`**:**

Input: t: An option trading date t.
T: A **market** option expiry.
$\mathbf{K}_{grid} = \{K_0, K_1, \ldots, K_N\}$: A grid of strikes for outputting option value.

Extract the set of strikes available in market at time t: $\mathbf{K}^{mkt}(t,T)$

Extract the set of market option prices for expiry T at time t:$\{V^{mkt}_{t,T,K} | \forall K \in \mathbf{K}^{mkt}(t,T)\}$

Set $\beta^* = 1$

Solve:

$$\alpha^*, \nu^*, \rho^* = \text{argmin}_{\alpha,\nu,\rho} \sum_{K \in \mathbf{K}^{mkt}(t,T)} \left(V_{SABR}(S_t, t, T, K, r, q; \alpha, \beta^*, \nu, \rho) - V^{mkt}_{t,T,K} \right)^2$$

Calculate the call option prices $C_{SABR}(T,K)$ for $K \in \mathbf{K}_{grid}$ using the SABR parameters $\beta^*, \alpha^*, \nu^*, \rho^*$.

Check if there is any violation of the following condition on the grid of strikes $\mathbf{K}_{grid}$ with $i = 1, \ldots, N-1$:

$$C_{SABR}(T,K_{i-1}) - C_{SABR}(T,K_i) > \frac{K_i - K_{i-1}}{K_{i+1} - K_i} \left(C_{SABR}(T,K_i) - C_{SABR}(T,K_{i+1}) \right) \quad \text{(E.18)}$$

if *No violation* **then**
    Set $K_L = K_0$, $K_L = K_N$
else
    Set K_L to be the smallest K_i where condition (E.18) hold
    Set K_U to be the largest K_i where condition (E.18) hold
end

Calculate call option value function with no-butterfly arbitrage:

$$\overline{C}_{SABR}(T,K) \leftarrow C_{\hat{g}(x)}(T,K) = \begin{cases} C_{BS}(T,K;\sigma_B(K_L)) & \text{if } 0 < K < K_L \\ C_{SABR}(T,K) & \text{if } K_L \le K \le K_U \\ C_{BS}(T,K;\sigma_B(K_U)) & \text{if } K > K_U \end{cases}$$

`/*` $\sigma_B(K)$ `is the short form of the SABR implied volatility approximation (2.2.11).` `*/`

return $\overline{C}_{SABR}(T,K)$

Algorithm 3: Function For Returning Arbitrage Free Option Value With SABR Model

Function `SABRArbitrageFree(` t, $\mathbf{T}_{mkt}$,$\mathbf{K}_{grid}$`):`

Input: t: An option trading date t.
$\mathbf{T}_t^{mkt} = \{T_0 = t, \ldots, T_M\}$: A grid of **market** available expiries at time t.
$\mathbf{K}_{grid} = \{K_0, K_1, \ldots, K_N\}$: A grid of strikes for outputting option value.
`SABRCalibration`: SABR Calibration Algorithm as in Algorithm 2.

Set $\widetilde{C_{SABR}}(T_0, K) = \max(S_t - K, 0)$

`/* Option value with expiry` $T_0 = t$ `is the payoff */`

for $i = 1;\ i \leq M;\ i = i+1$ **do**

  $\overline{C}_{SABR}(T_i, K) \leftarrow$ `SABRCalibration(`t,T_i,$\mathbf{K}_{grid}$`)`

end

for $i = 1;\ i < M;\ i = i+1$ **do**

  $shift_{T_i} = -min\left\{min_{K \in \mathbf{K}_{grid}}\left[\overline{C}_{SABR}(T_i, K) - \overline{C}_{SABR}(T_{i-1}, K)\right], 0\right\}$.

  if $shift_{T_i} \neq 0$ **then**

    for $j = i;\ j < M;\ j = j+1$ **do**

      `/* We need to shift every` $T_j \geq T_i$`, if shift is not zero */`

      $\overline{C}_{SABR}(T_j, K) \leftarrow \overline{C}_{SABR}(T_j, K) + shift_{T_i}$

    end

    $\widetilde{C_{SABR}}(T_i, K) \leftarrow \overline{C}_{SABR}(T_i, K)$

  else

    $\widetilde{C_{SABR}}(T_i, K) \leftarrow \overline{C}_{SABR}(T_i, K)$

  end

end

return $\{\widetilde{C_{SABR}}(T, K) | \forall T \in \mathbf{T}_{mkt}\ ,\ \forall K \in \mathbf{K}_{grid}\}$

Algorithm 4: Function For LVF Calibration

Function `LVFCalibration(` $t, \mathbf{T}_t^{mkt}, \mathbf{K}_{grid}, \mathbf{T}_{grid}$`):`

Input: t: An option trading date t.
$\mathbf{T}_t^{mkt} = \{T_0 = t, \ldots, T_M\}$: A grid of **market** available expiries at time t.
$\mathbf{K}_{grid} = \{K_0, K_1, \ldots, K_N\}$: A grid of strikes for outputting option value.
$\mathbf{T}_{grid} = \{T_0^B = t, \ldots, T_{N_B}^B\}$: A grid of expiries for outputting option value. This grid includes every business days between $T_0 = t$ and T_M.
`SABRArbitrageFree`: SABR Surface Algorithm as in Algorithm 3.

$\{\widetilde{C_{SABR}}(T,K) | \forall T \in \mathbf{T}_t^{mkt}\ ,\ \forall K \in \mathbf{K}_{grid}\} \leftarrow$ `SABRArbitrageFree(`$t, \mathbf{T}_t^{mkt}, \mathbf{K}_{grid}$`)`

Calculate

$$\widehat{C}_{SABR}(T,\widehat{K}) = \frac{\widetilde{C_{SABR}}(T,K)}{D(t,T)F(t,T)}, \widehat{K} = \frac{K}{F(t,T)}$$
$$D(t,T) = e^{-r(T-t)}, F(t,T) = S_t e^{(r-q)(T-t)}$$

' Set $\widehat{C}(T_0,\widehat{K}) = max(1-\widehat{K},0)$

`/* The normalized option value with expiry` $T_0 = t$ `is the normalized payoff */`

for $i = 0;\ i < M;\ i = i+1$ **do**
  Obtain the local volatility function $\widehat{\sigma}^*(T_i,K)$ by solving problem (E.16)
  Construct $\mathbb{S}$ parametrized by $\widehat{\sigma}^*(T_i,K)$ as in equation (E.14)
  Construct $\mathbb{D}$ as in equation (E.15)
  for $j = 1;\ j < N_B;\ j = j+1$ **do**
    if $T_j^B \in (T_i, T_{i+1}]$ **then**
      Construct $\mathbb{M}(T_j^B, T_i, \widehat{\sigma}^*(T_i,.)) = (I - \mathbb{S}\mathbb{D}(T_j^B - T_i))$
      Compute

$$\begin{bmatrix} \widehat{C}(T_j^B,\widehat{K}_0) \\ \widehat{C}(T_j^B,\widehat{K}_1) \\ \widehat{C}(T_j^B,\widehat{K}_2) \\ \vdots \\ \widehat{C}(T_j^B,\widehat{K}_N) \end{bmatrix} = \mathbb{M}^{-1}(T_j^B, T_i, \widehat{\sigma}^*(T_i,.)) \begin{bmatrix} \widehat{C}(T_i,\widehat{K}_0) \\ \widehat{C}(T_i,\widehat{K}_1) \\ \widehat{C}(T_i,\widehat{K}_2) \\ \vdots \\ \widehat{C}(T_i,\widehat{K}_N) \end{bmatrix}$$

    end
  end
end
return $\{\widehat{C}(T,\widehat{K}) | \forall T \in \mathbf{T}_{grid}\ ,\ \forall K \in \mathbf{K}_{grid}\}$

Algorithm 5: Function For Arbitrage Free Surface Construction

Function `ArbitrageFreeSurface(` t, $\mathbf{K}_{grid}$,$\mathbf{T}_{mkt}$,$optType$`)`**:**

Input: t: An option trading date t.
$\mathbf{K}_{grid} = \{K_0, K_1, \ldots, K_N\}$: A grid of strikes for outputting option value.
$\mathbf{T}_t^{mkt} = \{T_0 = t, \ldots, T_M\}$: A grid of **market** available expiries at time t.
optType: Call or Put Option.
`LVFCalibration`: The volatility Interpolation function as in Algorithm 4.

Extract $\mathbf{T}_{grid} = \{T_0^B = t, \ldots, T_{N_B}^B = T_M\}$: This grid includes every business days between t and T_M. The T_M is the maximum market expiry date in $\mathbf{T}_t^{mkt}$.

$\{\widehat{C}(T,\widehat{K})|\forall T \in \mathbf{T}_{grid}\ ,\ \forall K \in \mathbf{K}_{grid}\} \leftarrow$ `LVFCalibration(`t,$\mathbf{T}_t^{mkt}$,$\mathbf{K}_{grid}$,$\mathbf{T}_{grid}$`)`

`/* Construct Put opton value using call-put parity */`

$$C_{model}^t(T,K) = \widehat{C}(T,\widehat{K})D(t,T)F(t,T), D(t,T) = e^{-r(T-t)}, F(t,T) = S_t e^{(r-q)(T-t)}$$
$$P_{model}^t(T,K) = C_{model}^t(T,K) - S_t + KD(t,T)$$

if *optType*=*Call* **then**

$$V_{model}^t(T,K) = C_{model}^t(T,K)$$

else

$$V_{model}^t(T,K) = P_{model}^t(T,K)$$

end

return $\{V_{model}^t(T,K)|\forall T \in \mathbf{T}_{grid}\ ,\ \forall K \in \mathbf{K}_{grid}\}$

Step 3: On the initial date t_0, we prepare the training set as:

$$TrainSet = \{Scenario(T,K) | \forall K \in \mathbf{K}_{grid}(T - \frac{100}{250}, T), \forall T \in \mathbf{B}(T_{min}, t_0)\}$$

$$\mathbf{B}(T_{min}, t_0) = \{t | t \text{ is a business day}, T_{min} < t < t_0\}$$

where T_{min} is the earliest expiry to be included in the training set and $\mathbf{B}(T_{min}, t_0)$ is the set of all business days t with $T_{min} < t < t_0$. We set T_{min} to be 3 years prior to the intial date of the testing scenarios t_0. The grid of strikes $\mathbf{K}_{grid}(t,T)$ is defined as: $\mathbf{K}_{grid}(t,T) = \{0 = K_0 < K_1 < \cdots < 2 * K^{mkt}_{max}(t,T)\}$ with $K_i - K_{i-1} = 5, i \geq 1$ and $K^{mkt}_{max}(t,T)$ is defined as the maximum of strikes we observed in market between between t and T. In other words, we include all training hedging scenarios for which the expiry dates are before t_0 and later than T_{min}.

The validation set is:

$$ValSet = \{Scenario(T,K) | T = t_0, \forall K \in \mathbf{K}_{grid}(T - \frac{100}{250}, T)\}$$

In other words, we include all training hedging scenarios for which the expiry date is t_0 as the validation set. Note that for training and validation set, we do not require T and K to be observed directly from market. We train the model based on the training sets. We get the hedging position for the testing scenarios from the data-driven model for t_0 only: $\delta^M_{t_0,T,K}$.

Step 4: Similarly, on any rebalancing date $t_j > t_0$, the training set and validation set are:

$$TrainSet = \{Scenario(T,K) | \forall K \in \mathbf{K}_{grid}(T - \frac{100}{250}, T), \forall T \in \mathbf{B}(T_{min}, t_j)\}$$

$$\mathbf{B}(T_{min}, t_j) = \{t | t \text{ is a business day}, T_{min} < t < t_j\}$$

where $\mathbf{B}(T_{min}, t_j)$ is the set of all business days t with $T_{min} < t < t_j$.

$$ValSet = \{Scenario(T,K) | T = t_j, \forall K \in \mathbf{K}_{grid}(T - \frac{100}{250}, T)\}$$

We update the model based on the new training set. We get the hedging position from the data-driven model $\text{GRU}_{\text{TOTAL}}$ for t_j only: $\delta^M_{t_j,T,K}$. Notice that the range for the allowed expiries in the training set is extended because $\mathbf{B}(T_{min}, t_0) \subset \mathbf{B}(T_{min}, t_j)$ when $t_j > t_0$. Therefore, We have included new data into the training data set. We also validate our model based on most recent scenarios expiring on t_j.

Algorithm 6: Function For Constructing Training Hedging Scenarios With Expiry Date T

Function `TrainingScenarioGeneration(`T,*optType*`)`**:**

Input: T: an expiry date for the hedging scenarios.
optType: Call or put option.
`ArbitrageFreeSurface`: The function for surface construction as in Alogrithm 5.

Set $t_0 = T - \frac{100}{250}$: A initial date for setting up the hedging scenarios.

Extract $\mathbf{t}_B = \{t_0, \ldots, t_N = T\}$: the set of business dates between t_0 and T sorted in ascending order.

Extract $K^{mkt}_{max}(t_0, T)$: the maximum of strikes we observed in market between between t_0 and T.

Construct the grid of strikes: $\mathbf{K}_{grid}(t_0, T) = \{0 = K_0 < K_1 < \cdots < 2 * K^{mkt}_{max}(t_0, \widehat{T})\}$ where $K_i - K_{i-1} = 5$

for $t \in \mathbf{t}_B$ **do**

`/* Construct the Surface for each date` $t \in \mathbf{t}_B$ `*/`

$\mathbf{T}^{mkt}_t \leftarrow$ the set of market expiries at t

`ArbitrageFreeSurface(`t, $\mathbf{K}_{grid}(t_0, T)$,$\mathbf{T}^{mkt}_t$,optType)

end

for $t \in \mathbf{t}_B$ **do**

for $K \in \mathbf{K}_{grid}(t_0, T)$ **do**

Extract underlying price S_t directly from market.

Extract option value $V_{t,T,K} = V^t_{model}(T, K)$ on the ArbitrageFreeSurface at t.

Construct a vector of features $\mathbf{y}^{T,K}_t$ based on **model** option value.

end

end

for $K \in \mathbf{K}_{grid}(t_0, T)$ **do**

`/*` $\{S_t | \forall t \in \mathbf{t}_B\}$`:  the time-series of underlying prices.`
$\{V_{t,T,K} | \forall t \in \mathbf{t}_B\}$`:the time-series of option prices for a hedging scenario identified by` (T, K)`.`
$\{\mathbf{y}^{T,K}_t | \forall t \in \mathbf{t}_B\}$`:the time-series of feature vectors for a hedging scenario identified by` (T, K)`.` `*/`

$Scenario(T, K) \leftarrow \{S_t, V_{t,T,K}, \mathbf{y}^{T,K}_t | \forall t \in \mathbf{t}_B\}$

end

return $\{Scenario(T, K) | \forall K \in \mathbf{K}_{grid}(t_0, T)\}$

Algorithm 7: Function For Constructing Testing Hedging Scenarios With Expiry Date T

Function `TestingScenarioGeneration(`T,*optType*`)`**:**

Input: T: A **market** expiry date for the hedging scenarios.
optType: Call or put option.
`ArbitrageFreeSurface`: The function for surface construction as in Alogrithm 5.

Set $t_0 = T - \frac{100}{250}$: A initial date for setting up the hedging scenarios.

Extract $\mathbf{t}_B = \{t_0, \ldots, t_N = T\}$: the set of business dates between t_0 and T sorted in ascending order.

Extract all **market** available strikes at t_0 for the expiry T as the grid of strikes: $\mathbf{K}^{mkt}_{grid}(t_0, T) = \{K^{mkt}_{t_0,T,1}, \ldots, K^{mkt}_{t_0,T,N_K}\}$

for $t \in \mathbf{t}_B$ **do**

`/* Construct the Surface for each date` $t \in \mathbf{t}_B$ `*/`

$\mathbf{T}^{mkt}_t \leftarrow$ the set of market expiries at time t

`ArbitrageFreeSurface(`t, $\mathbf{K}^{mkt}_{grid}(t_0, T)$,$\mathbf{T}^{mkt}_t$,optType)

end

for $t \in \mathbf{t}_B$ **do**

for $K \in \mathbf{K}^{mkt}_{grid}(t_0, T)$ **do**

Extract underlying price S_t directly from market.

if $V^{mkt}_{t,T,K}$ *does not exist* **then**

Extract option value as $V_{t,T,K} = V^t_{model}(T, K)$ on the ArbitrageFreeSurface at t.

Construct a vector of fetures $\mathbf{y}^{T,K}_t$ based on **model** option value.

else

Extract option value as $V_{t,T,K} = V^{mkt}_{t,T,K}$.

Construct a vector of features $\mathbf{y}^{T,K}_t$ based on **market** option value.

end

end

end

for $K \in \mathbf{K}_{grid}(t_0, T)$ **do**

`/*` $\{S_t | \forall t \in \mathbf{t}_B\}$`:` `the time-series of underlying prices.`
$\{V_{t,T,K} | \forall t \in \mathbf{t}_B\}$`:the time-series of option prices for a hedging scenario identified by` (T, K)`.`
$\{\mathbf{y}^{T,K}_t | \forall t \in \mathbf{t}_B\}$`:the time-series of feature vectors for a hedging scenario identified by` (T, K)`.` `*/`

$Scenario(T, K) \leftarrow \{S_t,\ V_{t,T,K},\ \mathbf{y}^{T,K}_t | \forall t \in \mathbf{t}_B\}$

end

return $\{Scenario(T, K) | \forall K \in \mathbf{K}^{mkt}_{grid}(t_0, T)\}$

Algorithm 8: Function For Building Model To Hedge Scenarios With the Expiry Date T^{test}

Function `BuildModelForScenarios(` T^{test}, *optType*`)`**:**

Input: T^{test}: A **market** expiry date for the testing hedging scenarios.
optType: Call or put option.
`TestingScenarioGeneration`: The function for generating testing scenarios as in Algorithm 7.
`TrainingScenarioGeneration`: The function for generating training scenarios as in Algorithm 6.

Set $t_0 \leftarrow T^{test} - \frac{100}{250}$

Extract $\mathbf{t}_{RB} = \{t_0, \ldots, t_j, \ldots, t_{N_{rb}-1}\}$: the set of **rebalancing** dates sorted in ascending order. Each rebalancing time is $t_j = t_0 + j \times \Delta t$ and Δt is the gap between the adjacent rebalancing dates.

Extract all market available strikes at t_0 for the expiry T^{test} as the grid of strikes:$\mathbf{K}^{mkt}_{grid}(t_0, T^{test})$

TestSet $\leftarrow$ `TestingScenarioGeneration`$(T^{test}, \text{optType})$

Set $T_{min} \leftarrow t_0 - 3$: T_{min} is 3 years prior to t_0.

for $j = 0, j < N_{rb}, j = j+1$ **do**

  TrainSet $\leftarrow \emptyset$

  Extract $\mathbf{B}(T_{min}, t_j)$: the set of all business days between T_{min} and t_j.

  for $T \in \mathbf{B}(T_{min}, t_j)$ **do**

    TrainSubSet$\leftarrow$ `TrainingScenarioGeneration`$(T, \text{optType})$

    TrainSet $\leftarrow$ TrainSet $\cup$ TrainSubSet

  end

  ValSet$\leftarrow$ `TrainingScenarioGeneration`$(t_j, \text{optType})$

  Bulid model based on TrainSet

  Validate model based on ValSet

  Compute the hedging position at t_j for TestSet:
  $\{\delta^M_{t_j,T,K} | \forall K \in \mathbf{K}^{mkt}_{grid}(t_0, T), T = T^{test}\}$

end

return $\{\delta^M_{t,T,K} | \forall t \in \mathbf{t}_{RB}, \forall K \in \mathbf{K}^{mkt}_{grid}(t_0, T), T = T^{test}\}$