Calibrating the Hurst Parameter for Rough Volatility Models with Application in the South African Market

Paul Pettit

A dissertation submitted to the Faculty of Commerce, University of Cape Town, in partial fulfilment of the requirements for the degree of Master of Philosophy.

October 10, 2022

MPhil in Mathematical Finance,
University of Cape Town.

Declaration

I declare that this dissertation is my own, unaided work. It is being submitted for the Degree of Master of Philosophy in the University of the Cape Town. It has not been submitted before for any degree or examination in any other University.

October 10, 2022

Abstract

It is known that accurate and efficient calibration of any fractional stochastic volatility model is important for trading and risk management purposes. Under the rough Heston model proposed by El Euch *et al.* (2019), the Hurst parameter governs the roughness of the volatility process. This dissertation explores the different calibration methods used to obtain an estimate for the Hurst parameter, under the scope of the rough Heston model. Three different calibration methods are presented, namely, a Brute Force minimisation procedure, a Neural Network calibration and a Linear Regression procedure. European option prices are simulated from the rough Heston model using the characteristic function pricing approach as in El Euch and Rosenbaum (2019) and numerical techniques, such as the fractional Adams method which are implemented in MATLAB. These simulated prices are then used to test and compare the three proposed calibration methods in terms of accuracy and efficiency. Thereafter, additional experiments are conducted on South African market data from traded options and the fitted models are compared across the calibration methods used. The results of our numerical experiments are used to justify the nature of rough volatility in the South African options market and recommendations are made on the appropriateness of each calibration scheme in practice.

Overall, we find that the performance measured by accuracy on our simulated data of the Neural Network method is similar to the Brute Force minimisation method, whereas the Linear Regression method, is the least accurate. When calibrating on the market data, the results of the fitted models show that both the Neural Network and Brute Force method resembles the market behaviour. All three methods were shown to be suitable in estimating the Hurst parameter and suggesting rough volatility in this South African market.

Acknowledgements

I would like to express a sincere thank you to my supervisor, Andrew Soane, whose guidance, insight and support has been invaluable. I am also deeply grateful for the warm and generous support from my parents who continued to motivate me throughout the challenging pandemic period.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Calibration is a useful technique applied by practitioners in the financial trading market for estimating model parameters using observable market information (Liu *et al.*, 2019). Numerical techniques are used to calibrate a model to observable market information, such as prices or implied volatility surfaces of listed derivatives. Calibration and approximation methods for option prices have been presented in different formats in the past according to Horvath *et al.* (2019) and they have been extensively studied in Mitra (2012), Kilianová and Letko (2018), Funahashi and Kijima (2017), Fukasawa (2017), Livieri *et al.* (2018) and Alòs and Shiraya (2019).

Recently, it has been shown that the observed empirical volatility in financial markets is extremely consistent with rough volatility (Gatheral *et al.*, 2018). This enables more accurate pricing of options. "Rough" volatility uses the work of Mandelbrot and Ness (1968) in defining a stochastic process with rough paths.

Furthermore, fractional Brownian motion (fBm) has a long memory property meaning that past observations influence every future observation in a decaying manner, a dynamic argued to appear in financial markets. The Hurst parameter, H, controls the long (or short) term memory of a fractional Brownian motion, allowing it to deviate from a standard Brownian motion and is denoted by $H \in [0, 1]$. It has been widely applied within different fields as a measure of long range dependence in time series and of long-term non-linearity (Hurst, 1951). Consequently, it can be seen that models containing fBm with Hurst parameter $0 < H < \frac{1}{2}$ resemble empirical volatility in the financial market better than the classical Heston model for modelling the dynamics of asset prices (Gatheral *et al.*, 2018).

Recently, new methods, such as the COS method attributed to Fang and Oosterlee (2009) which is based on a Fourier-Cosine expansion series and modern calibration methods, such as Artificial Neural Networks as in Horvath *et al.* (2019), have been introduced to price and calibrate within rough stochastic volatility models.

This dissertation aims to provide a literature review by contrasting classical with fractional Brownian motion and present a rough stochastic volatility model,

namely, the rough Heston model with its characteristic function. Furthermore, research on various experiments in the literature on the calibration of fractional stochastic volatility models will be presented and discussed. Thereafter, experiments will be conducted to determine the extent to which the rough Heston model matches market behaviour. A focus will be placed on three different calibration methods to be implemented. In particular, a simple sum of squares Brute Force minimization calibration based on El Euch *et al.* (2019), an Artificial Neural Network (ANN) calibration based on Liu *et al.* (2019) and a Linear Regression method as described in Alòs and Shiraya (2019). Finally, numerical results will be obtained using simulated model prices as well as implied volatility surfaces in the South African market in order to compare the performance of the methods. A discussion will then evaluate the three different methods implemented to determine their performance and relevance to the South African market.

In *Section 2*, a literature review of the background information needed for the content of this dissertation is given. Firstly, an overview of classical and fractional Brownian motion is presented. Then, rough volatility models as researched by Gatheral *et al.* (2018), Fukasawa (2017) and Alòs and Shiraya (2019) will be covered. Thereafter, pricing under the rough Heston model with reference to the characteristic function and the fractional Adams method will be presented. Finally, calibration methods such as Brute Force, Neural Networks and Linear Regression experiments to substantiate their effectiveness are discussed from the literature. In *Section 3*, the methodology of the implemented calibration methods is presented. In *Section 4*, numerical experiments and results will be presented firstly on simulated data and thereafter on market data in order to compare the three calibration methods. Finally in *Section 5*, the dissertation is concluded with a summary of the results and future research recommendations.

Chapter 2

Literature Review

Much research has been conducted on efficient and accurate methods of estimating the Hurst parameter in rough volatility models, namely, Mitra (2012), Funahashi and Kijima (2017), Kirichenko *et al.* (2011), Fukasawa (2017), Livieri *et al.* (2018) and Alòs and Shiraya (2019). These involve statistical, empirical and more recently, machine learning methods as described in Horvath *et al.* (2019), Erkan (2020) and Liu *et al.* (2019). Other authors, such as Zeng and Tang (2011) have identified flaws in past methods and suggested improvements.

In particular, Clegg (2006) reminds researchers of certain methodology flaws which should be considered to ensure reliability of results, such as periodicity, trends and other sources of corruption in the data which can influence the estimator. He warns that misleading conclusions could arise if the results of only one single estimator are considered.

2.1 Classical Brownian Motion and Fractional Brownian Motion

In traditional quantitative finance theory, it is common to model underlying asset prices through time with a geometric Brownian motion. This is defined by the stochastic differential equation:

$$dS_t = S_t \mu dt + S_t \sigma dB_t$$

where S_t is the asset price at time t, μ and σ are constants and B_t is a (classical) Brownian motion. In the above, $S_t \mu dt$ is the drift term and $S_t \sigma dB_t$ is the diffusion term. Therefore, the randomness of an asset price can be modelled primarily by the Brownian motion process $(B_t)_{t \geq 0}$.

Classical Brownian motion displays a myriad of useful properties. Firstly, each realisation (sample path) of the Brownian motion is continuous almost surely. If we

split the Brownian motion into incremental steps

$$(B_{t_1} - B_{t_0}), (B_{t_2} - B_{t_1}), ..., (B_{t_n} - B_{t_{n-1}})$$

where $t_0 < t_1 < t_2 < ... < t_n$, then each of these increments are independent. Additionally, these increments are Gaussian i.e. $(B_{t_1} - B_{t_0}) \sim N(0, t_1 - t_0)$. One can also easily construct the Brownian motion to start at any constant c i.e. $B_{t_0} = c$ and evolve the process through time. It has therefore been shown that Brownian motion is a martingale as a result of these properties, i.e. that

$$\mathbb{E}[B_t|B_s] = B_s \quad \text{for } s \leq t$$

and thus is an ideal process for modelling the price of options on an underlying asset (Erkan, 2020).

However, fractional Brownian motion (fBm) has since been introduced by Mandelbrot and Ness (1968) which serves as an extension to its classical counterpart. In contrast, fBm is no longer guaranteed to be a martingale as the increments are not necessarily independent. Fractional Brownian motion can be defined in a number of ways, one of them is by the Mandelbrot-van Ness equation below (Mandelbrot and Ness, 1968):

$$B_t^H = \frac{1}{\Gamma(H + \frac{1}{2})} \left[\int_{-\infty}^{0} \left((t-s)^{H-\frac{1}{2}} - (-s)^{H-\frac{1}{2}} \right) dB_s + \int_0^t (t-s)^{H-\frac{1}{2}} dB_s \right] \quad (2.1)$$

where B_t is a classical Brownian motion and the Hurst parameter $H \in (0, 1)$ controls the long-range dependence of the process. We note that $H = \frac{1}{2}$ recovers a classical Brownian motion. Due to the covariance of this process, a lower or higher value for H would cause future increments to be negatively or positively correlated with past increments, respectively (Erkan, 2020).

To illustrate this, we simulate three realisations of a fractional Brownian motion with various Hurst parameters in the figures below. Each process is simulated for each t in $[0, 10]$ with 4096 equally spaced grid points.

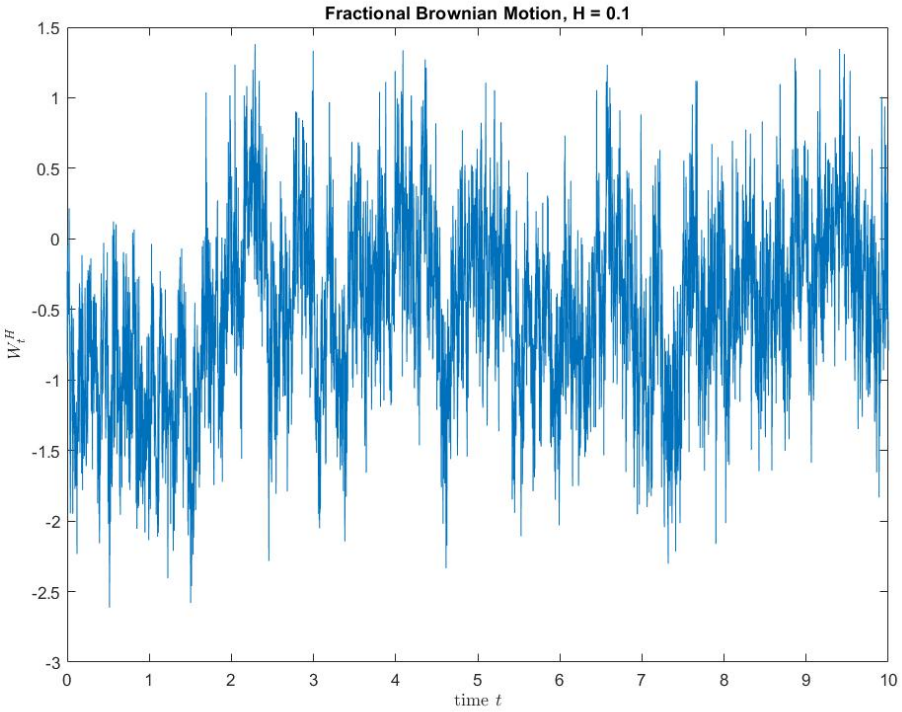

Fig. 2.1: Sample path of fractional Brownian motion with $H = 0.1$

As illustrated in Figure 2.1, the rough nature of fBm with $H = 0.1$ is indicated by the frequent erratic oscillations throughout time implying negative autocorrelations. Gatheral *et al.* (2018) argues that volatility in the market is not a long memory process but that volatility is rough, where the source of randomness follows a similar pattern to above.

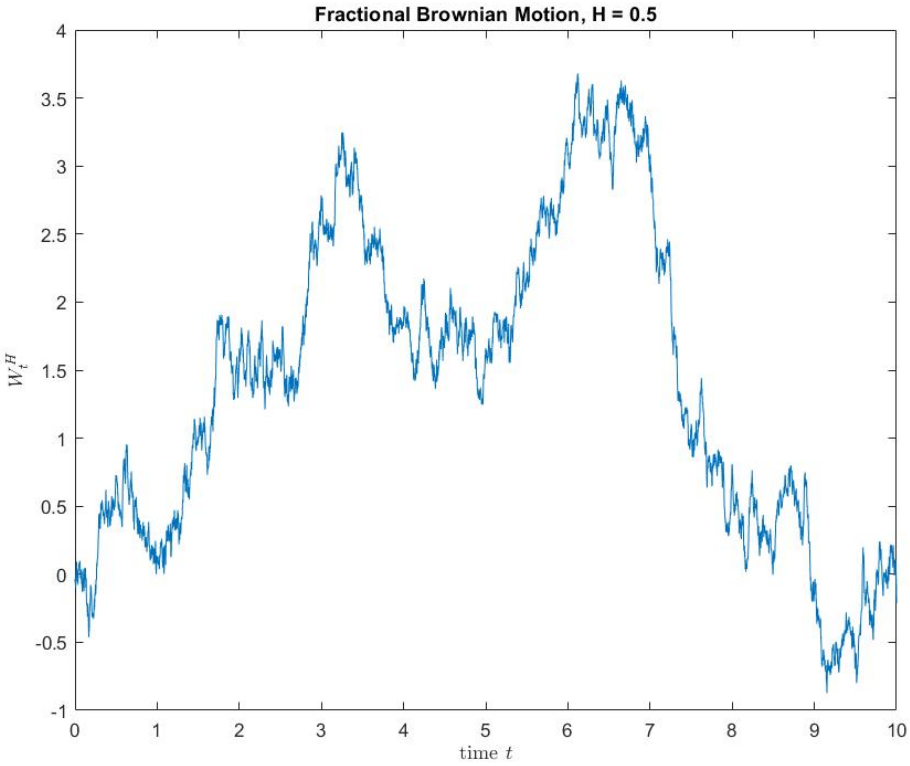

Fig. 2.2: Sample path of fractional Brownian motion with $H = 0.5$

In Figure 2.2 above, one can see that the sample path of a fBm with $H = 0.5$ is identical to a classical Brownian motion (Mandelbrot and Ness, 1968).

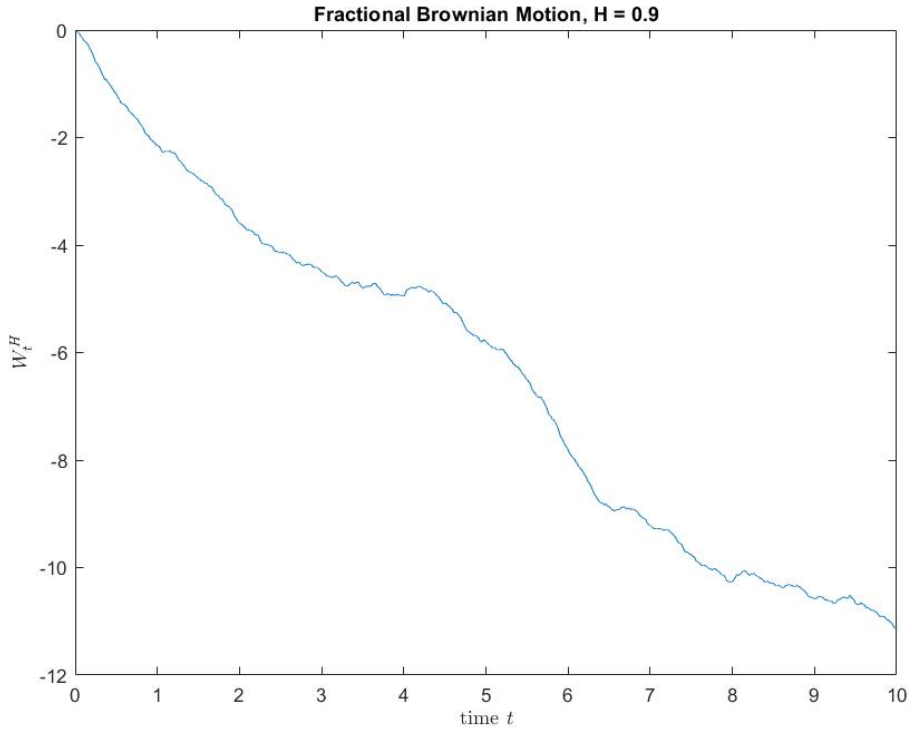

Fig. 2.3: Sample path of fractional Brownian motion with $H = 0.9$

In Figure 2.3 above, the sample path of a fBm with $H = 0.9$ is much smoother indicating positive autocorrelations. Thus, there is an inverse relationship between the Hurst parameter and the roughness of sample paths of a fBm. As the Hurst parameter is increased, the roughness of sample paths decreases. Fractional Brownian motion is therefore a process with long-range dependence i.e. future observations depend on past observations in a decaying manner (Erkan, 2020).

Comte and Renault (1998) pioneered the use of fractional stochastic volatility in their models. They included long-range dependence effects by selecting the Hurst parameter $H > \frac{1}{2}$, since it was then viewed that the volatility process had a long-term memory. However, due to recent research such as Gatheral *et al.* (2018) and Fukasawa (2017), the long-term memory fact can be disputed by analysing asymptotic behaviour.

2.2 Rough Volatility Models

"Rough" volatility uses the work of Mandelbrot and Ness (Mandelbrot and Ness, 1968) in defining a stochastic process with rough paths. There is much empirical

evidence in the literature to justify the consistency of rough volatility models with the dynamics of implied volatilities in the market.

Gatheral *et al.* (2018) justified rough volatility models as consistent with the market due to the contribution and proof provided by Fukasawa (2017). As pointed out by Jeng and Kilicman (2020), it is shown to resemble empirical volatility in the financial market better than classical Brownian motion.

Fukasawa (2017) showed how the slope of the implied volatility surface of SPX option prices at-the-money (ATM) obeys a power law relating to time-to-maturity. Denoting the implied volatility at time t with log-moneyness k and time to maturity τ as $\sigma_t(k,\tau)$, the power law equation is defined as:

$$\frac{\sigma_t\left(\sqrt{\tau}b,\tau\right)-\sigma_t\left(\sqrt{\tau}\beta,\tau\right)}{\sqrt{\tau}(b-\beta)}\sim A_t\tau^{H-0.5}\quad a.s. \tag{2.2}$$

as $\tau\to 0$ for $b\neq\beta$ and $t\geq 0$. $H\in(0,0.5)$ and A is a stochastic process.

The above power law behaviour is confirmed in the works of Alòs and Shiraya (2019) and Gatheral *et al.* (2018). In a more recent study, Fukasawa (2021) is adamant that volatility has to be rough. In particular, he proved that volatility must be rough where the time to maturity of options is very short and conversely, that non-rough volatility models are inconsistent to a power law of volatility skew (Fukasawa, 2021). This study is helpful as it ascertains the importance of using option price data with short maturity times for calibration to capture these market dynamics.

Empirical studies conducted by Livieri *et al.* (2018) revealed that estimates of the Hurst parameter values were lower for shorter maturities than longer maturities. This showed how implied volatility based approximations of the spot volatilities are modeled accurately by rough stochastic volatility when short maturities are considered. For instance, a value of $\hat{H}=0.06$ was obtained for a maturity of one day, whereas the actual simulated parameter value was $H=0.04$. In contrast, $\hat{H}=0.27$ was estimated for a twenty day maturity due to increasing bias. Livieri *et al.* (2018) attribute this bias to a smoothing effect as a result of the remaining time to maturity of the options i.e. the longer the time to maturity, the larger the smoothing effect.

Bayer *et al.* (2016) showed how actual SPX variance swap curves are consistent with model forecasts with examples that included the collapse of Lehman Brothers in 2008. The rough Bergomi model was applied, which fits more accurately than conventional Markovian stochastic models, with fewer parameters, defined as:

$$dX_t=-\frac{1}{2}V_t dt+\sqrt{V_t}dW_t,$$

$$d\xi_t^u = \xi_t^u \eta \sqrt{2\alpha + 1}(u - t)^\alpha dB_t,$$

where $\alpha = H - \frac{1}{2} \in (-\frac{1}{2}, 0)$ and $d\langle W, B\rangle_t = \rho dt$.

Experiments were performed to estimate parameters for the rough Bergomi model from market data for option prices and ATM implied volatilities. The forward variance curve and guessed parameter values of H, ρ and η were plugged into the rough Bergomi model. The results depicted a very good fit of the rough Bergomi model to the whole SPX volatility surface with guessed parameters of $H = 0.07$, $\eta = 1.9$ and $\rho = -0.9$ (Bayer *et al.*, 2016). This study is useful to determine the appropriate initial parameter values one must simulate when generating data from a rough volatility model.

Counter-opinions in the literature, provided by Cont and Das (2022) and Rogers (2019), refute the claim made by Gatheral *et al.* (2018) that 'volatility is rough' and stress the need for greater attention to exploration of the data. Their experiments indicate that one cannot assume that the roughness observed in realised volatility resembles similar behaviour in spot volatility. The fractional Ornstein-Uhlenbeck (OU) model is presented as a simpler and better alternative to a rough volatility model (Cont and Das, 2022), defined as:

$$\begin{aligned} dS_t &= \sigma_t S_t dB_t, \\ \sigma_t &= \sigma_0 e^{Y_t}, \\ dY_t &= -\gamma Y_t dt + \theta dB_t^H, \end{aligned}$$

where $\gamma = \theta = \sigma_0 = 1$, B is a Brownian motion and B^H a fractional Brownian motion with Hurst index $H \in (0, 1)$.

After computing realised volatility, comparisons were made of the estimated roughness index $\hat{H}_{L,K}$ (with $L = 300X300$, $K = 300$) for instantaneous and realised volatility. The results revealed that for smaller H (0.10), the instantaneous volatility is rougher (0.130) than realised volatility (0.190). However, when H increases (0.80), the realised volatility depicts significantly rougher behaviour (0.052) than instantaneous volatility (0.76). It was shown that the roughness index of realised volatility is consistently estimated to range between 0 to 0.3, regardless of the value of the Hurst exponent for instantaneous volatility for the price process. Cont and Das (2022) portrays that an inferior estimate for the Hurst estimate is found for realised volatility when there is smoother behaviour for the instantaneous volatility (corresponding to $H \geq 0.5$).

Rogers (2019) was particularly interested in analysing the large differences occurring in shorter time-scales and hence did further studies on high-frequency financial data based on seven days of WTI Crude Oil futures tick data in order to

explore this behaviour. The results showed that at shorter timescales, vast differences emerge with extreme fluctuations. Rogers (2019) suggests that this could be attributed to additive, independent and identically distributed (IID) noise.

Cont and Das (2022) and Rogers (2019) both conclude with a significant finding that a built-in estimation error, called microstructure noise exists in the data from which volatility is estimated that is attributable for the behaviour.

More recently, Mendes (2022) argues that volatility is rough in particular, when it is driven by fractional noise (fN) rather than fractional Brownian motion (fBm) and introduces a data reconstructed fractional volatility model to prove so. A comparison was made of a simulated path of 10000 steps of fBm at $H = 0.8$ with the one-step fractional noise ($B_H(t+1) - B_H(t)$). The results revealed that the apparent roughness of the fractional noise resembles fBm at $H = 0.1$.

Mendes (2022) makes a significant finding that the wrong Hurst index is obtained when using fBm as a basis for volatility and hence goes on to perfect this by integrating log-volatility and extracting the linear part to achieve a self-similar process. Consequently, long-range dependence and self-similarity are a characteristic of integrated log-volatility, not of volatility as such.

2.3 Pricing under the Rough Heston Model

A well-known model for encompassing rough stochastic volatility is the rough Heston model (El Euch *et al.*, 2019). Under the risk-neutral measure $\mathbb{Q}$, the one-dimensional asset price process dynamics are:

$$dS_t = S_t \sqrt{V_t} dW_t \tag{2.3}$$

where we assume that the risk-free rate $r = 0$ for simplicity.

The dynamics of the stochastic volatility are:

$$V_t = V_0 + \frac{1}{\Gamma(\alpha)} \int_0^t (t-s)^{\alpha-1} \lambda(\theta - V_s) ds + \frac{1}{\Gamma(\alpha)} \int_0^t (t-s)^{\alpha-1} \lambda \nu \sqrt{V_s} dB_s \tag{2.4}$$

where the parameters λ, θ, V_0 and ν are positive and W and B are two Brownian motions with correlation ρ. $\Gamma(.)$ is the Gamma function. The smoothness of volatility sample paths is determined by the parameter $\alpha \in (\frac{1}{2}, 1)$ and has the relation $\alpha = H + \frac{1}{2}$ where H is the Hurst parameter. Thus there are a total of six model parameters that could be calibrated.

One must first obtain option prices or implied volatilities (which could be simulated from the aforementioned model) to be used in the calibration process to

calibrate the rough Heston model to market data and thus estimate the Hurst parameter (El Euch *et al.*, 2019). By utilizing the characteristic function of the model, a clear and robust approach can be obtained for pricing options.

2.3.1 Characteristic Function

El Euch and Rosenbaum (2019) use a convergence-type proof of Hawkes processes to derive the characteristic function of the rough Heston model.

It follows a similar form to that of the classical Heston model, particularly:

$$\phi_{X_t}(a) = \mathbb{E}[e^{iaX_t}] = \exp\left[g_1(a,t) + V_0 \cdot g_2(a,t)\right]$$

where $X_t = \log(S_t/S_0)$ and

$$g_1(a,t) = \theta\lambda \int_0^t h(a,s)ds$$
$$g_2(a,t) = I^{1-\alpha}h(a,t)$$

and h is the solution of the fractional Riccati equation:

$$D^\alpha h = \frac{1}{2}(-a^2 - ia) + \lambda(ia\rho\nu - 1)h(a,s) + \frac{(\lambda\nu)^2}{2}h^2(a,s)$$
$$I^{1-\alpha}h(a,0) = 0$$

where D^α and I^α are operators denoting the Riemann-Liouville fractional derivative and integral of order α respectively.

The solution to the above fractional Riccati equation can be achieved by using various numerical techniques since the equation for $\alpha < 1$ is no longer explicit.

2.3.2 The Fractional Adams Method

Initially introduced by Diethelm *et al.* (2004), the well-known fractional Adams method is a common numerical method for solving fractional differential equations.

It is used in various literature such as El Euch and Rosenbaum (2019), Erkan (2020) and Jeng and Kilicman (2020) in order to solve for the specified fractional Riccati equation and is presented below:

Firstly, we need to find the solution of the fractional Riccati equation:

$$D^\alpha h(a,x) = F(a, h(a,x))$$
$$I^{1-\alpha}h(a,0) = 0$$
$$\text{where} \quad F(a,x) = \frac{1}{2}(-a^2 - ia) + \lambda(ia\rho\nu - 1)x + \frac{(\lambda\nu)^2}{2}x^2.$$

The characteristic function infers the Volterra equation (El Euch and Rosenbaum, 2019):

$$h(a,t) = \frac{1}{\Gamma(\alpha)} \int_0^t (t-s)^{\alpha-1} F(a, h(a,s)) ds.$$

To approximate this, we construct a regular time grid, $(t_k)_{k\in\mathbb{N}}$, where $t_k = k\Delta$, and $f(a,t) := F(a, h(a,t))$.

Therefore:

$$h(a, t_{k+1}) \approx \frac{1}{\Gamma(\alpha)} \int_0^{t_{k+1}} (t_{k+1} - s)^{\alpha-1} \hat{f}(a,s) ds$$

with

$$\hat{f}(a,t) = \frac{t_{j+1} - t}{t_{j+1} - t_j} f(a, t_j) + \frac{t - t_j}{t_{j+1} - t_j} f(a, t_{j+1}), \quad t \in [t_j, t_{j+1}), \quad 0 \le j \le k.$$

Then, for fractional integrals, we apply the trapezoidal rule and quadrature techniques to arrive at the approximation:

$$\hat{h}(a, t_{k+1}) = \frac{1}{\Gamma(\alpha)} \int_0^{t_{k+1}} (t_{k+1} - s)^{\alpha-1} \left[\frac{s_{j+1} - s}{s_{j+1} - s_j} \hat{f}(a, s_j) + \frac{s - s_j}{s_{j+1} - s_j} \hat{f}(a, s_{j+1}) \right] ds$$

where

$$s_k = k\Delta, \quad s \in [s_j, s_{j+1}).$$

A solution is then given by (El Euch and Rosenbaum, 2019):

$$\hat{h}(a, t_{k+1}) = \sum_{0 \le j \le k} a_{j,k+1} F(a, \hat{h}(a, t_j)) + a_{k+1,k+1} F(a, \hat{h}(a, t_{k+1})),$$

where

$$a_{j,k+1} = \frac{\Delta^\alpha}{\Gamma(\alpha+2)} ((k-j+2)^{\alpha+1} + (k-j)^{\alpha+1} - 2(k-j+1)^{\alpha+1}), \quad 1 \le j \le k,$$

$$a_{0,k+1} = \frac{\Delta^\alpha}{\Gamma(\alpha+2)} (k^{\alpha+1} - (k-\alpha)(k+1)^\alpha),$$

$$a_{k+1,k+1} = \frac{\Delta^\alpha}{\Gamma(\alpha+2)}.$$

However, this scheme is implicit so for the final explicit scheme, we use a predictor-corrector approach as follows:

The predictor is computed as:

$$h^P(a, t_{k+1}) = \sum_{0 \le j \le k} b_{j,k+1} F(a, \hat{h}(a, t_j)),$$

with

$$b_{j,k+1} = \frac{\Delta^\alpha}{\Gamma(\alpha+1)}((k-j+1)^\alpha - (k-j)^\alpha), \quad 0 \leq j \leq k.$$

Finally, the scheme results in an explicit approximation (El Euch and Rosenbaum, 2019):

$$\hat{h}(a, t_{k+1}) = \sum_{0 \leq j \leq k} a_{j,k+1} F(a, \hat{h}(a, t_j)) + a_{k+1,k+1} F(a, h^P(a, t_{k+1})).$$

It should be noted that the fractional Adams method is fundamentally not the most efficient method for approximating the characteristic function of the rough Heston model. To compensate therefore, the multi-point Padé approximation method recently introduced in Jeng and Kilicman (2020) improves upon the high computational cost of the Adams method and finds the solution with only a $O(1)$ order of complexity, but this will not be explored further in this dissertation.

2.4 Calibration Methods

Calibration is a technique for estimating model parameters using observable market information (Liu *et al.*, 2019). A variety of different calibration methods have been explored in the literature to estimate model parameters from market data.

In an experiment conducted by Horvath *et al.* (2019), calibration of the rough Bergomi model with historical data by using Neural Network approximation was compared to that of the Brute Force Monte Carlo calibration method. A comparison was made of surface relative errors of the Neural Network approximator with the Monte Carlo method across all training data (34000 random parameter combinations) in the rough Bergomi model.

The results showed that the difference between the rough Bergomi model's Neural Network approximation and the Brute Force Monte Carlo approach was usually less than 0.2 percent and conclude that the Neural Network approximation of the rough Bergomi model provides a better fit than the Brute Force Monte Carlo method. Horvath *et al.* (2019) conclude that this could be attributable to the exact gradients in the Neural Network, whereas approximate gradients are used in Brute Force calibration.

Some researchers have proposed alternatives to improve the speed and accuracy of calibration. In another comparative study of the rough Bergomi model with the Brute Force calibration method, Zeron and Ruiz (2020) use Chebyshev Tensors to overcome the computational bottleneck found with calibration of the rough Bergomi model. The results were 40000 times more efficient than if calibrated via Brute Force method using the pricing function.

Previously, Malliavin calculus techniques were applied by Alòs and Shiraya (2019) that incorporated the case of a Hurst parameter $H < \frac{1}{2}$ in order to derive the short-time behaviour of ATM implied volatility skews resembling those displayed in the financial market. They observed a log-linear relationship between the ATM implied volatility and volatility swaps with the Hurst parameter thus implying a simple linear regression calibration approach. The results depicted that most of the Hurst parameters are estimated accurately, especially when very short times to maturity are considered (Alòs and Shiraya, 2019).

In another calibration experiment conducted by El Euch *et al.* (2019), a Brute Force minimization procedure as applied within the rough Heston model was used to compare the SPX volatility surface results for two dates, namely, 14 August 2013 and 19 May 2017 and to show that the rough Heston model fits the market well. A Hurst parameter of $H = 0.0474$ was obtained confirming that the model fits the market well and corresponds to very rough volatility.

The application of Machine Learning, in particular, to solve optimization problems in the calibration procedure, is rapidly expanding (Hernandez, 2016), (Horvath *et al.*, 2019), (Liu *et al.*, 2019). In particular, through the use of Artificial Neural Networks (ANNs) the optimal parameter values for the rough Heston model can be determined (Erkan, 2020).

Finally, Liu *et al.* (2019) and Erkan (2020) discuss several advantages of utilizing Neural Networks as an efficient and accurate framework for calibrating financial models. Calibration is fast due to the use of global optimization techniques, such as differential evolution, where all the market samples can be computed at the same time and optimal values determined within a second. In addition, Neural Networks have proven to be extremely robust, computationally cheap and accurate (Liu *et al.*, 2019). However, cognisance should be taken of the limitations and shortcomings as reported in the literature (Horvath *et al.*, 2019). Therefore, a prudent approach would be to weigh up the benefits against the limitations and explore mitigation measures to overcome these limitations before implementing Neural Networks as a calibration method.

Chapter 3

Methodology

The following three calibration methods are used to map the (non-linear) relationship between implied volatility and model parameter values for the rough Heston model, namely a Brute Force minimization procedure, a Neural Network calibration and Linear Regression.

For each calibration procedure, the Hurst parameter H is the main parameter of interest to be calibrated to a volatility surface. Therefore, we have selected these methods from literature based on this objective. However, the remaining parameters within the rough Heston model specification are still able to be calibrated in both the Brute Force and Neural Network methods. The option-related parameters will remain fixed throughout, such as the times to maturity T, strike prices K and asset spot price S_0. A volatility surface can either be simulated from the model or sourced from market data for calibration.

To simulate volatility surface data from the rough Heston model, sets of model parameters were randomly generated to compute European option prices with fixed option-related parameters. The (Black-Scholes) implied volatility was then computed numerically from these prices using the `blsimpv` function in MATLAB.

The parameters in the rough Heston model which are simulated consist of:

$$\Theta = (V_0, \lambda, \theta, \nu, \rho, H)$$

3.1 Brute Force Minimization

A Brute Force minimization procedure is proposed as in El Euch *et al.* (2019). The implied volatility for a specific strike and maturity is denoted as $\sigma_i := \sigma_i(K_i, T_i)$ where K_i is the strike price and T_i the maturity time (in years). Given a set of model parameters Θ, implied volatility values can be generated from the model $\sigma_i^M(\Theta)$. These model parameters are thus calibrated by minimizing the distance between

the model implied volatility $\sigma_i^M(\Theta)$ and the market data implied volatility σ_i:

$$\sum_i \left(\sigma_i^M(\Theta) - \sigma_i\right)^2 \tag{3.1}$$

subject to the constraints:

$$V_0 > 0, \quad \lambda > 0, \quad \theta > 0, \quad \nu > 0, \quad \rho \in [-1, 1], \quad H \in (0, 0.5].$$

In general, optimization requires a choice of the objective function. In this case, a simple sum of squared differences is chosen as the objective function to be minimized. In addition, lower and upper bounds for each parameter and a predefined tolerance level must be set to determine when the algorithm should terminate. Therefore one may expect this method to be more time-consuming to run than the other methods proposed, depending on the choice of hyperparameters. However, it is easier to understand and simple to implement in practice. This would allow a practitioner to obtain a rough estimate of the Hurst parameter from market data with a relatively permissible implementation.

3.2 Neural Networks

Neural Networks have recently been used for the calibration problem (Liu *et al.*, 2019), specifically for rough volatility models as in Erkan (2020) and Horvath *et al.* (2019).

The Neural Network structure for the rough Heston model calibration comprises of six neurons in the output layer which correspond to the model parameters. The implied volatility surface is used as data for the input layer, where the number of neurons is equal to the number of implied volatilities on the surface provided. The choice of hidden layers and number of neurons in each hidden layer are hyperparameters that should be tuned while training the network.

We calibrate using Neural Networks with the following steps (Erkan, 2020):

1. Simulate a dataset for the financial model's input parameters, such as $\Theta = (V_0, \lambda, \theta, \nu, \rho, H)$ for the rough Heston model,

2. By applying the schema described in Section 2.3, we price the corresponding European options and compute implied volatilities from the model,

3. Split the data into subsets for purposes of training, cross-validation and testing. The Neural Network is then trained using a training algorithm, such as backpropagation on the training subset,

4. Using the remaining test data, the performance of the Neural Network is evaluated by relevant metrics such as the root mean square error (RMSE) and parameter relative error,

5. Finally, predict the model parameters with the trained Neural Network by using available market data as the input.

The parameters which are updated when training the Neural Network are the weights and biases of neurons in each subsequent layer, denoted as (Erkan, 2020):

$$\Theta' = (W_1, B_1, W_2, B_2, ..., W_L, B_L)$$

where B_i represents the bias vector and W_i the weight matrix of the n^{th} layer and L is the number of layers.

An activation function is used for the neuron behaviour in each hidden layer and output layer. The output signal of the i^{th} neuron in the n^{th} layer is:

$$z_i^n = \varphi^n(\sum_j w_{ji}^n z_j^{n-1} + b_i^n)$$

where b_i^n and w_{ji}^n represent the corresponding weight and bias in the bias vector and weight matrix, and $\varphi(.)$ may be a different activation function for each layer n for $1 \leq n \leq L$. The Neural Network is able to learn more complex solutions by using the activation function which enables the input data to be non-linearly transformed (Erkan, 2020).

3.3 Linear Regression

Recently, Alòs and Shiraya (2019) have used Malliavin calculus to show that there is a linear relationship between the at-the-money implied volatility skew and the logarithm of the maturity time for fractional stochastic volatility models where the skew has order $H - \frac{1}{2}$. This implies that the slope of a Linear Regression conducted on these two variables would be $H - \frac{1}{2}$. This would yield to be an efficient way to find an estimate for the Hurst parameter.

Firstly, we define the log-moneyness as:

$$k := \ln\left(\frac{S_0}{K}\right)$$

where S_0 is the spot asset price and K is the strike price.

If we denote the implied volatility with moneyness k and time to maturity T as $\sigma := \sigma(k, T)$, then we can define the ATM implied volatility skew as:

$$\left|\frac{\partial \sigma}{\partial k}\right|_{k=0}.$$

This is computed by a numerical approximation using finite differences. In our implementation, we make use of the central difference approximation:

$$\frac{\partial \sigma}{\partial k} = \frac{\sigma(k+h,T) - \sigma(k-h,T)}{2h}$$

where h is a fixed non-negative value.

This approximation is good when h is small i.e. the forward and backward moneyness values are close together. The larger this difference, the worse the approximation to the derivative becomes. In practice, implied volatility data from traded options with moneyness that is nearly at-the-money would be more suitable for this method.

Linear Regression provides us with a well-known way to estimate the slope from a linear relationship between two variables. A simple Linear Regression would have a response variable Y and one explanatory variable X. An intercept is taken into account to improve the model fit. In this case, we would have $Y = \left|\frac{\partial \sigma}{\partial k}\right|_{k=0}$ as the response variable and $X = \ln T$ as the explanatory variable. Thus, the model is described as:

$$Y = XB$$

where $B = (\beta_0, \beta_1)$ is the parameter vector consisting of parameters for the intercept and slope respectively.

Estimates for the parameter vector are obtained by ordinary least squares. Finally, an estimate for the Hurst parameter can then be obtained by adding $\frac{1}{2}$ to the slope i.e. $\hat{H} = \hat{\beta}_1 + \frac{1}{2}$.

Chapter 4

Numerical Experiments

An analysis was conducted to determine the behaviour and consistency of market prices with the rough Heston model for rough volatility in the options market. Firstly, preliminary experiments were performed to compare the behaviour of the rough Heston model with the classical Heston model. Thereafter, simulated data from the rough Heston model was generated and parameters calibrated using each of the three methods described in Chapter 3. Lastly, calibration experiments were conducted on market data from traded options and the fitted models were compared across the three calibration methods used.

All implementation was run in MATLAB R2019a on a Windows PC with an Intel 2.9 GHz CPU and 16GB RAM.

4.1 Preliminary Experiments

Classical Heston model option prices were obtained by implementing the Little Trap characteristic function pricing approach in MATLAB. European call option prices were obtained from the rough Heston model by implementing the characteristic function pricing approach and the fractional Adams numerical method in MATLAB as described in Section 2.3. Black-Scholes implied volatilities were then obtained from these model call option prices using the `blsimpv` function in MATLAB.

Rough and classical Heston volatility surfaces were generated from the schema in Section 2.3 and are shown in Figure 4.1 below.

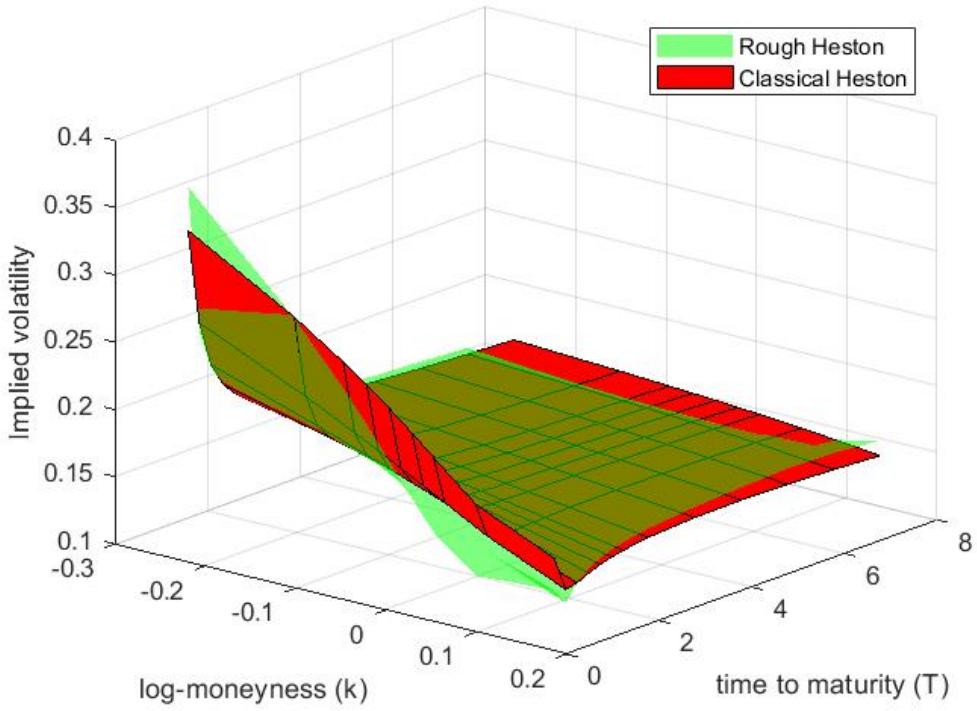

Fig. 4.1: Rough vs. Classical Heston model volatility surface.

At first glance, it seems that one cannot distinguish whether classical or rough Heston would fit market data better based on the shape of the implied volatility surface alone. However, once the term structure of ATM implied volatility skews are investigated one can see the major difference in how the rough Heston model captures the market behaviour more accurately as reported in the literature by Fukasawa (2017). As shown in Figure 4.2, the ATM skew in the rough Heston model explodes at short maturity times i.e. as T tends to zero.

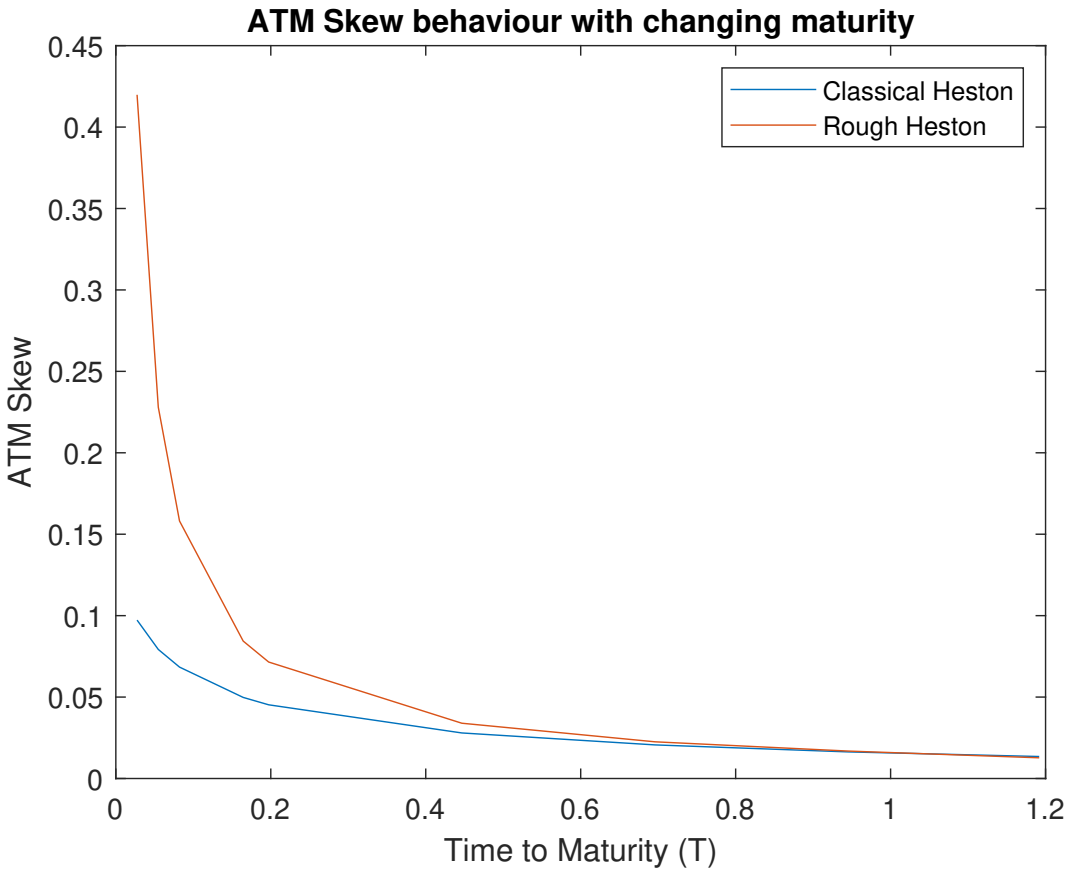

Fig. 4.2: Rough vs. Classical Heston model ATM skews.

The roughness of volatility provided by calibrating the Hurst parameter thus allows for more accurate pricing than other fractional stochastic volatility (FSV) models (Gatheral *et al.*, 2018).

Going further, based on the effects of Fukasawa (2017), we plot the ATM implied volatility skews of four different sources of market data (European options written on the FTSE/JSE Africa Top 40 (ALSI) Index in 2017, 2018, 2019 and 2020 respectively). Since the market for options written on ALSI futures is typically illiquid, a model for the construction of a volatility surface is calibrated to actual trade data on any given day. This constructed volatility surface is used as the official volatility surface for valuation and risk management and is sourced from Bloomberg. However, the volatility surface is a true reflection of the actual market and therefore serves as the best estimate (Kotzé and Joseph, 2009).

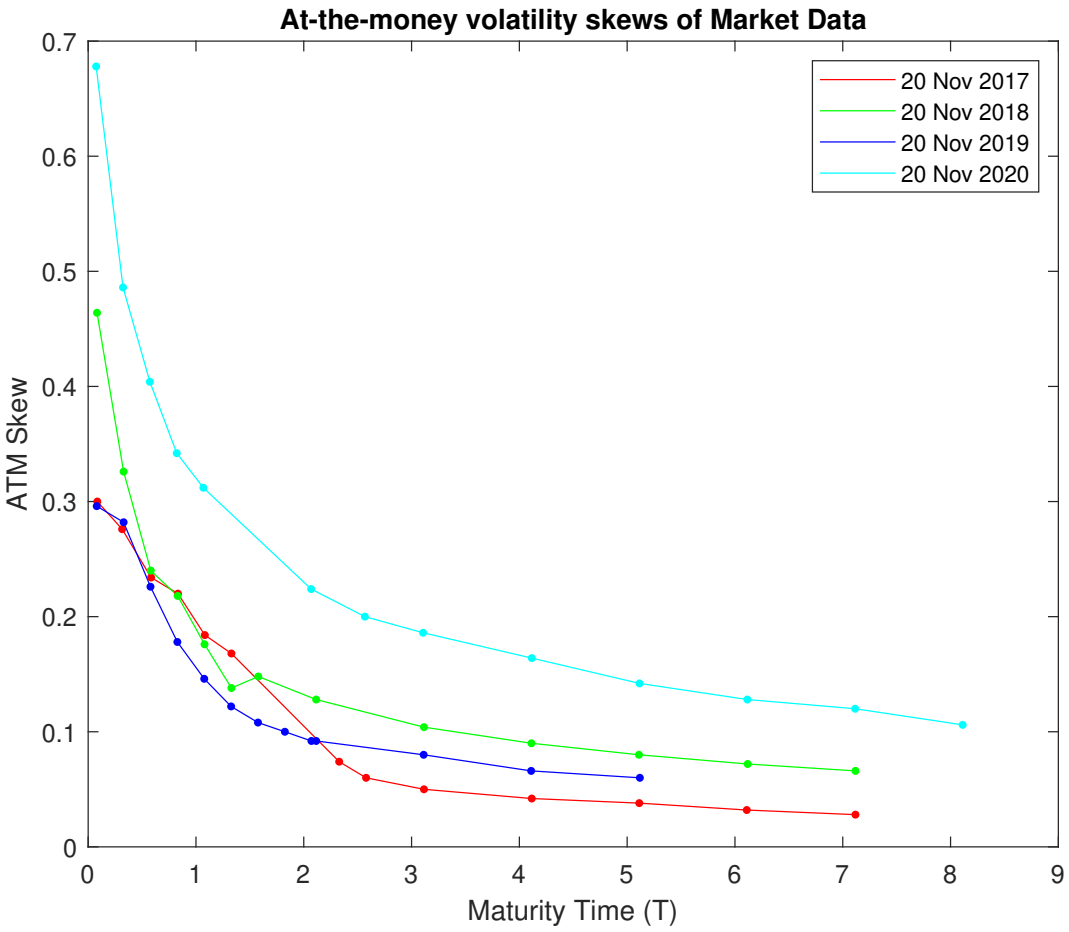

Fig. 4.3: ATM skews of market data.

The market behaviour in Figure 4.3 shows that as the ATM skews explode, the maturity of the option becomes shorter which is consistent with the literature (Alòs and Shiraya, 2019). This provides us with a basis for our rough volatility models as they reflect this same explosion at the short-time asymptotics of the option as shown in Figure 4.2.

The market data ATM skew term structure obeys the power law fit as described by Fukasawa (2017), a characteristic of the rough Heston model. This is true especially for short maturity times where the option is near expiry, as there is an explosion of the ATM skew value.

In order to infer that there exists a linear relationship between the ATM skew and the logarithm of time to maturity as proven in Alòs and Shiraya (2019), a Linear Regression was performed. The slope obtained is directly related to a Hurst parameter estimate.

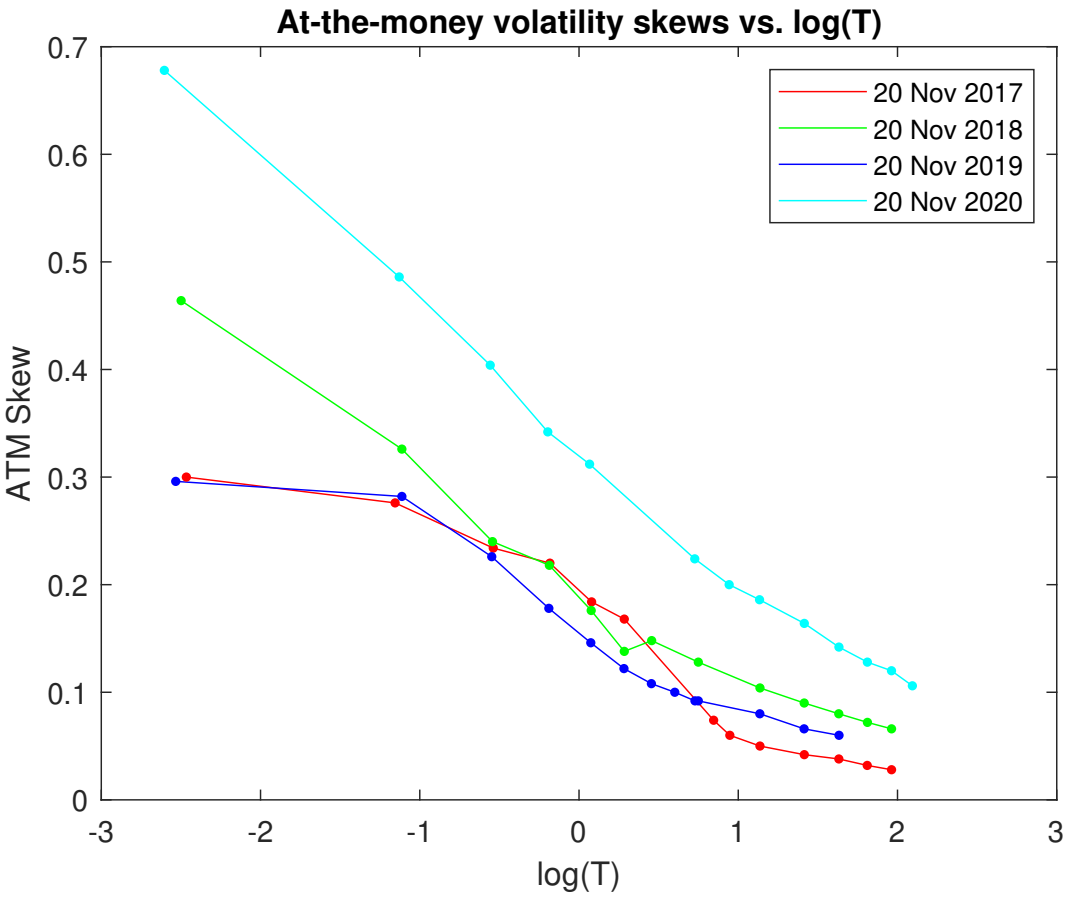

Fig. 4.4: Market data - ATM skews vs. log(T).

Figure 4.4 illustrates that an approximate linear relationship between the market data ATM skew and $\log T$ exists for each market volatility surface provided. There is no linear relationship for the classical Heston model but there is somewhat a linear relationship for rough Heston, especially at the shorter maturity times. This provides the basis for the Linear Regression calibration method as implemented in this dissertation.

4.2 Calibration on Simulated Data

4.2.1 Data Description and Pre-Processing

In this section, we fix a set of option-related parameters and simulate n sets of rough Heston model parameters Θ. These parameter values are used to generate n volatility surfaces to be used as input data for each calibration method, which will output estimates of these model parameters. Relevant accuracy metrics can therefore be used to compare the performance of each method, such as parameter

relative error, which is a good measure of the quality, accuracy and performance of the calibration process. The parameter relative error is defined as:

$$Error(\hat{\theta}) = \frac{|\hat{\theta} - \theta|}{|\theta|}$$

where $\hat{\theta}$ is the parameter estimate and θ is the actual (simulated) parameter value.

With the Neural Network method in mind, $n = 1000$ volatility surfaces with ten maturity times and three strike prices (or moneyness levels) were generated from the rough Heston model. These surfaces were simulated based on 1000 different (random) model parameter Θ values. 850 of these were for training the Neural Network and 150 were used for testing across all three methods.

The data was generated with the following bounds for the model-related parameters:

$$\begin{aligned}
V_0 &\in [0.01, 0.02] \\
\lambda &\in [1, 2] \\
\theta &\in [0.01, 0.02] \\
\nu &\in [0.01, 0.02] \\
\rho &\in [-1, -0.95] \\
H &\in [0.1, 0.5]
\end{aligned}$$

and fixed option-related parameters:

$$S_0 = 100$$

$$r = 0$$

$$T = 0.0137, 0.0274, 0.0548, 0.0822, 0.1644, 0.2466, 0.5, 1, 1.5, 2$$

i.e.

$$T = 5\text{D}, 10\text{D}, 20\text{D}, 1\text{M}, 2\text{M}, 3\text{M}, 6\text{M}, 1\text{Y}, 1.5\text{Y}, 2\text{Y}$$

and

$$K = 99, 100, 101.$$

It is noted that a limitation of the numerical approximation (fractional Adams method) to price options from the rough Heston model was found where negative prices were computed (for certain parameters). In these cases, implied volatilities could not be obtained. Additionally, a limitation for the numerical methods used (Newton's method) to compute implied volatilities from prices is also noted. For certain (positive) prices that were far out-the-money or in-the-money, the numerical method was not able to converge to compute the Black-Scholes implied volatility.

Due to these limitations, the moneyness and expiry times had to be restricted so that volatility surfaces could be generated from the rough Heston model smoothly. Although only few moneyness levels are considered, the aim is to compare the different calibration methods on the same test data. The robustness of each calibration method is therefore excluded from testing. Further research could be done on alternative numerical approximations to the rough Heston model and implied volatility which increase stability, robustness and would allow for broader analysis.

Thereafter, the total dataset was randomly split with 70% for training data and 15% for cross validation (for training the Neural Network and tuning the parameters of the Brute Force method) and the remaining 15% to be used as test data. Therefore, the same volatility surfaces and parameters in the test dataset would be used to compare the accuracy of each method.

4.2.2 Results

Each calibration method described in Chapter 3 was implemented to estimate rough Heston model parameters. The below results present the performance of each method on the same test data, which consisted of 150 implied volatility surfaces with corresponding sets of model parameters.

Brute Force Minimization

The Brute Force minimization scheme took a long time to run as each volatility surface had to be optimized using the `fmincon` function in MATLAB, resulting in parameter estimates for each volatility surface.

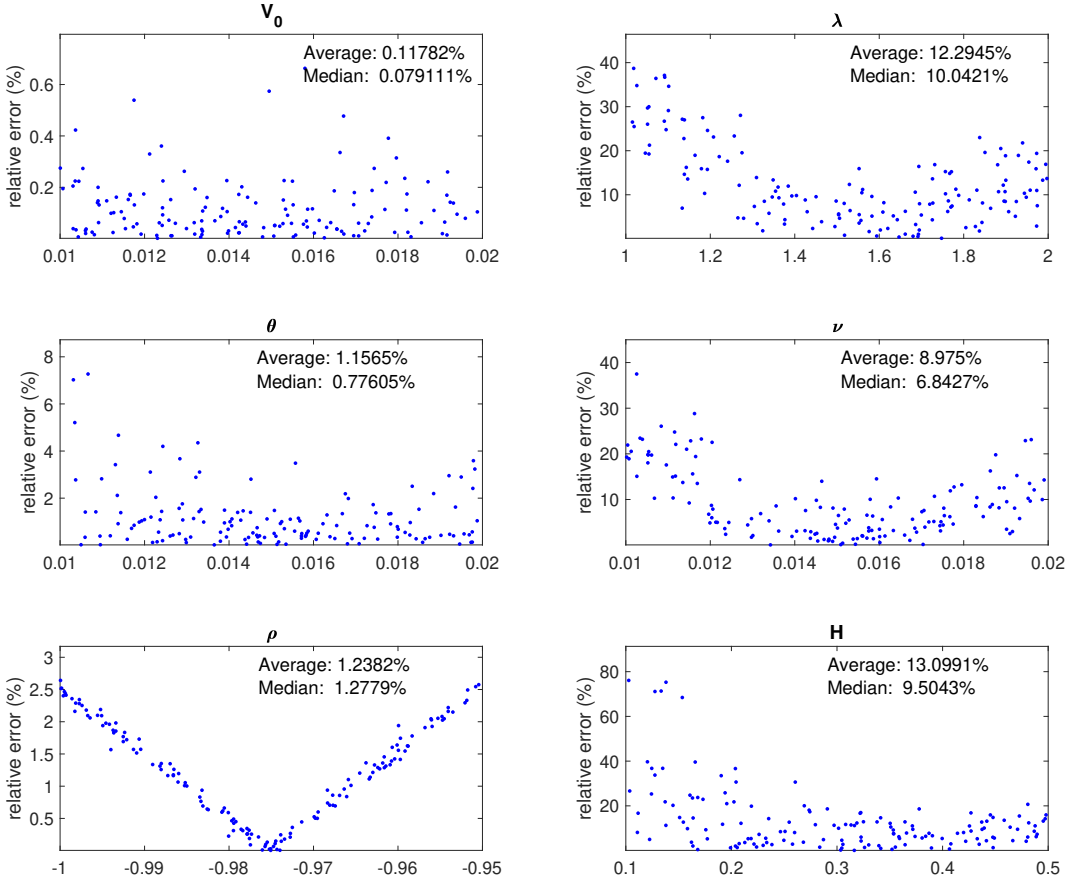

Fig. 4.5: Calibration relative error per parameter in the test set in the rough Heston model - Brute Force Method.

Figure 4.5 shows relative errors per parameter after calibration of the rough Heston model for the Brute Force method. We observe that the largest errors are concentrated for small H values with an average relative error of 13.0991% followed by λ with an average relative error of 12.2945%. This implies a lower performance of the calibration process using these parameters. On the other hand, the parameter V_0 proved to be the most accurate with an average relative error of 0.11782%.

Although this method may have a very long run time and is model dependent, it does return a good initial guess for the model parameters and can be used on any type of model for asset prices. The computation time may be impractical, however, the different hyperparameters can be tuned so that the difference between the model data and market data can be accepted to a certain degree of tolerance to reduce the computation time. In order to investigate this, an additional experiment was conducted to first tune the hyperparameters on training data before estimating model parameters on the test data.

The training data generated for the neural network (850 volatility surfaces)

would be used to tune the Brute Force hyperparameters and initial parameters which will be used when estimating model parameters on the 150 test surfaces. The options within `fmincon` were altered for different sets of the training data, and the average time per run and average accuracy (parameter relative error) was computed. The options (hyperparameters) altered were the *MaxFunctionEvaluations*, *MaxIterations*, *ConstraintTolerance*, *OptimalityTolerance* and *StepTolerance*.

After the average time and accuracy per case in the training dataset were recorded, the optimal hyperparameters were chosen and the calibration was re-run on the 150 test datapoints. Additionally, the initial parameters were set to be the average of the estimated parameters within the training dataset. However, these results were not significantly different from before i.e. no significant reduction in computational time and/or increase in accuracy and therefore the results in Figure 4.5 remain.

Neural Network

The neural network was constructed using the `nnstart` function in MATLAB. It is noted that a very simple neural network configuration was applied in this way.

For the configuration of the network architecture a general rule of thumb was used to prevent over-fitting: Only one hidden layer was used to avoid complexity and the number of neurons in the hidden layer was set to be no more than:

$$N_h = \frac{N_s}{\alpha \cdot (N_i + N_o)}$$

where:

- N_h = recommended number of neurons in the hidden layer
- N_i = number of neurons in the input layer
- N_o = number of neurons in the output layer
- N_s = number of samples in the training data
- α = an arbitrary scaling factor, chosen to be 2 in this case.

Therefore, we have:

$$N_h = \frac{700}{2 \cdot (30 + 6)} = 9.72.$$

The number of neurons in the hidden layer was therefore set to be $5 \leq 9.72 = N_h$.

The Neural Network was trained using the Bayesian regularization training function, and the mean squared error as the objective function. After the network

was trained, predictions were made for model parameters based on 150 implied volatility surfaces.

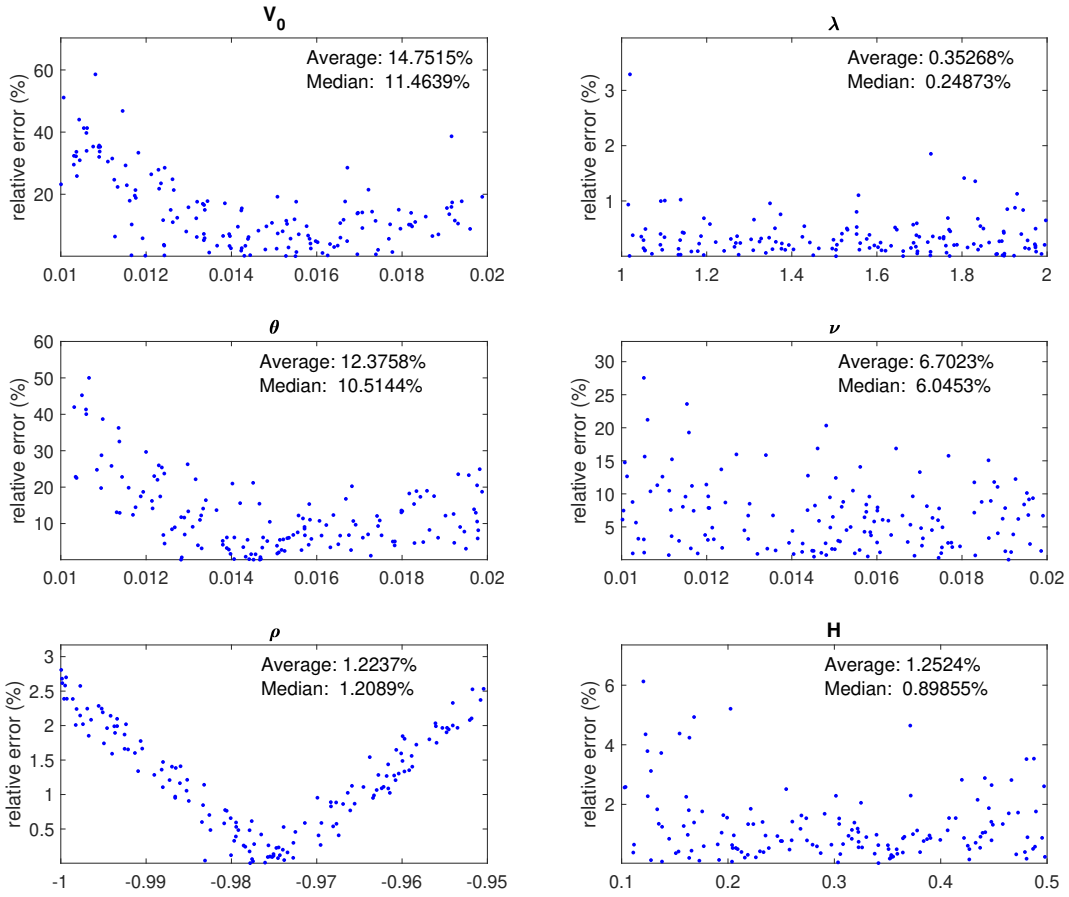

Fig. 4.6: Calibration relative error per parameter in the test set in the rough Heston model - Neural Network Method.

As shown in Figure 4.6, the largest relative errors are concentrated for the V_0 parameter with an average relative error of 14.7515% implying a lower performance of the calibration process in the Neural Network method for this parameter. On the other hand, estimates for λ proved to be the most accurate with an average relative error of 0.35268%, followed by ρ with an average relative error of 1.2237% and thereafter H with an average relative error of 1.2523%.

An advantage of this method is that training can be done offline when markets are closed in order to save time and thereafter parameters can be calibrated online when markets are open again. However, the disadvantage is that data has to firstly be trained which consists of an additional step. Furthermore, Hernandez (2016) recommends that retraining should take place every two to three months to improve the performance and pick up on new market trends.

Linear Regression

The Linear Regression method was then performed on the simulated test data to estimate the Hurst parameter H. As this method is model-free, it provides a quick way to judge if the volatility in a given market is rough, using just implied volatility and time to maturity data.

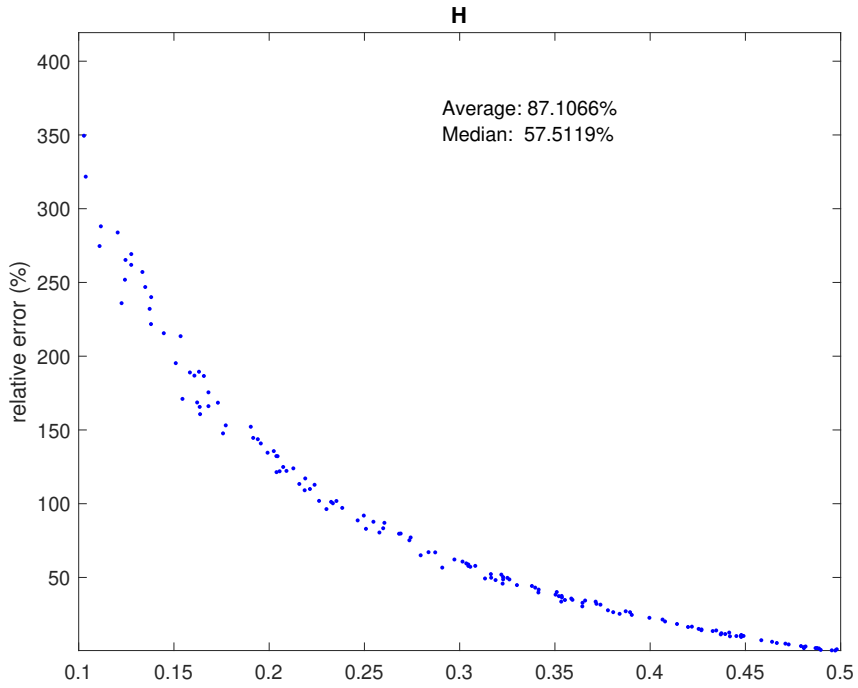

Fig. 4.7: Calibration relative error for Hurst parameter in the test set in the rough Heston model - Linear Regression Method.

We observe in Figure 4.7 that the average relative error for H is 87.1066% implying a lower performance of the calibration process in the Linear Regression method. This may be due to the limited data set which only makes use of the ATM values of the volatility surface, unlike the other methods which utilize the full data. Furthermore, the rough Heston model may not be appropriate to the theorems in Alòs and Shiraya (2019) as certain assumptions need to be met. In this paper, they make use of a model similar to the fractional SABR model which is not necessarily a model for rough volatility. Also, the Linear regression method is limited in that it only estimates the Hurst parameter H and thus the other model parameters would need to be calibrated in some other manner.

Although it ran fast, this method did not perform very well in terms of accuracy. In addition, the linear relationship was shown to diverge for longer-dated maturities and seemed to perform better on shorter maturities (less than one year).

However, it can be improved by expanding the maturity set.

Comparative Results

Overall, the comparative performance of the three calibration methods for the simulated test experiments are shown below:

Tab. 4.1: Comparison of Calibration Methods: Average Relative Error (%) per parameter.

	Method		
Parameter	*Brute Force*	*Neural Network*	*Linear Regression*
V_0	0.11782	14.7515	
λ	12.2945	0.35268	
θ	1.1565	12.3758	
ν	8.975	6.7023	
ρ	1.2382	1.2237	
H	13.0991	1.2523	87.1066

As depicted in Table 4.1 above, estimates for the Hurst parameter H proved to be the most accurate in the Neural Network method with an average relative error of 1.2523%, followed by Brute Force minimization with an average relative error of 13.0991%. Linear Regression proved to be the least accurate with an average relative error of 87.1066% as it only makes use of the ATM skew of the data so less data is used here.

In comparison with other parameters, the Neural Network seemed to perform most effectively for λ with an average relative error of 0.35268%, followed by ρ with an average relative error of 1.2237% and thereafter H with an average relative error of 1.2523%. On the whole, based on the combined average relative error across all parameters, the Neural Network performed similarly (36.66%) to the Brute Force minimization method (36.88%). This is consistent with the results of the literature experiments conducted by Horvath *et al.* (2019) who concluded that the Neural Network method may provide a better fit than other methods possibly attributable to the use of exact gradients compared to using approximate gradients.

Furthermore, the computational time was measured across the three methods. The Neural Network method took 46.31 minutes in total to generate the 1000 volatility surfaces for training i.e. 2.78 seconds on average to generate each surface. It took 21.86 seconds to train the Neural Network on the 850 surfaces and 0.0463 seconds on average to estimate the model parameters on one test surface. We note that the Neural Network has to train on simulated data similar to market data which takes

time. The time taken depends on the amount of data generated and is a choice to improve accuracy. However, once the neural network is trained on this data, it can estimate parameters quickly on (unseen) test or market data.

On the other hand, Brute Force took 19.18 minutes on average to estimate parameters per test surface. For 150 surfaces, the total time was 47 hours. Even though hyperparameters were tuned and initial parameters were selected as the average of estimates obtained on the training data, there were no significant improvements in results in terms of accuracy and speed.

Linear Regression is the fastest method with an average speed of 0.0000752 seconds to estimate H from a term structure of ATM volatility skew. It is virtually instant but suffers from inaccuracy. However, it is a model-free method and can be used to quickly gauge an estimate of the Hurst parameter in the market.

4.3 Calibration on Market Data

4.3.1 Data Description and Pre-Processing

The market data was sourced from the JSE Equity Derivatives Market (SAFEX) and consisted of European options written on the underlying Futures contract of the FTSE/JSE Top40 Index (ALSI) traded throughout the month of October 2009. For each trade, data included the option type, strike price, time to maturity, option price and traded volatility. The traded option data on the ALSI would be a good proxy to represent South African market data as it represents the performance of the largest 40 companies ranked by market capitalization included in the FTSE/JSE All Share Index.

In South Africa, the market for options on ALSI futures is typically illiquid compared to options on the more liquid S&P500 Index futures. Additionally, volatility jumps may occur for trades with short expiry dates and there is sparse data for longer expiry dates. Furthermore, there are rarely any at-the-money trades. Therefore, the illiquid quotes from this market cannot directly determine a volatility surface and rather a methodology for constructing this volatility surface must be applied.

There are a variety of methodologies used to construct a volatility surface from market data. Homescu (2011) provides a collection of these methodologies which have been used in different markets in the past. Additionally, Kotzé and Joseph (2009) and West (2005) both postulate models which are calibrated to South African market data.

In particular, Kotzé and Joseph (2009) provide a methodology which is applied specifically to options on the ALSI futures market. They postulate two independent

models which are combined to form the volatility surface. Additionally, they show that the calibration of these models with certain constraints avoids arbitrage and the constructed volatility surface is thus a true reflection of the market (Kotzé and Joseph, 2009).

The first model portrays the volatility skew given an expiry time with a quadratic deterministic function:

$$\sigma_{model}(\beta_0, \beta_1, \beta_2) = \beta_0 + \beta_1 K + \beta_2 K^2,$$

where $K = \frac{\text{Strike}}{\text{Spot}}$ is the moneyness, $\beta_0 > 0$, $-1 < \beta_1 < 0$ and $\beta_2 > 0$. The β parameters control the shift, slope and convexity of volatility respectively.

The second model depicts the ATM volatility term structure:

$$\sigma_{atm}(\tau) = \frac{\theta}{\tau^{\lambda}},$$

where τ is the months to expiry, λ controls the slope of the term structure and θ controls the curvature.

The above models were calibrated to actual trade data on 6 October 2009 using the methodology in Kotzé and Joseph (2009) and resulted in the following estimated parameters:

Tab. 4.2: Calibrated Parameters for Market Volatility Skew.

Time to Maturity (months)	β_0	β_1	β_2
2.3671	0.2483	-0.7869	0.7746
5.3589	0.2034	-0.6860	0.7178
8.3507	0.1825	-0.6367	0.6888
11.3425	0.1694	-0.6048	0.6694

Tab. 4.3: Calibrated Parameters for Market ATM Volatility Term Structure.

θ	λ
0.2515	0.012

Thereafter, three moneyness levels and four expiry times were selected and volatilities were computed from the calibrated models. Therefore, the final constructed market volatility surface is shown in the table below:

Tab. 4.4: Market Volatility Surface.

	Strike (K)		
Time to Maturity (years)	97.5	100	102.5
0.1972	0.217489	0.248825	0.255609
0.4466	0.216996	0.246364	0.254482
0.6960	0.216512	0.245038	0.253554
0.9452	0.216070	0.244127	0.252770

The above construction in Table 4.4 was then imported into MATLAB for the implementation of the calibration methods on the market volatility surface as at 6 October 2009.

The particular option-related parameters when computing model option prices throughout this section were fixed to be $S_0 = 100$ and $r = 0$.

4.3.2 Results

Brute Force Minimization

For the Brute Force method, the parameter bounds were chosen to be:

$$\begin{aligned} V_0 &\in [0.01, 0.1] \\ \lambda &\in [0.01, 2] \\ \theta &\in [0.01, 0.1] \\ \nu &\in [0.01, 0.02] \\ \rho &\in [-0.99, 0.99] \\ H &\in [0.01, 0.49]. \end{aligned}$$

The Brute Force model parameter estimates on the data are:

$$\hat{V_0} = 0.0470, \quad \hat{\lambda} = 1.9972, \quad \hat{\theta} = 0.0698, \quad \hat{\nu} = 0.0200, \quad \hat{\rho} = 0.9894, \quad \hat{H} = 0.0110.$$

Neural Network

With the estimates of model parameters from the Brute Force method obtained, the Neural Network method was then performed. New training data (model implied volatilities) had to be simulated to match the dimensions of the market data (four maturity times and three moneyness levels). Parameters were simulated with bounds that were centered around the Brute Force parameter estimates which had

run beforehand. This was done in an effort to train the Neural Network on data that is similar to what the market data resembles. These bounds are as follows:

$$
\begin{aligned}
V_0 &\in [0.04, 0.05] \\
\lambda &\in [1.8, 2] \\
\theta &\in [0.06, 0.07] \\
\nu &\in [0.015, 0.025] \\
\rho &\in [0.9, 1] \\
H &\in [0.01, 0.499].
\end{aligned}
$$

In practice, the Neural Network would have to be trained on a wide range of data, with implied volatility surfaces of different shapes and levels, so that the network can determine which parameter sets are most appropriate for many different market surfaces. Hernandez (2016) stresses the importance of including an adequate set of training data which needs to cover a wide range of parameter values and determine the Neural Network training bounds.

Thereafter, the simulated training data was cleaned to remove any errors which resulted from the limitations of the numerical methods implemented to obtain option prices from the rough Heston model and implied volatilities.

The neural network was built with two hidden layers containing five neurons each and the number of epochs was set to 10000. The remaining hyperparameters were the same as in Section 4.2. After the training stage, the trained network was used to predict parameter estimates given the market implied volatility surface.

The following rough Heston model parameter estimates were thus obtained:

$$\hat{V}_0 = 0.0470, \quad \hat{\lambda} = 1.9950, \quad \hat{\theta} = 0.0698, \quad \hat{\nu} = 0.0200, \quad \hat{\rho} = 0.9981, \quad \hat{H} = 0.0110.$$

Linear Regression

We then performed the Linear Regression method to find an estimate for the Hurst parameter H in this market:

$$\hat{H} = 0.4821.$$

From the above, the volatility surfaces from the calibrated models of the Brute Force and Neural Network methods were plotted and compared with the market data.

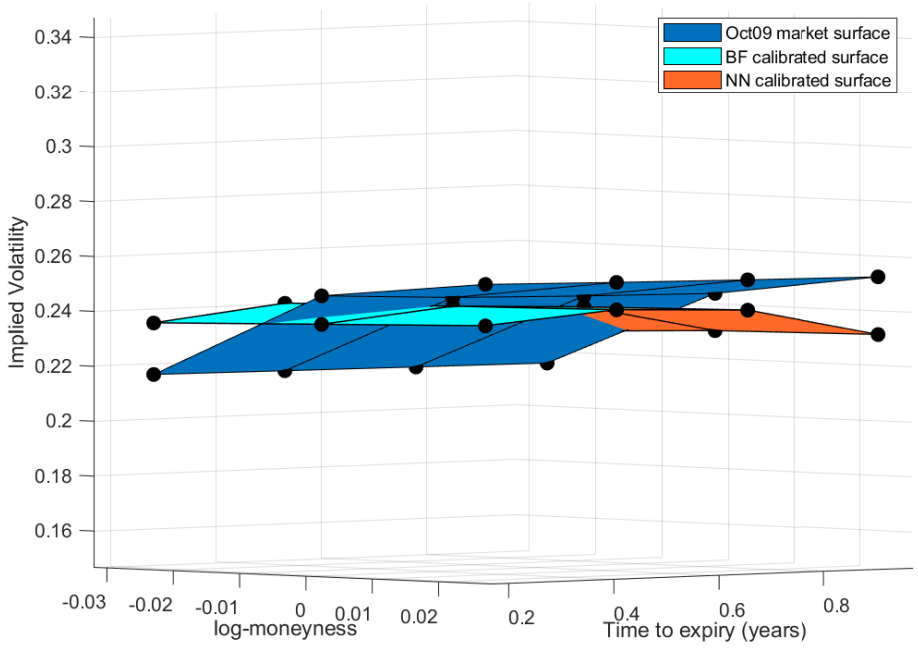

Fig. 4.8: Comparison of market and calibrated volatility surfaces.

As depicted in Figure 4.8, the calibrated volatility surfaces using both the Brute Force method and Neural Network are very similar. They are shown to slightly resemble the market volatility surface. However, the calibrated rough Heston volatilities permit higher errors when the market volatilities have higher or lower degrees of moneyness.

Possible reasons for the divergent behaviour could be attributed to the limited training data simulated for training the network. As pointed out by Hernandez (2016), the Neural Network should be trained on a myriad of volatility surfaces of different shapes and sizes, to ensure that supervised learning of the model parameters occurs best where there is a match to these volatility surfaces. Additionally, an optimal number and range of hyperparameters should be chosen to improve training of the neural network, which was not investigated in this dissertation.

Additionally, only twelve data points were used in the volatility construction and calibration. If a denser market volatility surface was considered, this could result in an improved fit using the calibration methods. However, due to the limitations of the numerical approximation used to compute rough Heston prices (fractional Adams), and the limitations of the numerical methods used to compute the Black-Scholes implied volatilities from these model prices, the construction and calibration was restricted to a 4x3 volatility surface.

Depending on the estimate of H obtained in this market, one can then decide on an appropriate model to use for the degree of roughness in a fractional stochastic volatility (FSV) model. All three of the calibration methods have estimated different Hurst parameters from the market. However, they all are similar in suggesting that volatility is rough in this market as $\hat{H} < 0.5$.

Chapter 5

Conclusion

The choice of an accurate, fast and robust calibration method to be able to model option prices from market data/implied volatilities is important for accurate pricing, hedging and managing risk in financial markets. The Hurst parameter H is regarded as an invaluable tool to ensure volatility is rough i.e. $H < 0.5$.

In this dissertation we explored three different calibration methods, namely, a Brute Force minimization procedure, Neural Networks and Linear Regression in order to determine and evaluate their performance as measured by accuracy (average relative error) and computational speed as applied to the rough Heston model.

It is evident from the calibration experiments on simulated data, that Neural Networks performed similarly to Brute Force in terms of accuracy. In terms of speed, Neural Networks may be slower overall than Brute Force depending on how much training data is generated. On any given day, the Brute Force will take a few minutes to estimate model parameters from a volatility surface. On the other hand, if the Neural Network has been pre-trained already, this is advantageous as it can estimate model parameters within a few seconds. It has a number of advantages such as offline training and all market samples can be computed simultaneously (Liu *et al.*, 2019). More time should be given to training which focuses on the ability to deal with the large number of price ranges and parameter values, as confirmed in the literature (Hernandez, 2016). As a possible solution, Hernandez (2016) proposes that neural networks should be retrained every 2-3 months. Horvath *et al.* (2019) propose including an intermediate step of learning involving pricing functions of volatility models before calibration rather than being trained directly on data in order to improve performance. Additionally, Zeron and Ruiz (2020) propose the use of Chebyshev tensors in order to improve the speed.

The Brute Force minimisation method, proved to be reasonably accurate in accordance with the literature El Euch *et al.* (2019). However, it is slower in terms of speed, possibly due to the `fmincon` constrained minimization function used in

MATLAB. As a possible solution, the different hyperparameters should be tuned so that the difference between the model data and market data can be accepted to a certain degree of tolerance to reduce the computation time.

The Linear Regression method was shown to be the least accurate as measured by average relative error, in contrast to the literature in Alòs and Shiraya (2019). This could possibly be due to the limited data set. However, computational speed was fast and it is model-independent and hence, could possibly be recommended as a viable calibration method for estimating a Hurst parameter from the market without building a model.

In sum, we find that the performance of the Neural Network method is not much better than the Brute Force minimisation method for simulated calibration, whereas the Linear Regression method, although it was the fastest, was the least accurate. Based on the results obtained, one can conclude that Brute Force minimization and Neural Networks and to a lesser extent the Linear Regression method, can be relied upon to determine the consistency of market prices/implied volatility with rough volatility models. Furthermore, the results are consistent with the rough volatility behaviour described in Gatheral *et al.* (2018) implying that the behaviour of the rough Heston model is also consistent with real-life market data.

It is evident from the calibration experiments on market data, that the implied volatility surfaces obtained from the Brute Force method as well as the Neural Networks resemble the market implied volatility surfaces.

Finally, it should be noted that this dissertation focused on an overall comparison of three different calibration methods for option pricing models to obtain estimates for the Hurst parameter H. However, there is a need for in-depth analysis of the behaviour of these methods, in particular, Neural Networks and their behaviour when calibrating market data, for future research. In addition, further research can be performed on more appropriate numerical methods, such as Pade's Multipoint approximation method described in Jeng and Kilicman (2020) in order to address the limitations of the fractional Adams method.

In conclusion, overall the calibration methods described in this paper can be considered to be suitable to the South African market in order to obtain estimates of the Hurst parameter for rough volatility models. The final results for the Hurst parameter estimates on market data were $\hat{H} = 0.0110$ for the Brute Force minimization method, $\hat{H} = 0.0110$ for the Neural Network method and $\hat{H} = 0.4821$ for the Linear Regression method which suggests that volatility is rough for this South African market as $\hat{H} < 0.5$. However, it should be noted that the calibration of rough volatility within the South African market is still relatively new with sparsity of research available and it remains a challenge to obtain appropriate South

African market data for these purposes. Hence, there is a need for further research to be conducted in this field.

Bibliography

Alòs, E. and Shiraya, K. (2019). Estimating the Hurst Parameter from Short Term Volatility Swaps: a Malliavin Calculus Approach, *Finance and Stochastics* **23**(2): 423–447.

Bayer, C., Friz, P. and Gatheral, J. (2016). Pricing Under Rough Volatility, *Quantitative Finance* **16**(6): 887–904.

Clegg, R. G. (2006). A Practical Guide to Measuring the Hurst Parameter, *arXiv preprint math/0610756* .

Comte, F. and Renault, E. (1998). Long Memory in Continuous-time Stochastic Volatility Models, *Mathematical finance* **8**(4): 291–323.

Cont, R. and Das, P. (2022). Rough Volatility: Fact or Artefact?, *arXiv preprint arXiv:2203.13820* .

Diethelm, K., Ford, N. J. and Freed, A. D. (2004). Detailed Error Analysis for a Fractional Adams Method, *Numerical algorithms* **36**(1): 31–52.

El Euch, O., Gatheral, J. and Rosenbaum, M. (2019). Roughening Heston, *Risk* pp. 84–89.

El Euch, O. and Rosenbaum, M. (2019). The Characteristic Function of Rough Heston models, *Mathematical Finance* **29**(1): 3–38.

Erkan, K. (2020). European Option Pricing Under the Rough Heston Model using the COS Method.

Fang, F. and Oosterlee, C. W. (2009). A Novel Pricing Method for European Options Based on Fourier-cosine Series Expansions, *SIAM Journal on Scientific Computing* **31**(2): 826–848.

Fukasawa, M. (2017). Short-time At-the-money Skew and Rough Fractional Volatility, *Quantitative Finance* **17**(2): 189–198.

Fukasawa, M. (2021). Volatility Has to be Rough, *Quantitative Finance* **21**(1): 1–8.

Funahashi, H. and Kijima, M. (2017). Does the Hurst Index Matter for Option Prices Under Fractional Volatility?, *Annals of Finance* **13**(1): 55–74.

Gatheral, J., Jaisson, T. and Rosenbaum, M. (2018). Volatility is Rough, *Quantitative Finance* **18**(6): 933–949.

Hernandez, A. (2016). Model Calibration with Neural Networks, *Available at SSRN 2812140* .

Homescu, C. (2011). Implied Volatility Surface: Construction Methodologies and Characteristics, *arXiv preprint arXiv:1107.1834* .

Horvath, B., Muguruza, A. and Tomas, M. (2019). Deep Learning Volatility A Deep Neural Network Perspective on Pricing and Calibration in (Rough) Volatility Models, *arXiv preprint arXiv:1901.09647* .

Hurst, H. E. (1951). Long-term Storage Capacity of Reservoirs, *Trans. Amer. Soc. Civil Eng.* **116**: 770–799.

Jeng, S. W. and Kilicman, A. (2020). Fractional Riccati Equation and Its Applications to Rough Heston Model Using Numerical Methods, *Symmetry* **12**(6): 959.

Kilianová, S. and Letko, B. (2018). An Empirical Study on using Hurst Exponent Estimation Methods for Pricing Call Options by Fractional Black–Scholes Model, *Risk and Decision Analysis* **7**(1-2): 51–62.

Kirichenko, L., Radivilova, T. and Deineko, Z. (2011). Comparative Analysis for Estimating of the Hurst Exponent for Stationary and Nonstationary Time Series, *Information Technologies & Knowledge* **5**(1): 371–388.

Kotzé, A. and Joseph, A. (2009). Constructing a South African Index Volatility Surface from Exchange Traded Data, *Available at SSRN 2198357* .

Liu, S., Borovykh, A., Grzelak, L. and Oosterlee, C. (2019). A Neural Network-based Framework for Financial Model Calibration, *Mathematics in Industry* **9**(1): 1–28.

Livieri, G., Mouti, S., Pallavicini, A. and Rosenbaum, M. (2018). Rough Volatility: Evidence From Option Prices, *IISE transactions* **50**(9): 767–776.

Mandelbrot and Ness, V. (1968). Fractional Brownian Motion, Fractional Noises and Applications, *SAIM Review* **10**(4): 422–437.

Mendes, R. V. (2022). The Fractional volatility Model and Rough Volatility, *arXiv preprint arXiv:2206.02205* .

Mitra, S. (2012). Is Hurst Exponent Value Useful in Forecasting Financial Time Series, *Asian Social Science* **8**(8): 111.

Rogers, L. (2019). Things We Think We Know, *Preprint, available at https://www. skokholm. co. uk/lcgr/downloadable-papers* .

West, G. (2005). Calibration of the Sabr Model in Illiquid Markets, *Applied Mathematical Finance* **12**(4): 371–385.

Zeng, W. S. and Tang, S. Z. (2011). Bias Correction in Logarithmic Regression and Comparison with Weighted Regression for Nonlinear Models, *Nature Precedings* pp. 1–1.

Zeron, M. and Ruiz, I. (2020). Tensoring Volatility Calibration, *arXiv preprint arXiv:2012.07440* .