Option Pricing, Hedging, and Risk Analysis under Rough Volatility Models

Cameron B. Sahirad *
(Advised by: Mehdi Sonthonnax, Alireza Javaheri)
New York University, The Courant Institute of Mathematical Sciences

May 19, 2021

1 Introduction

Recent research has shown that the logarithm of volatility is well modeled using a fractional Brownian motion process with Hurst parameter of 0.1 and has led to the development of so called rough volatility models [1]. The paths of fractional Brownian motion are rougher than that of standard Brownian motion and are more consistent with realized time series data [2]. It is generally believed that in equity markets the overall shape of the implied volatility surface does not change but rather the level and orientation of the surface thus suggesting the surface to be modeled as a time homogenous process [1].

Naturally it is desirable to use some type of stochastic volatility model when pricing derivatives as opposed to the simple Black-Sholes model where volatility is assumed to be constant. Using a stochastic volatility model allows for the modelling of the implied volatility smile. However, most stochastic volatility models are not able to capture the term structure of at-the-money volatility skew, given by eq. (1), particularly close to maturity.

$$\psi(\tau) = \left| \frac{\partial}{\partial k} \sigma_{BS}(k, \tau) \right|_{k=0} \tag{1}$$

Some models such as the exponential Levy process introduce jumps into the stock price process to overcome this. The term structure of ATM volatility skew is empirically observed as proportional to $1/\tau^{H-1/2}$, where H is the Hurst parameter, for a wide range of maturities. The rough Bergomi (referred to as rBergomi) model, first introduced in [1], is a stochastic volatility model that implements fractional Brownian motion in order to achieve this effect of modeling $\psi(\tau)$. The rBergomi model is still arbitrage free because fractional Brownian motion is not used on modeling the return of the stock but that of the instantaneous volatility. I begin by building a simulation framework capable

*Department of Mathematics in Finance, Courant Institute of Mathematical Sciences, New York University, New York, NY. E-mail: cbs476@nyu.edu

of executing Monte Carlo simulation to generate sample paths of the stock price and variance processes. This implementation will utilize the hybrid scheme of [3] to generate fractional Brownian motion and will also be capable of setting the full initial forward variance curve. I then consider the problem of calibration and train a neural network to represent the pricing map from model parameters to implied volatilities and analyze the accuracy. Next I perform a sensitivity analysis of the model parameters to gain insight into what effect each has on changing the volatility surface. Finally, I perform a vega risk analysis of several exotic options by hedging implied volatility dynamics with a static hedge of vanilla calls.

2 Rough Bergomi Model

The rBergomi pricing framework of [1] is a non-Markovian generalization of the Bergomi variance curve model and is defined by the following set of equations. The stock price process, S_t, and variance process, v_t, are:

$$dS_t = \sqrt{v_t} S_t dB_t \tag{2}$$

$$v_t = \xi_0(t) \exp\left(\eta W_t^\alpha - \frac{\eta^2}{2} t^{2\alpha+1}\right) \tag{3}$$

Where W^α is a Volterra process, which can be thought of as a fractional Brownian motion conditioned to the present, defined as follows with $\alpha = H - 1/2$ where H is the Hurst parameter $0 < H < 1/2$. For modeling purposes changing the lower bound of the below integral from $-\infty$ as would be the case for pure fractional Brownian Motion to 0 is not a problem as what matters the most is the roughness of the sample paths which is obtained because of the power kernel of the integrand in the stochastic integral.

$$W_t^\alpha = \sqrt{2\alpha + 1} \int_0^t (t-u)^\alpha dW_u^1 \tag{4}$$

ξ_0 is the initial forward variance curve obtained from market data on variance swaps.

$$\xi_0(t) = \mathbb{E}[v_t | \mathcal{F}_0] \tag{5}$$

B and W_u^1 are correlated standard Brownian motions with correlation ρ, $-1 \leq \rho \leq 0$. B can be decomposed into two independent standard Brownian motions W_u^1 and W_u^2 as the following:

$$dB_t = \rho W_t^1 + \sqrt{1 - \rho^2} W_t^2 \tag{6}$$

The rBergomi model is therefore parameterized by four parameters: α, η, ρ, and ξ_0 which can be a full vector itself.

2.1 Hybrid scheme for Brownian semistationary processes

One of the main challenges in simulating the rBergomi model is generating the fractional Brownian motion. The first way this can be done, although computationally slow, is to consider the covariance function of W^H expressed as

$$\mathbb{E}[W^H(t)W^H(s)] = \frac{1}{2}(|t|^{2H} + |s|^{2H} - |t-s|^{2H})$$

and use the method of Cholesky decomposition. For a grid of size n, let Γ be the covariance matrix where $\Gamma_{i,j} = \mathbb{E}[W^H(t_i)W^H(t_j)]$ for $i, j = 1, ..., n$ and let $\Sigma\Sigma^T = \Gamma$. A sample path, u, of the fractional Brownian motion can be generated by multiplying $u = \Sigma v$ where v is a vector of n independent samples from a standard normal distribution. This method has complexity of order $O(n^3)$ and is therefore too slow except for small simulations. Instead, a method outlined in [3] is employed and outlined as follows. The method to represent fractional Brownian Motion as an integral with respect to Brownian Motion was introduced by Mandelbrot & van Ness (1968). A Brownian semistationary process is defined by the following integral representation

$$X(t) = \int_{-\infty}^{t} g(t-s)\sigma(s)dW(s)$$

When the kernel function g is a power-law

$$g(x) \propto x^{\alpha}$$

trajectories of X are fractional Brownian motion with Hurst parameter $H = \alpha + \frac{1}{2}$. This method will proceed by discretizing the kernel function as a combination of two pieces, a power function near zero and a series of step functions everywhere else, hence the name hybrid scheme. First $X(t)$ is discretized as follows

$$X(t) = \sum_{k=1}^{\infty} \int_{t-\frac{k}{n}}^{t-\frac{k}{n}+\frac{1}{n}} g(t-s)\sigma(s)dW(s) \approx \sum_{k=1}^{\infty} \sigma(t-\frac{k}{n}) \int_{t-\frac{k}{n}}^{t-\frac{k}{n}+\frac{1}{n}} g(t-s)dW(s)$$

If k is small then the kernel function g can be written as,

$$g(t-s) \approx (t-s)^{\alpha} L_g\left(\frac{k}{n}\right)$$

where L_g is a slow varying measurable function near zero and bounded away from 0 $L : (0, 1] \to [0, \infty)$. If k is large then the kernel function g can be written as,

$$g(t-s) \approx g\left(\frac{b_k}{n}\right)$$

with $b_k \in [k-1, k]$ as the sequence of evaluation points. Using the first representation for g for the first κ terms and the second representation for the remaining terms, $X(t)$ can now be written as,

$$X(t) \approx \sum_{k=1}^{\kappa} L_g\left(\frac{k}{n}\right) \sigma(t-\frac{k}{n}) \int_{t-\frac{k}{n}}^{t-\frac{k}{n}+\frac{1}{n}} (t-s)^{\alpha} dW(s) + \sum_{k=\kappa+1}^{\infty} g\left(\frac{b_k}{n}\right) \sigma(t-\frac{k}{n}) \int_{t-\frac{k}{n}}^{t-\frac{k}{n}+\frac{1}{n}} dW(s)$$

Hence, $X(t)$ is comprised of two components

$$X_n(t) = \check{X}_n(t) + \hat{X}_n(t)$$

where

$$\check{X}_n(t) = \sum_{k=1}^{\kappa} L_g\left(\frac{k}{n}\right)\sigma\left(t-\frac{k}{n}\right)\int_{t-\frac{k}{n}}^{t-\frac{k}{n}+\frac{1}{n}}(t-s)^{\alpha}dW(s)$$

$$\hat{X}_n(t) = \sum_{k=\kappa+1}^{N_n} g\left(\frac{b_k}{n}\right)\sigma\left(t-\frac{k}{n}\right)\left(W\left(t-\frac{k}{n}+\frac{1}{n}\right)-W\left(t-\frac{k}{n}\right)\right)$$

Now, to be of use within the rBergomi model it is necessary to consider a similar process to $X(t)$ but obtained by truncating the stochastic integral at 0 known as a truncated Brownian semistationary process defined as

$$Y(t) = \int_0^t g(t-s)\sigma(s)dW(s)$$

The hybrid scheme for discretizing Y is similar but with changing the upper bound of the sums to correspond to only non-negative values.

$$Y_n(t) = \check{Y}_n(t) + \hat{Y}_n(t)$$

$$\check{Y}_n(t) = \sum_{k=1}^{\min(\lfloor nt\rfloor,\kappa)} L_g\left(\frac{k}{n}\right)\sigma\left(t-\frac{k}{n}\right)\int_{t-\frac{k}{n}}^{t-\frac{k}{n}+\frac{1}{n}}(t-s)^{\alpha}dW(s)$$

$$\hat{Y}_n(t) = \sum_{k=\kappa+1}^{\lfloor nt\rfloor} g\left(\frac{b_k}{n}\right)\sigma\left(t-\frac{k}{n}\right)\left(W\left(t-\frac{k}{n}+\frac{1}{n}\right)-W\left(t-\frac{k}{n}\right)\right)$$

Finally, in order to practically simulate $Y_n(t)$ to be used in generating sample paths of the rBergomi price process, the following computation is performed with $\kappa = 1$,

$$Y_n\left(\frac{i}{n}\right) = \sum_{k=1}^{\min(i,\kappa)} L_g\left(\frac{k}{n}\right)W^n_{i-k,k} + \sum_{k=\kappa+1}^{i} g\left(\frac{b_k}{n}\right)W^n_{i-k}$$

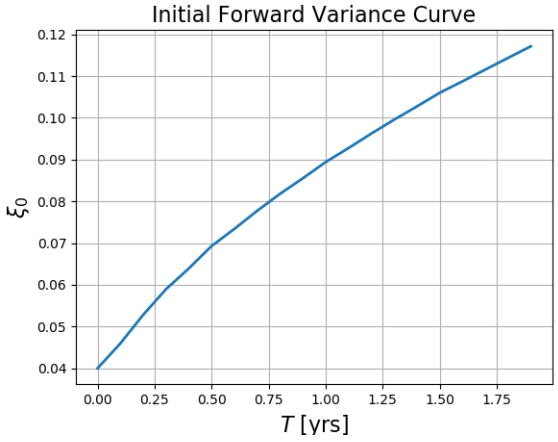
with random vectors $\{W_i^n\}_{i=0}^{\lfloor nT\rfloor-1}$.

2.2 Initial Forward Variance Curve

The initial forward variance curve, ξ_0, in the rBergomi model must be extracted from market data such as the price of variance swaps. Forward variance is the expectation under the pricing measure of future instantaneous variance [1]. By taking the fair strike of variance swaps given by,

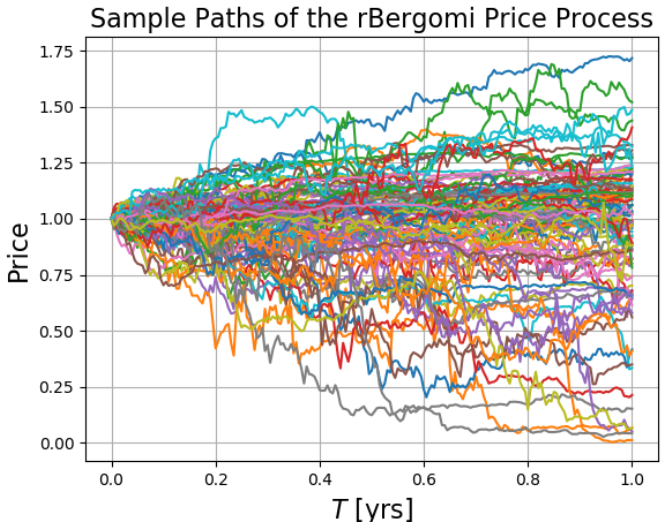
$$\mathcal{V}(T) = \int_t^T \xi_t(u)du \tag{7}$$

at several different maturities, several nodes of the forward variance curve can be computed. Then using some interpolation scheme such as piece-wise constant, log-linear, or log-cubic spline, the entire curve can be computed and used in the rBergomi simulation framework. Plotted below is a sample of what the initial forward variance curve would look like under normal market conditions using log-linear interpolation, that is, upward sloping. During periods of elevated volatility this curve could invert and become downward sloping as the expectation is that the current level of volatility is only transient and will revert back closer to some long-term average in future months.

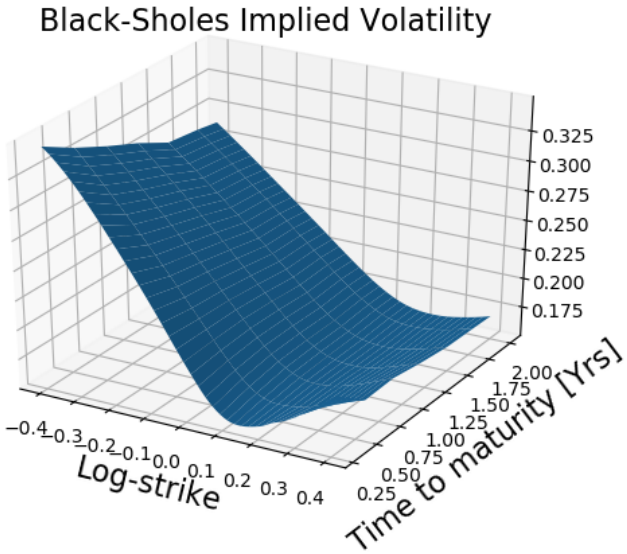


2.3 Monte Carlo Simulation

Due to the lack of Markovianity in the rBergomi model analytical pricing methods such as PDE or Fourier do not apply [4] and thus I rely on Monte Carlo simulation to generate sample paths of the stock price process in order to price contingent claims. A simulation framework for generating sample paths of the stock price and variance processes is implemented in Python from the equations in the section above. Plotted below are sample paths of the price process over 1 year.



Once the price paths have been simulated a volatility surface can be constructed. First 30,000 price paths are simulated and then used to price a grid of vanilla calls for various strikes and maturities. Once the price has been computed, the Black-Sholes formula is inverted to and the Black-Sholes implied volatility is computed for each strike and maturity and a surface is created such as that plotted below.



3 Payoffs

Within the rBergomi simulation framework several types of exotics will be priced and studied for risk analysis purposes. The rBergomi model with the incorporation of fractional Brownian Motion driving the instantaneous volatility is still arbitrage free and

discounted prices of derivatives are still a martingale since they are defined as a discounted payoff. Since the model is non-Markovian now, Ito's formula does not directly apply and therefore there is no "asset" driven by fractional Brownian Motion. These exotics, along with the vanilla options, are outlined below.

3.1 Vanilla

One of the simplest types of contingent claims is the European vanilla call or put. The payoff of such a contract with exercise price K and maturity T is the following:

$$\text{Vanilla call payoff} = \max(S_T - K, 0)$$
$$\text{Vanilla put payoff} = \max(K - S_T, 0)$$

3.2 Asian

In the case of Asian options, the payoff does not depend solely on the terminal value of the asset price but rather on the average price of the asset measured at pre-determined dates. The payoff can be stated as:

$$\text{Asian call payoff} = \max(A_T - K, 0)$$
$$\text{Asian put payoff} = \max(K - A_T, 0)$$

Where $A_T = \frac{1}{n}\sum_{i=1}^{n} S_{t_i}$ There is no closed form solution for Asian options in the Black-Sholes framework, however, a similar variant known as a geometric asian option does have a closed form solution. In a geometric asian option the payoff is $\exp[\frac{1}{T}\int_0^T \ln S_t dt]$. The closed form solution is obtained by taking the solution to the Black-Sholes model and replacing $\sigma_{\text{Asian}} = \sigma/\sqrt{(3)}$ and $q_{\text{Asian}} = \frac{1}{2}\left(r + q + \frac{\sigma^2}{6}\right)$. Computing the closed form solution for an Asian option can be useful when used as a control variate when doing Monte Carlo simulation.

3.3 Barrier

In a barrier option the price of the underlying must never hit (knock-out) or must hit (knock-in) a barrier level, H, before maturity. Barrier options are similar to vanilla options but with the added constraint that the option will not give a payoff until a barrier is reached in the case of knock-in options or will become worthless once a barrier

is reached in the case of knock-out.

$$\text{Up-and-out payoff} = \begin{cases} \max((S_T - K)\phi, 0) & S_t < H \\ 0 & \text{otherwise} \end{cases}$$

$$\text{Up-and-in payoff} = \begin{cases} 0 & S_t < H \\ \max((S_T - K)\phi, 0) & \text{otherwise} \end{cases}$$

$$\text{Down-and-out payoff} = \begin{cases} \max((S_T - K)\phi, 0) & S_t > H \\ 0 & \text{otherwise} \end{cases}$$

$$\text{Down-and-in payoff} = \begin{cases} 0 & S_t < H \\ \max((S_T - K)\phi, 0) & \text{otherwise} \end{cases}$$

$$\text{Where } \phi = \begin{cases} 1 & \text{for calls} \\ -1 & \text{for puts} \end{cases}$$

A barrier option will always be cheaper than the corresponding European option due to the possibility of the option paying off zero in the case the barrier is hit or never hit depending on the type.

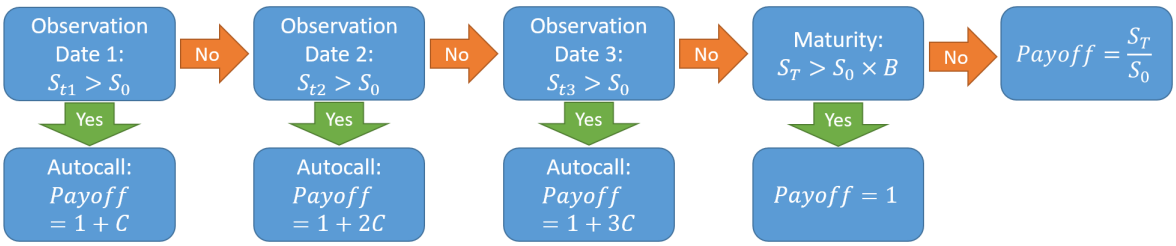
3.4 Cliquet

A cliquet option will payoff the greater of 0 and the sum of a series of fixed interval returns capped at a certain percentage.

$$\text{n-Period cliquet option payoff} = \max\left[\sum_{i=1}^{n} \min\left(\text{cap}, \frac{S_{i/n}}{S_{(i-1)/n}} - 1\right), 0\right]$$

3.5 Autocallable

An autocallable option is a structured product whose payoff is dependent on a reference asset. The autocallable note modeled here combines elements of principal protection, yield enhancement, and some downside risk. The payoff structure is best understood with the following flow chart. If on the first observation date the price of the reference asset S_{t1} is above its initial price, S_0, the option pays a coupon and returns its principal for a payoff of $1 + C$ and the option is terminated. Otherwise, the option remains active and upon the next observation date the same inequality is checked. If the reference asset is above its initial price now the payoff will be a larger coupon. This pattern continues until maturity, when if the reference asset is still not above its initial price but it is above some barrier level then the option will simply return its principal amount. If the asset is below this barrier, the full negative return is exposed for a payoff of S_T/S_0 [5].



4 Calibration

The level of tractability of an option pricing model leads to how prevalent is its use in industry applications. In the past, rough volatility models have largely been excluded due to the calibration bottleneck of needing to use slow Monte Carlo based simulations in order to calibrate a model. Often is the case that accuracy is traded off for speed. Using a neural network solves many of these issues by executing the approximation of the pricing functional to an off-line preprocessing step [6]. The approximate parameter calibration problem can be stated as

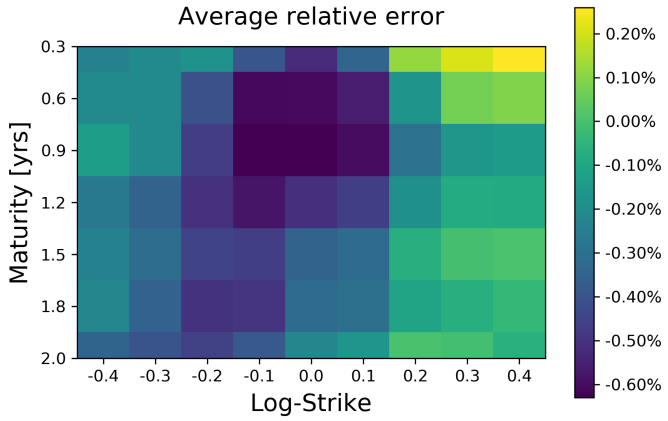
$$\hat{\theta} = \arg\min_{\theta \in \Theta} \delta \left(\tilde{P}(\mathcal{M}(\theta), \zeta), \mathcal{P}^{MKT}(\zeta) \right) \tag{8}$$

Where $\tilde{P}$ represents the price output of the trained neural network, $\mathcal{M}(\theta)$ in this case is the rBergomi model, ζ is the set of contract parameters (strike price for vanillas), $\mathcal{P}^{MKT}$ is the observed market price, and θ is the vector of rBergomi model parameters. The solution using a neural network approach involves splitting the problem into two steps. As performed in [6], the pricing function that maps model parameters to implied volatilities is first learned and stored offline. Secondly, this now deterministic pricing map represented as a neural network is used to perform the online calibration to market data. Any need to numerically simulate stochastic models is now removed from the calibration step as it is embedded into the process of training the neural network pricing map and leads to orders of magnitude of speed-up. Supervised training of the neural network is performed using data generated from the stochastic pricing models. In other words, Monte Carlo simulation is performed for a wide range of model parameters to generate a set of implied volatilities corresponding to model inputs. This dataset is what is used to train the neural network.

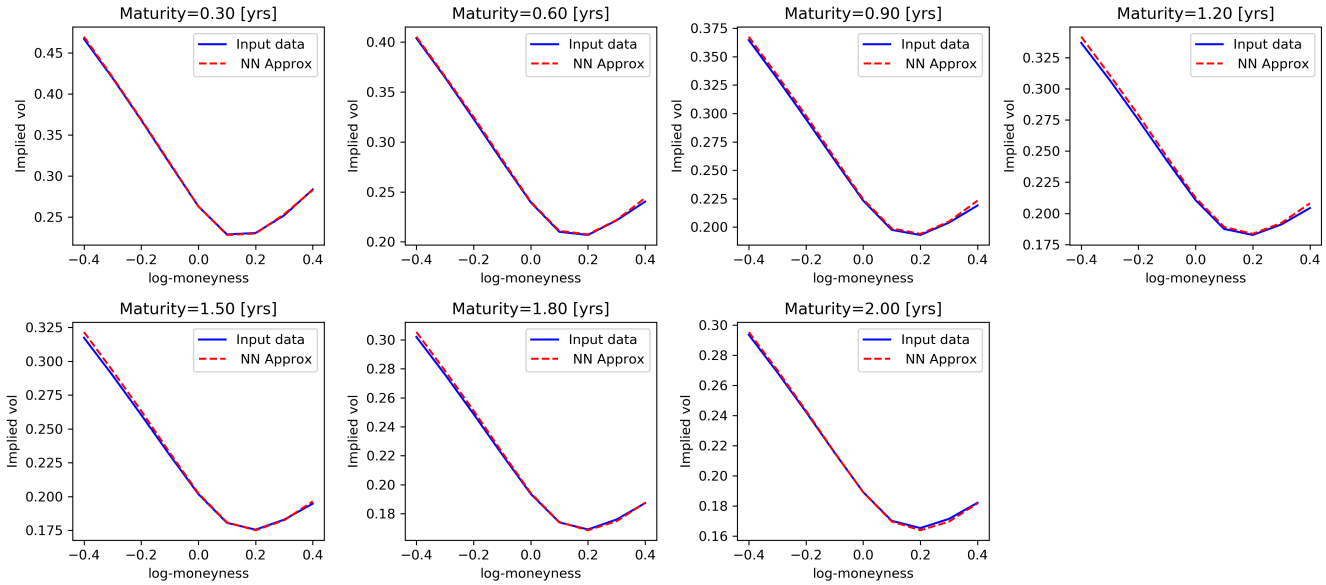
One large advantage to using the two-step approach for calibration rather than using a neural network to directly calibrate to data is that the pricing scheme is still fundamentally model based and therefore benefits from being more robust to unseen conditions. The other main advantage is from a risk management perspective; the model parameters still have a physical interpretation and the neural network is only used as a computational improvement.

To train a neural network to represent the rBergomi model considered in this framework, 20,000 parameter combinations were generated to create 20,000 implied volatility surfaces. The bounds of each parameter were set as $\alpha \in [-0.475, 0]$, $\eta \in [0.5, 4.0]$, $\rho \in [-0.95, -0.1]$, and $\xi_0 \in [0.01, 0.16]$. The network was set up with an input layer to

take these 4 inputs, 3 hidden layers with 30 nodes each, and an output layer to output a grid of 63 implied volatilities across 9 strikes and 7 maturities. The figure below displays the average relative error between the reserved test set of data and the neural network approximation.



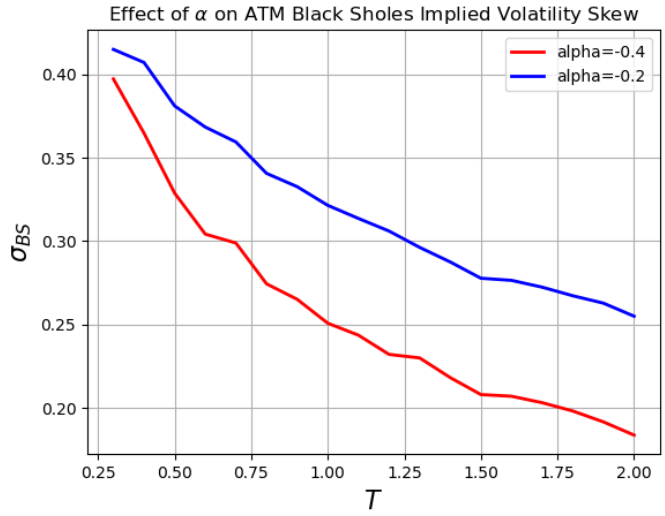
For one choice of θ the plots below compare the performance of the neural network approximation to the input data for several slices of the implied volatility surface.



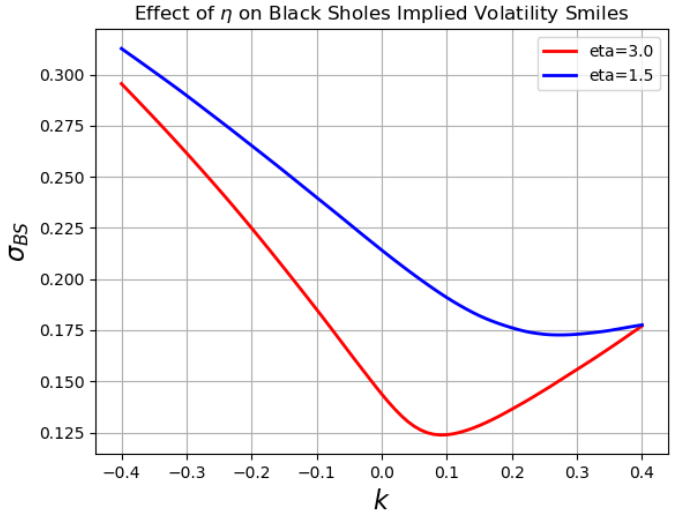
5 Sensitivity Analysis

It is generally accepted that the underlying structure of the implied volatility surface does not change over time but rather shifts, rotates, or bends through time. These changes to the shape of the surface are controlled by the rBergomi model parameters α, η, ρ, and ξ_0.

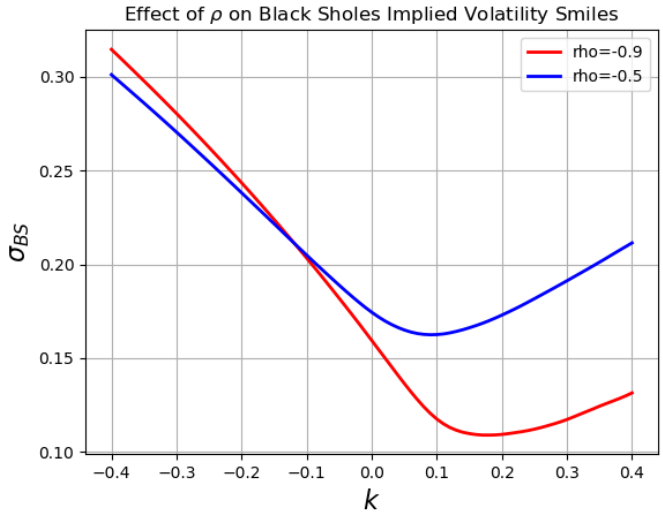
The parameter α has the effect of controlling the term structure of ATM volatility skew. Smaller values of α lead to a steeper term structure and a greater explosion of the skew and smile near maturity. This feature of controlling the term structure of volatility skew in the rBergomi model is not commonly found in other stochastic volatility models.



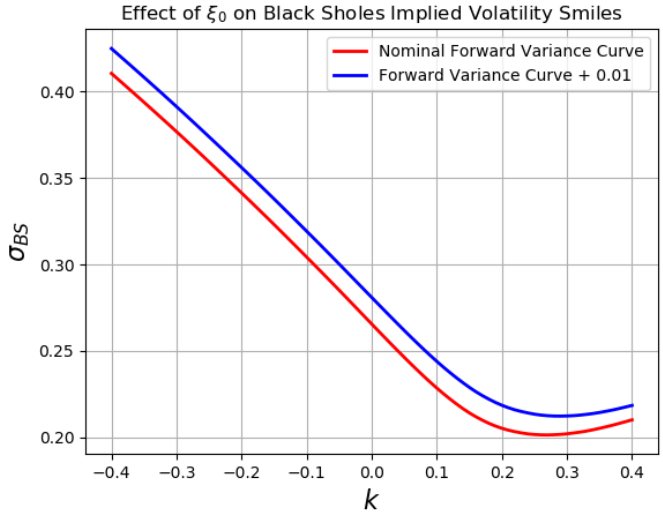
The parameter η has the effect of controlling the degree to which there exists a smile in the volatility vs strike profile. Larger values of η result in a larger smile whereas smaller values result in a shape closer to what is known as a volatility smirk.



The parameter ρ has the effect of controlling the skew of the volatility smile. If $\rho = 0$ then the smile would be symmetric. Since ρ controls the correlation between the driving stochastic terms in the stock price and variance processes it makes intuitive sense that a larger negative correlation leads to a larger skew. Essentially this represents the market phenomenon that price is negatively correlated with volatility - when prices drop volatility tends to increase.



The effect of ξ_0, the initial forward variance curve, has the effect of parallel shifting the volatility surface. Values in the front end of the initial forward variance curve will have more of an effect of close to maturity options whereas the back end of the curve will have a greater effect on longer maturity options.



6 Risk Analysis

6.1 Motivation

It is a known fact that any arbitrary European payoff can theoretically be replicated with a static portfolio of vanilla options when traded along a continuum of strikes. To see this consider the option value

$$f_0 = e^{-rT}\mathbb{E}[f(S_T)] = e^{-rT}\int_0^\infty f(K)h(K)dK$$

Where $h(K)$ is the implied distribution density derived by considering a butterfly spread with strikes $K-\epsilon$, K, $K+\epsilon$ with quantity $1/\epsilon^2$.

$$h(K) = \mathbb{P}[S_T = K] = e^{rT} \lim_{\epsilon \to 0} \frac{c(K-\epsilon) - 2c(K) + c(K+\epsilon)}{\epsilon^2} = e^{rT} \frac{d^2 c}{dK^2}$$

Applying integration by parts we obtain,

$$f_0 = \int_0^\infty f(K) \frac{d^2 c}{dK^2} dK = f(K) \frac{dc}{dK}\Big|_0^\infty - \int_0^\infty f'(K) \frac{dc}{dK} dK = -f(0) \frac{dc}{dK} - \int_0^\infty f'(K) \frac{dc}{dK} dK$$

$$= -f(0) \frac{dc}{dK} + f'(0) c(0) + \int_0^\infty f''(K) c(K) dK$$

$$= f(0) e^{-rT} + f'(0) F e^{-rT} + \int_0^\infty f''(K) c(K) dK$$

This says the payoff can be replicated with a portfolio of zero-coupon bonds, forwards, and the final term in the expression - all vanilla calls at strike K in quantities $f''(K)dK$ [7]. This fact provides a bit of motivation for the next section in which a portfolio of vanilla calls will be created to hedge exotic options to changes in implied volatility.

6.2 Vega Risk Profile of Exotics

When entering net long positions in exotic options, as is common practice on equity derivative desks, it is necessary to understand the vega risk profile of each product due to implied volatility dynamics. To determine the vega risk profile of the exotic options described earlier it is necessary to first form a static hedge portfolio of vanilla calls. In incomplete markets there exist risk factors that exotic options can be particularly sensitive to such as volatility risk. If the market were complete, then static derivative hedges to not contribute any benefit over dynamic hedging [8]. When incorporating static hedges, the instruments used such as vanilla calls will be exposed to the same risk factors as the exotics and therefore prove to be a powerful tool to hedge these risks. The problem of hedging can be expressed as

$$E = Hw \tag{9}$$

where E is a vector of prices of exotics, H is a price matrix of hedging instruments, and w is a vector of weights for each hedging instrument. Each exotic option in E has the same contract parameters (strike and barrier in the case of a barrier option) but each price is computed using a different perturbation to the rBergomi model parameters α, η, ρ, and ξ_0. These perturbations represent different shifts in the implied volatility surface. To hedge these shifts a portfolio of vanilla calls, H, is constructed. H will have number of columns equal to the number of strikes used for hedging and will be priced under the same perturbation scenarios as each element in E. Therefore the number of rows in H, and E, will be equal to the number of different perturbation scenarios. The idea

is that there exists an optimal w such that when a portfolio of vanilla calls at various strikes is purchased in quantities w_i that the change in price of the exotic option under consideration due to implied volatility dynamics is captured by the change in price due to the portfolio of vanilla calls. To solve for w the Ridge regression algorithm is applied such that

$$w = \arg\min_{w}\{||E - Hw||^2 + \lambda||w||^2\} \tag{10}$$

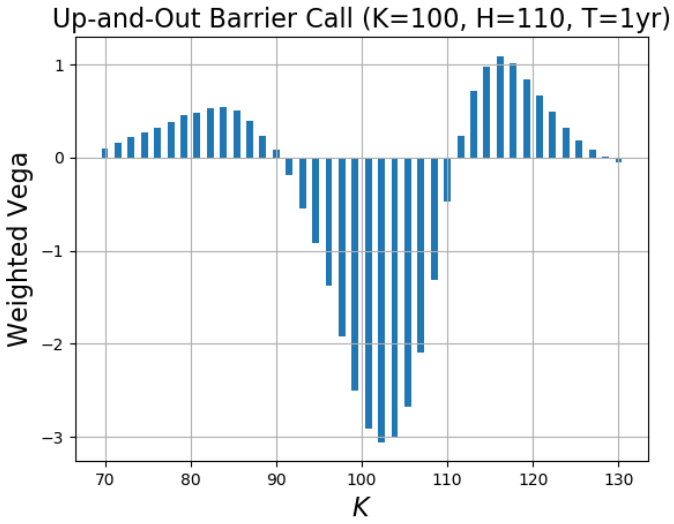
To construct E for each type of exotic option, eight perturbation scenarios were considered to each represent a different shock to the implied volatility surface. These eight scenarios are created by perturbing $\alpha' = \alpha \pm 0.1$, $\eta' = \eta \pm 0.5$, $\rho' = \rho \pm 0.2$, and $\xi' = \xi \pm 0.01$. First, an up-and-out barrier call is considered with strike \$100, barrier at \$110, and 1 year to maturity. H is constructed by pricing vanilla calls with the same perturbations in strikes ranging from \$70 to \$130. Second, a cliquet option with monthly payout frequency, 5% cap, and 1 year to maturity is considered and H also contains strikes ranging from \$70 to \$130. Next, an Asian call with strike \$100 and 1 year to maturity is considered with hedging strikes ranging from \$85 to \$115. Finally, the autocallable is considered with quarterly observation frequency, 1 year to maturity, 5% coupon and 70% barrier. In the case of the autocallable the hedging strikes range from \$35 to \$165. The barrier at 70% requires that hedging strikes at least as low as \$70 are considered.

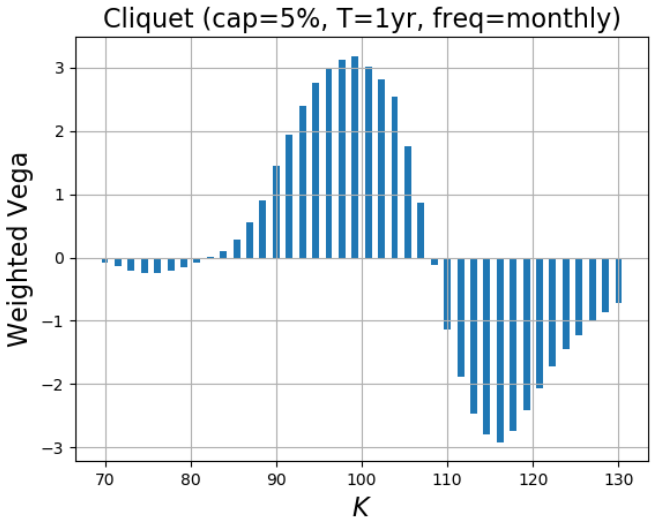
Once w has been computed, it is then weighted by the Black-Sholes vega of each vanilla call in H.

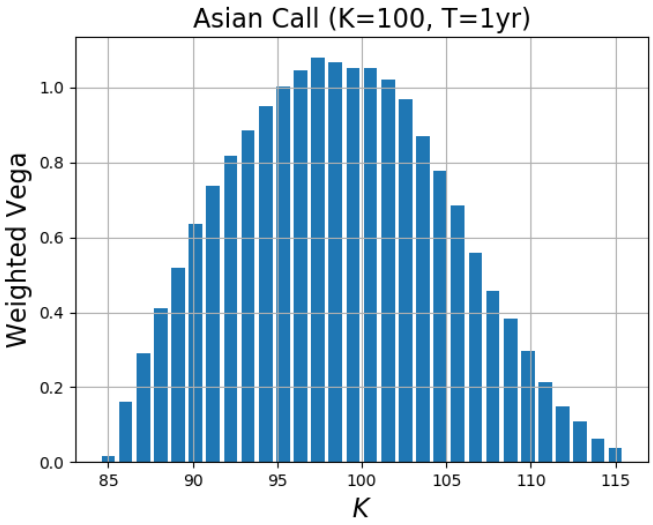
$$\nu_{BS} = SN'(d_1)\sqrt{T-t} \tag{11}$$

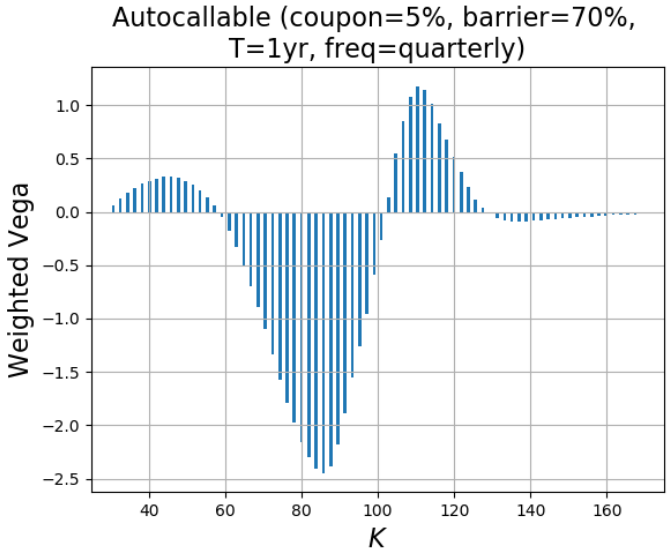
$$d_1 = \frac{1}{\sigma_{BS}\sqrt{T-t}}\left[\ln\left(\frac{S_t}{K}\right) + \left(r + \frac{\sigma^2}{2}\right)(T-t)\right]$$

Plotted below are the vega profiles of each exotic described. The profiles of the autocallable and barrier are similar as an autocallable can roughly be thought of as a combination of barrier calls.




Cliquet (cap=5%, T=1yr, freq=monthly)
Weighted Vega
K


Asian Call (K=100, T=1yr)
Weighted Vega
K


Autocallable (coupon=5%, barrier=70%,
T=1yr, freq=quarterly)
Weighted Vega
K

7 Conclusion

The rBergomi model is a rough stochastic volatility model that fits extremely well to observed volatility surfaces. I have demonstrated the approach to building a simulation framework to analyze this model in Python capable of generating sample paths of the stock price and variance processes while taking advantage of known computational speed-ups. This framework could be further improved by implementing certain variance reduction techniques such as implementing a control variate as is done in [4]. I have successfully trained a neural network to represent the pricing function of the model which can be used for calibration purposes. Calibrating in this way solves the intractability of the rBergomi model due to the calibration bottleneck if Monte Carlo simulation were used in calibration. I have performed a sensitivity analysis of the model parameters and finally projected the risk of implied volatility dynamics onto the vega risk profile of each option by solving the problem of static hedging with vanilla calls. Overall, this framework allows for the pricing of exotics in the rBergomi model, calibrating the model to market data, and analyzing the risk of derivatives priced in this framework. Future work could include extending the scope to other rough volatility models such as the rough Heston model.

References

[1] C. Bayer, P. K. Friz, and J. Gatheral (2016): Pricing under rough volatility. Quant. Finance 16(6), 887–904.

[2] J. Gatheral, T. Jaisson and M. Rosenbaum (2014+): Volatility is rough. Quantitative Finance, to appear.

[3] M. Bennedsen, A. Lunde, and M. S. Pakkanen (2017): Hybrid scheme for Brownian semistationary processes. Finance Stoch. 21(4) 931–965.

[4] R. McCrickerd and M. S. Pakkanen (2017): Turbocharging Monte Carlo pricing for the rough Bergomi model. Preprint: http://arxiv.org/abs/1708.02563

[5] Scotiabank Market Structure Commentary. (2018). https://www.gbm.scotiabank.com/content/dam/gbm/market-insights/2018/2019-12-11-Preferred-Notes.pdf.

[6] Horvath, B., Muguruza, A. Tomas, M. 2020. Deep Learning Volatility. https://arxiv.org/abs/1901.09647

[7] Bossu, S. (2014). In Advanced equity derivatives volatility and correlation (pp. 36–37). essay, Wiley.

[8] A. İlhan, M. Jonsson, and R. Sircar. (2008). Optimal static-dynamic hedges for exotic options under convex risk measures. Stochastic Processes and their Applications. 119. 3608-3632. 10.1016/j.spa.2009.06.009.