Article

SWIFT Calibration of the Heston Model

Eudald Romo [1] and Luis Ortiz-Gracia [2],*

[1] Trading Department, Xanadu Trading Limited, 08011 Barcelona, Spain; eudald.romo@xanadutrading.com
[2] Department of Econometrics, Statistics and Applied Economics, University of Barcelona, 08034 Barcelona, Spain
* Correspondence: luis.ortiz-gracia@ub.edu

Abstract: In the present work, the SWIFT method for pricing European options is extended to Heston model calibration. The computation of the option price gradient is simplified thanks to the knowledge of the characteristic function in closed form. The proposed calibration machinery appears to be extremely fast, in particular for a single expiry and multiple strikes, outperforming the state-of-the-art method we compare it with. Further, the a priori knowledge of SWIFT parameters makes a reliable and practical implementation of the presented calibration method possible. A wide range of stress, speed and convergence numerical experiments is carried out, with deep in-the-money, at-the-money and deep out-of-the-money options for very short and very long maturities.

Keywords: Heston model; calibration; European options; Shannon wavelets

1. Introduction

The Heston model is a well-known stochastic volatility (SV) model for driving the dynamics of the assets. In order to use the Heston model, we need to calibrate its five parameters to real-market data. The goal of calibrating a model using market data is to estimate the model parameters in such a way that, when it is used for option valuation with an appropriate option valuation method, it yields prices similar to the real market ones. Calibration is an important task that requires efficient numerical methods. It encompasses machinery for option pricing as well as an optimization procedure, aiming at minimizing the differences between the market option prices and the prices given by the valuation method. Regarding the pricing, we use the SWIFT method presented in [1] for European options. This belongs to the class of Fourier inversion methods and has already been used for pricing Bermudan, barrier and Asian options (see [2,3]). The power of the SWIFT method partly relies on the knowledge of the characteristic function (ChF) in analytical form. Since the ChF associated with the Heston model is known in closed form, we can tackle the optimization problem by means of the gradient-based Levenberg–Marquardt (LM) algorithm [4]. For sake of comparison, we consider the state-of-the-art calibration method of [5], which is based on the Fourier inversion pricing method of [6], and it also uses the LM optimization algorithm. The main contributions of this work are the following:

- We extend the SWIFT method to the calibration problem by deriving the option price gradient;
- We implement and test the speeding-up techniques mentioned in [1] based on multiple strike valuation;
- We propose a novel method for calibrating the Heston model with a set of options with certain fixed strikes that can be later used for arbitrary strikes by interpolation;
- We develop and implement speeding up techniques for the option price gradient.

We carry out a wide variety of stress, speed and convergence tests with at-the-money (ATM), deep in-the-money (ITM) and deep out-the-money (OTM) options, ranging from very short to very long maturities. The results show that SWIFT is extremely fast for sets of options with a single expiry and different strikes, being about ten times faster than

the calibration method of [5]. For options with a fixed number of maturities and a fixed number of strikes per maturity, both methods perform similarly in terms of speed, with the SWIFT method being more robust, thanks to the possibility of selecting the parameters of the pricing machinery a priori. This last feature makes the SWIFT method a reliable and practical methodology for the real-time updating of Heston model calibration.

The paper is organized as follows. We define the calibration and the valuation problem in Section 1. The Heston model, the valuation method of [5] and the calibration challenges are presented in Section 2. Section 3 is devoted to the SWIFT method and its speeding up features. In Section 4, we put forward the calibraton problem with all the mathematical details, and we test our proposed method through the numerical experiments of Section 5. Section 6 concludes and gives some pointers for future research.

1.1. Option Valuation

The option valuation will be tackled under the framework of the expected discounted payoff pricing formula that we recall next. Consider a European option contract with strike price K, expiring at time T, with $\tau := T - t$ the time to maturity, and S_t the price of its underlying asset at time t. Then, if one considers state variables x and y, which fully describe S_t and the random variable S_T, respectively, the general pricing formula becomes

$$v(x,t) = e^{-r\tau}\mathbb{E}^{\mathbb{Q}}(v(y,T)|x) = e^{-r\tau}\int_{\mathbb{R}} v(y,T)f(y|x)dy, \tag{1}$$

where $v(x,t)$ denotes the option value at time t, $v(y,T)$ is the payoff, r the risk-neutral interest rate, $\mathbb{E}^{\mathbb{Q}}$ the expectation under the risk-neutral measure, and $f(y|x)$ is the probability density of y given x. A common choice for the state variables x and y is to define

$$y = \ln\left(\frac{S_T}{K}\right), \tag{2}$$

$$x = \ln\left(\frac{S_t}{K}\right). \tag{3}$$

More precisely, given the states variables' choice, we can express the payoff of a European option in log-asset price as

$$v(y,T) = \left[\alpha \cdot K(e^y - 1)\right]^+ := K \cdot g(y), \quad \text{with,} \quad \alpha = \begin{cases} 1, & \text{for a call,} \\ -1, & \text{for a put,} \end{cases} \tag{4}$$

where $g(y) := \max(\alpha(e^y - 1), 0)$ denotes the strike-free payoff.

2. Heston Dynamics and Calibration Issues

The widely used Black and Scholes (BS) model fails to capture essential well-known properties of the real-world market dynamics of the underlying return distributions, such as its high kurtosis, its negative skew, the correlation between the underlying price and its volatility, the risk premium investors give deep (ITM) or (OTM) options, etc. All these properties result in the well-known BS implied volatility surface. SV models treat both the underlying price and its volatility as (potentially correlated) stochastic processes, which helps to better capture some of these properties. One of the first and most well-known SV models is the Heston model [6], defined by the following system of stochastic differential equations.

Definition 1. *The Heston model price–volatility dynamics are defined by*

$$dS_t = \mu S_t dt + \sqrt{v_t}S_t dW_t^{(1)},$$
$$dv_t = \kappa(\bar{v} - v_t)dt + \sigma\sqrt{v_t}dW_t^{(2)},$$

where $W_t^{(1)}$ and $W_t^{(2)}$ are two correlated Wiener processes $dW_t^{(1)} dW_t^{(2)} = \rho dt$ and v_t is the variance of the underlying asset price at time t. Then, if one specifies the initial value of the variance v_0, the model is properly defined. From now on, $\boldsymbol{\theta} := (v_0, \bar{v}, \rho, \kappa, \sigma)^T$ will refer to the vector of model parameters.

Several works have shown the relationship between the Heston model parameters and the shape of the implied volatility surface [7–10] necessary to obtain the same prices with a BS model. It can be summarized as follows:

- v_0 controls the position of the volatility surface;
- ρ controls its skewness;
- κ and σ control the convexity of the surface;
- $\kappa(v_0 - \bar{v})$ controls the term structure of implied volatility.

A method for calibrating the Heston model is presented in [5]. This method starts from the expression of the price of a European call option presented in the original work by Heston [6], and it is adapted here in terms of the state variables x and y defined in Section 1.1

$$v(x, \tau) = Ke^x P_1(\boldsymbol{\theta}; x, \tau) - Ke^{-r\tau} P_2(\boldsymbol{\theta}; x, \tau), \tag{5}$$

where P_1 and P_2 are defined as

$$P_1(\boldsymbol{\theta}; x, \tau) = \frac{1}{2} + \frac{1}{\pi} \int_0^\infty \mathrm{Re}\left(\frac{e^{iux}}{iu} \hat{f}(-u+i)\right) du, \tag{6}$$

$$P_2(\boldsymbol{\theta}; x, \tau) = \frac{1}{2} + \frac{1}{\pi} \int_0^\infty \mathrm{Re}\left(\frac{e^{iux}}{iu} \hat{f}(-u)\right) du, \tag{7}$$

and $\hat{f}(u)$ is the initial state independent ChF of the process. It is worth remarking that, as we will see in Section 3, the SWIFT method will benefit from the fact that the ChF $\hat{f}(u)$ does not depend on the initial state variable.

Remark 1. *The dependence of the ChF on time and the model parameters is omitted for readability.*

Remark 2. *The expression (5) omits the dividend yield term q, which appears in [5] but is assumed to be 0 here for readability. The results presented here are valid for any constant value q.*

Remark 3. *Typically, the ChF of a random variable with density function f is defined as $\tilde{f}(u) = \int_{\mathbb{R}} f(x)e^{iux} dx$. However, to be consistent with [1], it is defined in this work as $\hat{f}(u) = \int_{\mathbb{R}} f(x)e^{-iux} dx$. We can see that there is a sign difference in all the u-dependent equations and expressions in [5].*

We write expression (5) in a more compact form in the following lemma

Lemma 1 (Heston's pricing method). *Let $V(\theta; x, \tau)$ be the price of a European call option with strike K and Heston dynamics, given by expression (5). If we define $\hat{f}(u; x) = e^{-iux} \hat{f}(u)$, then*

$$V(\theta; x, \tau) = K \left[\frac{1}{2}(e^x - e^{-r\tau}) + \frac{e^{-r\tau}}{\pi} \int_0^\infty \mathrm{Re}\left(\frac{\hat{f}(-u+i; x) - \hat{f}(u; x)}{iu}\right) du \right]. \tag{8}$$

Proof. Having into account that $S_t/K = e^{\log(S_t/K)} = e^x$ then, from expression (5)–(7) we have

$$V(\theta; x, \tau) = K\frac{1}{2}(e^x - e^{-r\tau}) + K\frac{e^{-r\tau}}{\pi} \left[\int_0^\infty \mathrm{Re}\left(\frac{e^{i(u-i)x}}{iu} \hat{f}(-u+i)\right) du - \int_0^\infty \mathrm{Re}\left(\frac{e^{iux}}{iu} \hat{f}(u)\right) du \right].$$

Finally, since $\hat{f}(u; x) = e^{-iux} \hat{f}(u)$, then the result follows. $\quad\square$

2.1. Calibration Challenges

As opposed to simpler one-dimensional models, Heston model calibration is a multidimensional optimization problem with five degrees of freedom given by $\boldsymbol{\theta} := (v_0, \overline{v}, \rho, \kappa, \sigma)^T$. Furthermore, the structure of this optimization problem is not known. According to [5], no consensus exists among researchers regarding whether the objective function of this optimization problem is convex or not. Some results point to a non-convex function, such as the calibration methods proposed in [11,12] (which yielded different results for different initial points) and one must use long- or short-term approximations and rules to provide a convenient initial guess. Recent research [13] claims to provide methods that reach a unique solution independently of the initial point which, according to that study, indicates some structure that, even if not necessarily convex, tends to lead an initial guess to a stationary result. There is also no consensus on whether the problem always has a single optimum. In particular, it is known that there exist dependencies between the parameters that yield similar results. For example, $\lim_{t \to \infty} \text{Var}(v_t) = \dfrac{\sigma^2 \overline{v}}{\kappa}$ (where $\text{Var}(\cdot)$ refers to the variance operator), so large values of κ and σ can provide a model that prices options similarly to one with proportionally smaller values of these two parameters. The work by [5] claims that this results in the objective function of the optimization problem being flat close to the optimum.

As said above, there is no guarantee that a gradient-based method converges to the global optimum of the model parameters, but even obtaining a local optimum has traditionally been difficult. Many papers in the literature use numerical gradients (see [13]) for these methods when trying to solve the Heston calibration problem (which are less accurate and more computationally consuming), because no simple analytical gradients were available and the ones obtained with symbolic algebra packages from the expressions of the ChF were intractable. Prior to [5], the existing methods could be summarized as follows:

- Heuristic based models. Using the relationships outlined above, some works reduce the dimension of the optimization problem by assuming some values or relationships between the parameters from the observation of a specific volatility surface. For example, [8] sets v_0 to the short-term ATM implied variance, obtained by using a BS model, a heuristic further justified by [11], where the linearity between v_0 and the BS implied volatility was verified for short maturities (less than 2 months). Other heuristics used in the industry are $\kappa = \dfrac{2.75}{\tau}$ and setting $\overline{v}$ to the BS short-term volatility [7]. These assumptions may restrict the optimization problem domain and exclude the optimum;
- Stochastic methods. They are typically used in combination with deterministic search methods, such as the Nelder–Mead simplex method [14] and avoid the pitfalls of the gradient-based methods if the optimization problem is not convex. Some examples are used in [11], and differential evolution and particle swar are used in [15]. These methods are too computationally expensive for real-time use as [16], which employs GPU computations to calibrate options using an SV model called SABR, and it took 421.72 s to calibrate 12 instruments with a tolerance of 10^{-2} using 2 NVIDIA Geforce GTX470 GPUs.

In this work, we consider the analytical expression for the ChF provided in [5].

2.2. The Characteristic Function

For long-term maturities, [17] shows that the original ChF provided in [6] has discontinuities as u increases, which can lead to numerical problems. The source of these discontinuities was discussed in [18], and an alternative expression which was continuous in the full parameter space was presented in [19]. A more compact version of the ChF was later derived in [20] from the moment-generating function of the process. This expression has the benefit of having simpler analytical expressions of the gradient of the ChF than

in previous expressions, but it also presents discontinuities as u increases. Finally, an expression with both simple derivatives and continuity in the full-parameter domain was provided in [5]

$$\hat{f}(u) = \exp\left(-iur\tau + \frac{\kappa\bar{v}\rho\tau iu}{\sigma} - A + \frac{2\kappa\bar{v}}{\sigma^2}D\right), \tag{9}$$

where

$$
\begin{aligned}
\xi &:= \kappa + \sigma\rho iu, \\
d &:= \sqrt{\xi^2 + \sigma^2(u^2 - iu)}, \\
A_1 &:= (u^2 - iu)\sinh\frac{d\tau}{2}, \\
A_2 &:= \frac{d}{v_0}\cosh\frac{d\tau}{2} + \frac{\xi}{v_0}\sinh\frac{d\tau}{2}, \\
A &:= \frac{A_1}{A_2}, \\
B &:= \frac{de^{\kappa\tau/2}}{v_0 A_2}, \\
D &:= \log\frac{d}{v_0} + \frac{(\kappa - d)\tau}{2} - \log\left(\frac{d+\xi}{2v_0} + \frac{d-\xi}{2v_0}e^{-d\tau}\right).
\end{aligned} \tag{10}
$$

Further, its gradient with respect to the Heston model parameters $\boldsymbol{\theta} = (v_0, \bar{v}, \sigma, \kappa, \rho)^T$ is given by

$$\nabla\hat{f}(u) = \boldsymbol{h}(u)\hat{f}(u), \tag{11}$$

with $\boldsymbol{h}(u) = [h_{v_0}(u), h_{\bar{v}}(u), h_{\sigma}(u), h_{\kappa}(u), h_{\rho}(u)]^T$ and

$$
\begin{aligned}
h_{v_0}(u) &= -\frac{A}{v_0}, \\
h_{\bar{v}}(u) &= \frac{2\kappa}{\sigma^2}D + \frac{\kappa\rho\tau iu}{\sigma}, \\
h_{\sigma}(u) &= -\frac{\partial A}{\partial\rho} + \frac{2\kappa\bar{v}}{\sigma^2 d}\left(\frac{\partial d}{\partial\rho} - \frac{d}{A_2}\frac{\partial A_2}{\partial\rho}\right) + \frac{\kappa\bar{v}\tau iu}{\sigma}, \\
h_{\kappa}(u) &= -\frac{1}{\sigma iu}\frac{\partial A}{\partial\rho} + \frac{2\bar{v}}{\sigma^2}D + \frac{2\kappa\bar{v}}{\sigma^2 B}\frac{\partial B}{\partial\kappa} + \frac{\bar{v}\rho\tau iu}{\sigma}, \\
h_{\rho}(u) &= -\frac{\partial A}{\partial\sigma} - \frac{4\kappa\bar{v}}{\sigma^3}D + \frac{2\kappa\bar{v}}{\sigma^2 d}\left(\frac{\partial d}{\partial\sigma} - \frac{d}{A_2}\frac{\partial A_2}{\partial\sigma}\right) - \frac{\kappa\bar{v}\rho\tau iu}{\sigma^2},
\end{aligned}
$$

where the partial derivatives of A, A_2, B, and d are given in [5] and can be seen in Appendix A.

3. European Option Valuation and Calibration with SWIFT

In this section, we give a brief overview on the SWIFT method, originally developed for pricing European options in [1]. In this work, the method will be extended to European options' calibration. For sake of completeness, a section is devoted to the basic theory on Shannon wavelets.

3.1. Multi-Resolution Analysis and Shannon Wavelets

Consider the space of square-integrable functions, denoted by $L^2(\mathbb{R})$, where

$$L^2(\mathbb{R}) = \left\{ f : \int_{-\infty}^{+\infty} |f(x)|^2 dx < \infty \right\}.$$

Then, we can define a useful structure for function approximation called multi-resolution analysis (MRA). Let us start with a family of closed nested subspaces in $L^2(\mathbb{R})$

$$\ldots \subset \mathcal{V}_{-2} \subset \mathcal{V}_{-1} \subset \mathcal{V}_0 \subset \mathcal{V}_1 \subset \mathcal{V}_2 \subset \ldots,$$

where

$$\bigcap_{m\in\mathbb{Z}} \mathcal{V}_m = \{0\}, \qquad \overline{\bigcup_{m\in\mathbb{Z}} \mathcal{V}_m} = L^2(\mathbb{R}),$$

and

$$f(x) \in \mathcal{V}_m \iff f(2x) \in \mathcal{V}_{m+1}.$$

If these conditions are met, then there exists a function $\phi \in \mathcal{V}_0$, known as scaling function, that generates a family of orthonormal bases of $\mathcal{V}_m$, denoted $\{\phi_{m,k}\}_{k \in \mathbb{Z}}$

$$\phi_{m,k}(x) = 2^{m/2}\phi(2^m x - k).$$

These families allow one to approximate any $f \in L^2(\mathbb{R})$ with increasing resolution by means of the projection map $\mathcal{P}_m : L^2(\mathbb{R}) \to \mathcal{V}_m$

$$P_m f(x) = \sum_{k \in \mathbb{Z}} D_{m,k} \phi_{m,k}(x),$$

where $D_{m,k} = \langle f, \phi_{m,k} \rangle$, $\langle f, g \rangle = \int_{\mathbb{R}} f(x)\overline{g(x)}dx$, and $\overline{\cdot}$ is the complex conjugate operator.

Increasing the considered number of elements of the finite family will increase the resolution of the approximation, converging to a perfect representation when all the functions of the original family are used (see [21]). Apart from increasing the resolution by means of m, wavelets can be moved by means of k and stretched or compressed by means of m to represent the local properties of a function. A basic reference on wavelets is [22].

In this work, we employ Shannon wavelets [23], since they are regular and orthogonal functions with compact support in the Fourier domain. The regularity, as opposed to the Haar family used in [24], gives us much better accuracy. The rapid decay of the scaling function in the Fourier domain replicates the behaviour of the ChF of the heavy-tail processes that we encounter in finance. A set of Shannon scaling functions $\phi_{m,k}$ in the subspace $\mathcal{V}_m$ is defined as

$$\phi_{m,k}(x) = 2^{m/2}\frac{\sin(\pi(2^m x - k))}{\pi(2^m x - k)} = 2^{m/2}\phi(2^m x - k), \quad k \in \mathbb{Z}, \tag{12}$$

where

$$\phi(z) = \operatorname{sinc}(z) = \begin{cases} \dfrac{\sin(\pi z)}{\pi z}, & \text{if } z \neq 0, \\ 1, & \text{if } z = 0, \end{cases}$$

is the scaling function, also called cardinal sine function.

The following lemma about the bound on the error of the orthogonal projection $\epsilon_m(x) := f(x) - P_m f(x)$ is provided in [3].

Lemma 2 (Lemma 3 of [3]). *Let $\epsilon_m(x) := f(x) - P_m f(x)$ be the point-wise approximation error due to the projection of f into V_m. Then*

$$|\epsilon_m(x)| \leq H(2^m \pi), \tag{13}$$

where

$$H(\xi) := \frac{1}{2\pi} \int_{|u| > \xi} \left| \hat{f}(u) \right| du,$$

is the normailzed mass of the two-side tails of $\hat{f}$.

3.2. SWIFT Method

The SWIFT method belongs to the class of Fourier inversion methods for pricing European options within the discounted expected payoff formula (1). The density function f of (1) is replaced by a finite combination of Shannon wavelets, and the coefficients of the approximation are computed from its ChF. The overall process provides an efficient way to obtain the value of an option, and it can be summarized, as in [1,3], by a set of consecutive approximation steps, which are described below.

- Wavelet projection: the function f is replaced by its Shannon wavelet projection at scale $m \in \mathbb{Z}$,

$$f(y|x) \approx f_1(y|x) := P_m f(y|x) = \sum_{k \in \mathbb{Z}} D_{m,k}(x)\phi_{m,k}(y), \qquad (14)$$

with $D_{m,k}(x) := \langle f(\cdot|x), \phi_{m,k} \rangle$.

- Series truncation: the set of values of k involved in the sum of expression (14) is reduced to a finite interval $[k_1, k_2]$,

$$f_1(y|x) \approx f_2(y|x) = \sum_{k=k_1}^{k_2} D_{m,k}(x)\phi_{m,k}(y). \qquad (15)$$

It is important to notice that the first approximation lets us express

$$f(2^{-m}k|x) \approx f_1(2^{-m}k|x) = 2^{m/2}D_{m,k}(x), \qquad (16)$$

which quickly justifies that, for any given x, the density coefficients vanish as $|k|$ increases, because the mass at the tails of f tends towards 0. It is also worth noting that increasing m will result in this mapping being less favorable. That is, for each k, $D_{m,k}$ will be bounded by a point closer to the center of the density function, potentially requiring an increase in the range of values for k in interval $[k_1, k_2]$.

Remark 4. *From this point onward, a symmetric interval $[1 - \eta, \eta]$ will be considered for both convenience and consistency with the code implementation.*

- Density coefficients approximation: the integral required to compute $D_{m,k}$ is replaced by an approximation $D_{m,k}^*$, as will be shown in Section 3.2.1

$$f_2(y|x) \approx f_3(y|x) = \sum_{k=1-\eta}^{\eta} D_{m,k}^*(x)\phi_{m,k}(y), \qquad (17)$$

We can then define $V_{m,k} := \int_{\mathbb{R}} \phi_{m,k}(y)v(y, T)dy$, and substitute $f_3(y|x)$ into expression (1), obtaining

$$v(x, t) \approx v_3(x, t) := e^{-r\tau} \sum_{k=1-\eta}^{\eta} D_{m,k}^*(x)V_{m,k}dy,$$

For European options, one can instead express it in terms of the strike-free payoff by defining $U_{m,k} := \dfrac{V_{m,k}}{K} = \int_{\mathbb{R}} \phi_{m,k}(y)g(y)dy$, obtaining

$$v_3(x, t) := Ke^{-r\tau} \sum_{k=1-\eta}^{\eta} D_{m,k}^*(x)U_{m,k}dy. \qquad (18)$$

- Payoff coefficients approximation: the integral required to compute $U_{m,k}$ is approximated in an analogous way as the integral to compute the density coefficients, and $U_{m,k}$ is replaced by an approximation $U_{m,k}^*$, as will be shown in Section 3.2.1

$$v_3(x, t) \approx v_4(x, t) = e^{-r\tau} \sum_{k=1-\eta}^{\eta} D_{m,k}^*(x)U_{m,k}^*. \qquad (19)$$

These coefficients can be precomputed when initializing the SWIFT procedure and shared across different strikes and maturities, saving computation time.

3.2.1. Density and Payoff Coefficients Approximation

Density and payoff coefficients calculation rely on the approximation of $\mathrm{sinc}(x)$ by a finite combination of cosines (all the details can be found in [1,3] and the references therein). As a result of this approximation, a new parameter called J appears. This parameter will be labeled J_d and J_p to denote density and payoff coefficients, respectively. Following the notation of [3], the density coefficients are given by

$$D_{m,k}^* = \frac{2^{m/2}}{J_d} \sum_{j=1}^{J_d} \Re\left(\hat{f}\left(u_j^d 2^m; x\right) e^{iku_j^d} \right),\tag{20}$$

where $\Re$ denotes the real part, $u_j^d = \frac{\pi}{2J_d}(2j-1)$, and the payoff coefficients are given by

$$U_{m,k} \approx U_{m,k}^*(-c,c) := \frac{2^{m/2}}{J_p} \sum_{j=1}^{J_p} \Re\left(e^{iku_j^p} I_j(c) \right),$$

where

$$I_j(c) := \int_{|y| \leq c} g(y) e^{-iu_j^p y} dy = H(c) - H(-c),\tag{21}$$

and

$$H(y) := -ie^{iu_j^p y}\left(\frac{1}{u_j^p} - \frac{e^y}{-i + u_j^p} \right), \quad u_j^p = \frac{\pi}{2J_p}(2j-1).\tag{22}$$

We can see that the value of $I_j(c)$ is periodic on c. In general, all the approximations to $\mathrm{sinc}(x)$ used in the SWIFT method are periodic, which can give rise to boundary issues and undervaluation of option prices when the option strikes approach the boundary of $(-c,c)$. This issue also appears in the COS option pricing method [25], another Fourier-transform-based option-pricing method closely related to the SWIFT method, and is discussed in [3]. In that work, the authors use the independence between the parameter c regulating the payoff integral domain and the parameter η regulating the wavelet series truncation to carefully choose a value for c to avoid this problem. An iterative method to choose appropriate values for m, η, J_d, and J_p is provided in [1,3].

As most operators used in the SWIFT method are linear, one can easily obtain an expression for the option price gradient that will be used for calibration. In particular, the only dependence the European option price formula has on the model parameters vector $\boldsymbol{\theta}$ appears in the term $D_{m,k}^*$, so we can simply define,

$$D_{m,k}^{*(n)}(x; \boldsymbol{\theta}_i) := \frac{\partial^n D_{m,k}^*(x; \boldsymbol{\theta}_i)}{\partial \boldsymbol{\theta}_i^n} = \frac{2^{m/2}}{J_d} \sum_{j=1}^{J_d} \Re\left(\frac{\partial^n \hat{f}\left(u_j^d 2^m; x; \boldsymbol{\theta}_i \right)}{\partial \varsigma^n} e^{iku_j^d} \right),\tag{23}$$

and replace it in expression (19) to obtain the expression of the gradient.

3.3. Speeding Up the SWIFT Method

We elaborate on several enhancements of the SWIFT method in terms of efficiency, either on the pricing or during the calibration process. In Section 3.3.1, how the density and payoff coefficients can be computed by means of the Fast Fourier Transform is explained (in all the numerical examples in this article, the C library FFTW will be used to compute the FFT [26]) (FFT). Section 3.3.2 is devoted to the adaptation of the SWIFT pricing machinery for multiple strikes valuation. Those two transformations were already pointed out in [1]. Moreover, when the vector of strikes meets a certain property, then the calibration can be carried out in a two-stage procedure detailed in Section 3.3.3. Finally, Section 3.3.4 describes another improvement in regard to the option price gradient calculation.

3.3.1. Fast Computation of the Density and Payoff Coefficients

We start with a general expression of the summation term that appears in both the density and payoff coefficients approximation

$$f_k = \sum_{j=1}^{J} g_j e^{i \frac{2j-1}{2J} \pi k} \tag{24}$$

We can extend this by defining $g_j = 0$ for $j = 0$ and $J < j < 2J$ and take the j-independent terms outside of the summation, obtaining,

$$f_k = e^{\frac{-i\pi k}{2J}} \sum_{j=0}^{2J-1} g_j e^{i\frac{2j\pi k}{2J}} \tag{25}$$

This last summation expression coincides with the Discrete Fourier Transform (DFT) of length $2J$ of $\{g_j\}$, and the computation of all the values f_k can then be speeded up by the FFT implementation.

Remark 5. *Note that computing the density or payoff coefficients imposes a restriction on the wavelet series truncation parameter $2\eta < J$.*

3.3.2. Valuation with Multiple Strikes

A key property of the SWIFT method is that, given a scale of approximation m, the payoff and density coefficient associated to each wavelet $\phi_{m,k}$ can be computed through two FFTs (one for all the density coefficients, and one for all payoff coefficients). Without this property, the SWIFT computation speed would not be competitive with other numerical option pricing methods [27].

In the option calibration problem, one usually needs to consider the option prices of several options at different strikes. In this specific case, if one were to compute the option prike of M options at strikes $\boldsymbol{K} := (K_1, \dots, K_M)^T$, then the formulation proposed in expression (19) would need to recompute the density and payoffs coefficients for every strike K_i. This involves evaluating the ChF $\eta \cdot J_p \cdot M$ times, an operation which, for the Heston model, is more costly than evaluating the strike-free payoff function, or its integral. As stated in [1] one can improve the computation time of option pricing for multiple strikes when $\hat{f}(u; x) = \hat{f}(u)e^{-iux}$, a property present in both Lévy and Heston models.

As stated in [3], let us start from expression (19), and considering the previously mentioned vector of strikes $\boldsymbol{K}$, with its associate vector of initial states $\boldsymbol{x} := (\log(S_0/K_1), \dots, \log(S_0/K_M))^T$. We can then substitute the density coefficients approximation (20) into the option price expression (19) and interchange the two resulting summations, obtaining

$$v_4(\boldsymbol{x}, t) := e^{-r\tau} \boldsymbol{K} \sum_{k=1-\eta}^{\eta} \Re\left\{ \sum_{j=1}^{J_d} \hat{f}(u_j^d 2^m; \boldsymbol{x}) e^{iu_j^d k} U_{m,k}^*(-c, c) \right\} \tag{26}$$

$$= e^{-r\tau} \boldsymbol{K} \sum_{j=1}^{J_d} \Re\left\{ \hat{f}(u_j^d 2^m; \boldsymbol{x}) \left[\sum_{k=1-\eta}^{\eta} U_{m,k}^*(-c, c) e^{iu_j^d k} \right] \right\} \tag{27}$$

$$= e^{-r\tau} \boldsymbol{K} \sum_{j=1}^{J_d} \Re\left\{ \hat{f}(u_j^d 2^m) e^{-iu_j^d 2^m x} \left[\sum_{k=1-\eta}^{\eta} U_{m,k}^*(-c, c) e^{iu_j^d k} \right] \right\} \tag{28}$$

$$= e^{-r\tau} \boldsymbol{K} \sum_{j=1}^{J_d} \Re\left\{ \hat{f}(u_j^d 2^m) e^{-iu_j^d 2^m x} \tilde{U}_j(-c, c) \right\}. \tag{29}$$

where,

$$\tilde{U}_j(-c,c) := \sum_{k=1-\eta}^{\eta} U_{m,k}^*(-c,c)e^{iu_j^d k}.$$

The original formulation from expression (19) requires the following computations:

- For each of the M strikes:
    - 1 FFT of length $2J_d$ to compute 2η density coefficients;
    - J_d evaluations of the ChF $\hat{f}(u_j^d 2^m; x)$;
- 1 FFT of length $2J_p$ to compute 2η payoff coefficients;
- J_p evaluations of the strike-free payoff integral $I_j(c)$ defined in expression (21).

The payoff computations are independent of the strike price and can be computed only once, and be reused for all strikes.

On the other hand, the alternative formulation provided in expression (29) requires:

- For each of the M strikes:
    - J_d evaluations of the x-dependent term of the ChF $e^{-iu_j^d x}$;
- J_d evaluations of the x-independent ChF $\hat{f}(u_j^d 2^m)$;
- 2 FFT of lengths 2η and $2J_p$ to compute the J_d values of $\tilde{U}_j(-c,c)$;
- J_p evaluations of the strike-free payoff integral $I_j(c)$ defined in expression (21).

The required values of $\hat{f}(u_j^d 2^m)$, $\tilde{U}_j(-c,c)$, and the values of $I_j(c)$, required to compute the latter, can be computed only once and be reused for all strikes.

Remark 6. *In general, whenever the dependency on x in $\hat{f}(u; x)$ can be easily isolated and is cheap to compute, one can benefit from the alternative formulation proposed in this section.*

The computation of the ChF tends to be more expensive than the computation of the payoff integral, so a SWIFT implementation through expression (29) tends to outperform the one through expression (19) when several strike prices are involved. A discussion on the benefits of using a formulation equivalent to the one provided in (29) for multiple strikes appears in [1], where it is shown that is possible to define $F_j := \hat{f}(u_j^d 2^m)$ and compute its J_d required values once in advance, and reuse them for all strikes.

3.3.3. Fixed Set of Strikes

Let us consider

$$G_j := \begin{cases} F_j, \tilde{U}(-c,c) & \text{for} \quad j \leq J_d, \\ 0, & \text{for} \quad J_d < j \leq 2J_d. \end{cases} \tag{30}$$

Then, expression (29) can be rearranged as

$$v_4(x,t) = e^{-r\tau} K \Re \left\{ e^{\frac{\pi i 2^m x}{J_d}} \cdot \sum_{j=1}^{2J_d} G_j e^{-\frac{2\pi i j 2^m x}{2J_d}} \right\} \tag{31}$$

If one selectively chooses the values of the strikes K_l, so that $2^m x_l$ is an integer number, this computation can be sped up by the use of an FFT algorithm. If one chooses $x_k := \dfrac{2k - J_d}{2^{m+1}}$, then expression (31) becomes

$$v_4(x,t) = e^{-r\tau} K Re \left\{ e^{\frac{\pi i 2^m x}{J_d}} \sum_{j=1}^{2J_d} G_j e^{-\frac{2\pi i j k}{2J_d}} e^{\pi i j} \right\} = e^{-r\tau} K Re \left\{ e^{\frac{\pi i 2^m x}{J_d}} \sum_{j=1}^{2J_d} \tilde{G}_j e^{-\frac{2\pi i j k}{2J_d}} \right\}, \tag{32}$$

where

$$\tilde{G}_j := G_j e^{\pi i j} = G_j (-1)^j. \tag{33}$$

Remark 7. *Note that, as with other FFT-based computations presented in this work, this approach imposes a bound $M \le J_d$ on the number of different strikes that can be computed with the FFT.*

If one considers the domain $\mathcal{D}$ of $x = \log(S_t/K)$, this approach allows pricing options in a symmetrical boundary $(-\frac{J_d}{2^{m+1}}, \frac{J_d}{2^{m+1}}) \in \mathcal{D}$ at J_d uniformly distributed points at distance 2^m. We cannot usually choose the strike prices at which to price the options, particularly not when calibrating a model with real market data, as only a limited set of strike values are listed on any exchange market, but this method could be used to quickly compute the option prices of an already calibrated model at a grid of points that could be tuned by the choice of m and J_d. Then, the option prices at any intermediate strike could be interpolated with the help of a derivative-free spline (or, if the derivative with respect to K in expression (32) preserves the same speed properties, with the help of any spline method that uses derivatives).

3.3.4. Option Price Gradient

The option price gradient must be computed during the calibration process that will be presented in Section 4. All the aforementioned techniques can be applied to the option price gradient. We can also enumerate three more speed-up properties,

- The value of $e^{-iu_j^d 2^m x_l}$ can be reused for the price as well as for the gradient computations;
- If the parameters of the SWIFT method, are not changed during the gradient descent used in the calibration problem, then the values of both $\tilde{U}_j(-c, c)$ and $e^{-iu_j^d 2^m x_l}$ can be reused throughout all the calibration steps;
- We can reuse the values of $\hat{f}(u_j^d 2^m)$ from the price computation to compute the gradient.

Therefore, combining all the speed properties above, when solving a gradient-based calibration problem, we only need to first compute $\tilde{U}_j(-c, c)$ and $e^{-iu_j^d 2^m x_l}$, and then, in each gradient-descent step, one can simultaneously calculate both the price and the gradient of all strikes by computing once for each $j \in [1, J_d]$ the values of $\hat{f}(u_j^d 2^m)$ and $h(u_j^d 2^m)$.

4. Calibration

Calibrating a model for the asset underlying the option is a sophisticated procedure that requires highly efficient numerical methods. In particular, the pricing of the options used for calibration should be carried out by means of an accurate, fast and robust valuation method. In this work, we calibrate the Heston model by means of the SWIFT method, and compare it with the Heston's pricing method of [5], which we have written in a more compact form in Lemma 1. It is worth noting that the choice of a specific objective function will have an impact on how accurately the model of the underlying asset that we will obtain through callibration will describe different real market scenarios [28]. For comparison sake, the same one as in [5] will be used.

Let $V^*(x_i, \tau_i)$ be the market price of a European call option and $V(\boldsymbol{\theta}; x_i, \tau_i)$ be the price at the same strikes and maturities obtained by using either the SWIFT method or the Heston pricing formula in expression (8) of Lemma 1. Let us also assume that we use a set of n different options to calibrate the model, so that $i \in [1, n] \subset \mathbb{Z}$. Then the calibration of the model is defined as the minimization problem

$$min_{\boldsymbol{\theta} \in \mathbb{R}^5} f(\boldsymbol{\theta}), \qquad\qquad f(\boldsymbol{\theta}) := \frac{1}{2}||\boldsymbol{r}(\theta)||^2 = \frac{1}{2}\boldsymbol{r}^T(\boldsymbol{\theta})\boldsymbol{r}(\boldsymbol{\theta}), \tag{34}$$

where $r(\theta)$ is the n-dimensional vector of the residuals obtained when pricing the options considered for calibration using the model parameters. That is

$$r(\theta) := [r_1(\theta), \ldots, r_n(\theta)]^T, \qquad r_i(\theta) := V(\theta; x_i, \tau_i) - V^*(x_i, \tau_i), \qquad i = 1, \ldots, n. \qquad (35)$$

If we calculate the Jacobian of r, it gives us

$$J := \nabla_\theta r^T = \nabla_\theta V(\theta; x, \tau), \qquad (36)$$

where

$$J_{ji} = \left(\frac{\partial r_i}{\partial \theta_j} \right) = \left(\frac{\partial V(\theta; x_i, \tau_i)}{\partial \theta_j} \right). \qquad (37)$$

The Hessian matrix of the residual element r_i reads

$$H(r_i) := \nabla_\theta \nabla_\theta^T r_i = \nabla_\theta \nabla_\theta^T V(\theta; x_i, \tau_i), \qquad (38)$$

where

$$H_{jk}(r_i) = \left(\frac{\partial^2 r_i}{\partial \theta_j \partial \theta_k} \right) = \left(\frac{\partial^2 V(\theta; x_i, \tau_i)}{\partial \theta_j \partial \theta_k} \right). \qquad (39)$$

Then, the gradient and Hessian matrix of the objective function defined by expressions (34) and (35) are

$$\nabla_\theta f(\theta) = Jr, \qquad (40)$$

$$\nabla_\theta \nabla_\theta^T f(\theta) = JJ^T + \sum_{i=1}^{M} r_i H(r_i). \qquad (41)$$

We solve the optimization problem (34) by means of the LM (for the implementation, we use the LEVMAR C package [29] as well as the LAPACK linear algebra package [30]) method. This method is as a blend of gradient descent (GD) and Gauss–Newton (GN) iteration, depending on whether the current guess is close to or far from the optimum. The exact expression of the step $\Delta\theta$ is

$$\Delta\theta = (JJ^T + \mu I)^{-1} \nabla_\theta f(\theta), \qquad (42)$$

where I is the identity matrix and $JJ^T + \mu I$ substitutes the Hessian matrix used in the Newton method. When the current guess is far from the optimum, a large value is given to μ, so that

$$\Delta\theta \approx \Delta\theta^{(\text{SD})} := (\mu I)^{-1} \nabla_\theta f(\theta), \qquad (43)$$

and a small step of a steepest-descent method is taken. When the current guess is close to the optimum, a small value is given to μ, so that

$$\Delta\theta \approx \Delta\theta^{(\text{GN})} := (JJ^T)^{-1} \nabla_\theta f(\theta), \qquad (44)$$

and the Hessian usually used in the Newton method is replaced by its GN approximation. This approximation is reliable if either r_i or $H(r_i)$ are small, and [5] justifies its usage by conjecturing that f is nearly linear, a condition that guarantees the latter. We should note that even if f were not linear, then LM should only use small values of μ when $|r|$ is small at the current step of the optimization problem.

The iterative algorithm implemented for the LM method stops when, at a certain iteration n, any of the following criteria are fulfilled,

$$|\boldsymbol{r}_n| \leq \epsilon_1, \tag{45}$$

$$|\boldsymbol{J}_n|_\infty \leq \epsilon_2, \tag{46}$$

$$\frac{|\nabla \boldsymbol{\theta}_n|}{|\boldsymbol{\theta}_n|} \leq \epsilon_3. \tag{47}$$

The first stopping criteria (45) is fulfilled when the objective function defined by expressions (34) and (35) has reached a value closer to zero than the prescribed tolerance. It is only when the method stops due to this criteria that we will consider that the model has been properly calibrated. The second criteria (46) corresponds to a flat gradient, and the third (47) corresponds to a stagnating update (this last one has never happened while testing the convergence during the Heston model calibration).

5. Numerical Results

In this word, the SWIFT (the code implemented for this work can be consulted in the following Github public repository: https://github.com/eudaldrg/SWIFTOptionCalibration accessed on 2 March 2020) method is used to calibrate a Heston model with European call options price data at different strikes and maturities, and it will be compared to the pricing and calibration method based on expression (8) proposed by [5], which, for the sake of readability, will be called Cui pricer (CP). CP will be implemented using a Gauss–Legendre quadrature with 64 nodes for its numerical integration step. The upper limit of the integral will be truncated, whenever possible, at $\overline{u} = 200$, but will be adjusted if necessary. The calibration process will consist of applying an LM method to the objective function defined in expression (34). The SWIFT method will be implemented using the ChF expression and its derivatives, provided in [5]. We perform a wide variety of tests that can be summarized as follows:

- Stress tests: the CP and SWIFT methods will be tested with several combinations of extreme strikes (ATM and deep ITM and OTM) as well as with long-term and short-term maturities, to detect any possible limitation or numerical issue in a wide usage range;
- Speed (The computations were performed on a 64-bit Ubuntu 18.04.4 LTS with a 3.70GHz Intel Core i7-8700K processor and 62.8 gigabytes of ram.) tests: the option calibration speeds for the regular SWIFT method (defined by expression (19)) and the one devised to quickly compute several option prices with different strike and the same maturity (defined by expression (29), which will be denoted KSWIFT), will be compared against CP for three different strike and expiry sets to check whether the multiple-strike alternative formulation is necessary to obtain a competitive option calibration method. These scenarios will represent:
  - A single expiry and multiple strikes;
  - A fixed number of maturities and a fixed number of strikes per maturity;
  - Different expiries for each strike.
  When computing options with more than one different strike, a combination of OTM and ITM options will be used to provide an heterogeneous sample of contracts. Similarly, when more than one maturity is considered, a sample of long- and short-term expiries will be used;
- Realistic convergence tests: as in [5], convergence of the method will be tested for realistic model parameters representative of long-dated Foreign Exchange (FX), interest rate, and equity options, as they are relevant and, according to [31], challenging for simulations of the Heson model.

Several sets of Heston parameters will be used for the different numerical tests and are presented in Table 1. The last three sets of parameters are representative of long-term FX, interest rate, and equity options, respectively [32].

Table 1. Set of Heston parameters used in the numerical tests.

Name	κ	$\overline{v}$	σ	ρ	ν_0
$\boldsymbol{\theta}^{(1)}$	3	0.1	0.25	-0.8	0.08
$\boldsymbol{\theta}^{(2)}$	1.5768	0.0398	0.0175	-0.5711	0.0175
$\boldsymbol{\theta}^{(FX)}$	0.5	0.04	1	-0.9	0.04
$\boldsymbol{\theta}^{(IR)}$	0.3	0.04	0.9	-0.5	0.04
$\boldsymbol{\theta}^{(EQ)}$	1	0.09	1	0.04	0.09

Remark 8. $\boldsymbol{\theta}^{(1)}$ *is obtained from [5] and* $\boldsymbol{\theta}^{(2)}$ *is a plausible set of parameters proposed by us. It may not be representative of any real-world market, but it is only used as our objective value in our speed tests. In Section 5.2, we will use an initial guess of* $\boldsymbol{\theta}_0^{(2)} = (1.5768, 0.0398, 0.5751, -0.5711, 0.0175)^T$.

5.1. Stress Tests

Deep ITM and OTM call options are priced together with ATM call options for long- and short-term maturities using the SWIFT method at two different scales of approximation ($m = 3$ and $m = 7$) and the CP method. The time until maturity τ is given in years. Thus, the expiries of 0.04 attempt to simulate a situation of around two weeks until expiration of the option contract.

Table 2 presents the pricing results. We call V_{SW}^3 and V_{SW}^7 to the prices given by the SWIFT method at scales of approximation $m = 3$ and $m = 5$, respectively, while V_{CP} refers to the price obtained with the CP method. Both methods run into numerical issues with either extremely large or extremely small expiries, provided no other changes are performed in the methods.

Table 2. Set of Heston parameters used in the numerical tests.

Parameters	S	K	τ	V_{SW}^3	V_{SW}^7	V_{CP}
$\boldsymbol{\theta}^{(1)}$	100	50	45	65.565	nan	nan
$\boldsymbol{\theta}^{(1)}$	100	100	45	46.911	nan	nan
$\boldsymbol{\theta}^{(1)}$	100	200	45	27.198	nan	nan
$\boldsymbol{\theta}^{(1)}$	100	50	0.04	44.221	50.000	50.000
$\boldsymbol{\theta}^{(1)}$	100	100	0.04	0.380	1.045	1.046
$\boldsymbol{\theta}^{(1)}$	100	200	0.04	0	0	$1.079 \cdot 10^{-3}$

- CP and V_{SW}^7 did not produce number (nan) results when evaluating very long expiries. Looking at the option price execution with the integrated debugger of the GDB compiler [33] showed that expression (9) runs into numerical overflow when the exponent $\frac{d\tau}{2}$ of its hyperbolic functions is big enough (the same error happened when using the original expression provided in [20]). In most of the tests above, the overflow could be avoided when carefully setting an appropriate value for the upper bound $\overline{u}$ of integral in expression (8), and by using a smaller value of the scale m. The error can also be avoided by selecting the ChF expression provided in [19] (we use this later on, and we denote the obtained prices by V_{SH} and present the results in Table 3);
- The SWIFT method at scale $m = 3$ tends to underprice short expiry options. After checking the SWIFT parameters obtained through the parameter choice method defined in Section 3, it was observed that the initial value for η, obtained by simply using the cumulant expression proposed in [1], resulted in a truncated Shannon

wavelet expansion that did not cover a sufficient domain of the density function $f(y|x)$. A dynamical choice of the parameter η based on the calculation of the area underneath the curve of the density function, as described in [1], can avoid this issue. Increasing the value of m also fixes the problem;

- None of the methods can handle the deep OTM option with a short expiry. The expected value should be close to but bigger than 0, as there are only 10 trading days to expiry and the price of the underlying should increase 50% so that the option contract would not expire and become worthless. CP value seems too high and, in fact, when moving the value of $\overline{u}$ in the interval $[100, 400]$, the price never clearly converges to a certain value, and it can give higher estimates for $\overline{u} > 200$ than 1.079×10^{-3}, or even negative values. Changing the ChF expression does not fix this issue. SWIFT consistently gives it a price of 0. The contribution that makes the price different than zero probably lies in the tails of the distribution function, and one would require a really big value of c so that a point with a positive payoff is even considered in expression (29).

Table 3 shows some results either selecting an appropriate value $\overline{u}$ or keeping $\overline{u}$ equal to 200 and using the ChF from [19]. The column $\overline{u}$ indicates the value at which CP integral is truncated, and the option price obtained is shown in column V_{CP}. Column V_{SH} shows the price obtained when keeping $\overline{u} = 200$ but implementing CP using the ChF provided in [19]. The last row is an example of the negative values obtained in the deep OTM short-term call.

Table 3. Results for different $\overline{u}$ and/or using the ChF from [19].

Parameters	S	K	τ	$\overline{u}$	V_{CP}	V_{SH}
$\boldsymbol{\theta}^{(1)}$	100	50	45	6	65.565	65.565
$\boldsymbol{\theta}^{(1)}$	100	100	45	6	46.911	46.911
$\boldsymbol{\theta}^{(1)}$	100	200	45	6	27.198	27.198
$\boldsymbol{\theta}^{(1)}$	100	50	0.04	200	50.000	50.000
$\boldsymbol{\theta}^{(1)}$	100	100	0.04	200	1.046	1.046
$\boldsymbol{\theta}^{(1)}$	100	200	0.04	300	$-1.174{\cdot}10^{-5}$	$1.079{\cdot}10^{-3}$

As it has been shown so far, a crucial step of the calibracion process is the selection of $\overline{u}$ for the CP method and m for the SWIFT method. A method to set an optimal value $\overline{u}$ is not provided in [5], and it is, therefore, a matter of trial and error, since it must be manually determined when changing the time to expiry of the options one wants to price. As for the SWIFT method, we can use the iterative parameter choice provided in [3]. In particular, a suitable scale of approximation m can be selected by means of Lemma 2. Once the level of approximation m is fixed, the parameter η can be adaptively calculated in order to determine the wavelet series truncation more accurately.

5.2. Speed Tests

The calibration speed has been tested for three different sets of strike prices and maturities, which are available in Appendix B. In order to be sure that the calibration problem was properly converging, we chose $\boldsymbol{\theta}^{(2)}$ as the objective value for the Heston parameters. When testing each set, we perform the following actions,

- Generate option price values for each strike-expiry pair using $\boldsymbol{\theta}^{(2)}$ as input. For this step, we used the SWIFT method (we also generated them using the CP method and checked that the difference between both results stayed under 10^{-7});
- Chose an initial guess for the calibration problem $\boldsymbol{\theta}_0^{(2)}$;

- Solve the calibration problem with the desired method using $\theta_0^{(2)}$ as the initial guess and use the strike–expiry pairs and the prices obtained in the first step as inputs.

Set 2 was obtained from the code provided in [5], and represents a total of 40 options, distributed in eight different maturities, with each maturity having five different strikes. Set 1 and set 3 are extreme cases derived from set 2, in order to test the best and worst calibration points distribution for KSWIFT. Set 1 has the same 40 different strike prices than set 2, but only a single maturity. What we denote as set 3 is not really a different dataset, since it consists of running the calibration problem with set 2 and preventing the KSWIFT algorithm from applying the speed-up techniques discussed in Section 3.3.2. For this reason, in Table 4, set 3 contains values only for KSWIFT (as set 3 is equivalent to set 2 for the other two calibration methods).

Table 4. Iterations, time needed to calibrate each speed scenario and objective function value reached. I refers to the number of iterations that LM requieres until it stops, and ϵ_1 corresponds to the LM first stopping criteria (see Section 4), which refers to the objective function final value.

Strike and Maturities Set	Heston Parameters	Method	Time (Seconds)	I	ϵ_1
Set 1	$\theta^{(2)}$	SWIFT	6.9	10	$3.932 \cdot 10^{-11}$
Set 1	$\theta^{(2)}$	KSWIFT	$4.5 \cdot 10^{-3}$	10	$3.932 \cdot 10^{-11}$
Set 1	$\theta^{(2)}$	CP	$4.6 \cdot 10^{-2}$	10	$3.932 \cdot 10^{-11}$
Set 2	$\theta^{(2)}$	SWIFT	35.9	13	$1.002 \cdot 10^{-12}$
Set 2	$\theta^{(2)}$	KSWIFT	$5.0 \cdot 10^{-2}$	13	$1.002 \cdot 10^{-12}$
Set 2	$\theta^{(2)}$	CP	$6.3 \cdot 10^{-2}$	13	$1.002 \cdot 10^{-12}$
Set 3	$\theta^{(2)}$	KSWIFT	$1.7 \cdot 10^{-1}$	13	$1.002 \cdot 10^{-12}$

The values for KSWIFT and CP have been averaged over 100 executions of the calibration to provide a good estimate of the required calibration time. It can be seen that regular SWIFT is orders of magnitude slower without averaging the required time over several executions. Hence, the multiple-strike alternative formulation presented in Section 3.3.2 and all the speed-up techniques discussed through Section 3 are necessary to provide a competitive method that can be used for real-time model updating.

KSWIFT performance is comparable to CP for set 2, an order of magnitude faster for set 1, and an order of magnitude slower for set 3. It can be argued that both set 1 and set 3 are extreme cases that are not really relevant for real option trading situations. One would rarely use a single strike per expiry to calibrate an option pricing model, and using data from a single expiry only seems reasonable when trading a single-option expiry (in this case, one could benefit from the speed properties of KSWIFT on scenarios like set 1). According to [5], a reasonable calibration scenario consists of using option prices from strikes at 0%, ±25%, and ±50% BS delta (derivative of the option price with respect to the underlying price value. It has a closed analytical expression for European BS options).

The calibration time of KSWIFT and CP is about 0.05 seconds, which seems sufficient for real-time model updating to provide market information to a human trader. In a more computationally demanding trading environments, like high-frequency trading, neither KSWIFT nor CP would be competitive enough.

Remark 9. *All the single expiry tests (the first three tests on Table 4) converged to an approximated value different than $\theta^{(2)}$ but approximated all the option prices properly. Using different initial guesses led to different approximated values, which minimized the objective function. It would be interesting to see whether this is a property of the Heston distribution (that is, it has at least a degree of freedom when defined from option prices in a single expiry) or due to the specific scenario being tested.*

5.3. Realistic Convergence Tests

We use the same procedure as in previous section to solve several different calibration problems. For each case, we use one of the proposed realistic parameter sets as objective value and generate option prices for each strike–expiry pair. Then, for each objective parameter set, 100 different initial guesses are generated. Each component of the initial parameters guess is drawn uniformly and randomly within $\pm 10\%$ distance of the optimal value. According to [5], this is representative of real option calibration as, usually, the initial guess used for a certain calibration problem is the last available parameter estimation. If the calibration is updated fast enough, it is expected that the initial guess will be this close to the optimum. The maturities used in [5] are not available, so, for these tests, the strike–expiry set 2 will be used.

As can be seen in Table 5, even under challenging parameters setups representative of real option trading, KSWIFT is able to provide accurate estimations of the Heston parameters, taking, on average, a computation time of hundreds of milliseconds. These results, in terms of both speed and accuracy, are comparable to the tests in [5], so it is concluded that KSWIFT is as efficient as CP for real market scenarios. Further, if we take into account the robustness of KSWIFT in terms of the a priori knowledge of its parameters, as stated in Section 3.2, we conclude that KSWIFT is a very competitive method for calibration.

Table 5. Convergence statistics averaged over 100 calibrations. x^a refers to the calibration problem's estimation of variable x^a. For example, $|\kappa^a - \kappa^*|$ refers to how close LM approximation of κ was to the real value.

	$\boldsymbol{\theta^{(\textbf{FX})}}$	$\boldsymbol{\theta^{(\textbf{IR})}}$	$\boldsymbol{\theta^{(\textbf{EQ})}}$
$|\kappa^a - \kappa^*|$	$6.640 \cdot 10^{-4}$	$2.657 \cdot 10^{-4}$	$1.160 \cdot 10^{-3}$
$|\bar{v}^a - \bar{v}^*|$	$1.547 \cdot 10^{-4}$	$1.321 \cdot 10^{-5}$	$1.746 \cdot 10^{-5}$
$|\sigma^a - \sigma^*|$	$1.978 \cdot 10^{-3}$	$2.248 \cdot 10^{-4}$	$3.725 \cdot 10^{-4}$
$|\rho^a - \rho^*|$	$2.649 \cdot 10^{-4}$	$1.365 \cdot 10^{-5}$	$8.661 \cdot 10^{-6}$
$|v_0^a - v_0^*|$	$3.629 \cdot 10^{-5}$	$4.790 \cdot 10^{-6}$	$8.339 \cdot 10^{-6}$
Iterations	14	6	7
Time (seconds)	$3.3 \cdot 10^{-1}$	$1.9 \cdot 10^{-1}$	$2.0 \cdot 10^{-1}$
ϵ_1	$2.867 \cdot 10^{-11}$	$2.030 \cdot 10^{-11}$	$3.643 \cdot 10^{-11}$

6. Conclusions and Future Research

We have investigated the problem of calibrating the Heston model, which belongs to the class of stochastic volatility models. An extension of the SWIFT method has been provided in this work for European options calibration, along with novel speed-up techniques, which can radically improve the performance when several of the priced and calibrated options have the same time to maturity.

Some numerical issues arise with the ChF for very long-term expiries. Following the a priori knowledge of parameter selection for the SWIFT method seems to be enough to avoid these problems, while the parameters of CP need to be adjusted manually. The proposed speed-up techniques are deemed necessary in order to make SWIFT a competitive calibration method, as has been seen in the numerical speed tests. In particular, it has been shown that the only situation where the proposed calibration is significantly slower than CP is when one calibrates the model with many different maturities with no more than one or two strikes per maturity. As the number of strikes per expiry increases, the relative speed of the SWIFT method increases, and it is about ten times faster than CP when calibrating 40 options with a single maturity. Both extreme situations are not representative of real option trading needs, and for a reasonably real situation of five

strikes per expiry, the SWIFT technique is slightly faster than CP. A SWIFT implementation without the previously discussed speed-up techniques has also been tested and deemed non-competitive, with calibration times that reached dozens of seconds. Further, the proposed calibration strategy passes the realistic calibration tests for challenging Heston model parameters setups presented.

In summary, the proposed SWIFT method is a robust and efficient machinery for real-time updating of option models used in human-supervised trading schemes. Neitherthe SWIFT nor the CP method are suitable for the most demanding algorithmic trading situations, like high-frequency trading. Future work may encompass the following topics:

- Most of the calibration tests with a single expiry have run into an optimal value different than the original one. It is to be seen if this is a property of the Heston model or if this was due instead to the specific parameter or strike/maturity values being used;
- It would be interesting to study the properties of the SWIFT implementation proposed for a chosen set of strikes in expression (32). We could interpolate the values at all strikes with spline methods that require derivatives, and not only derivative-free ones;
- Options with very long maturities may hamper the calibration process due to numerical overflows during the pricing step. The problem of long maturities has been tackled with Haar wavelets in [24]. It might be worth investigating whether we can do the same with Shannon wavelets;
- Deep OTM options with very short maturities are challenging to price. The problem seems to be the lack of accuracy of the approximation on the tails of the density function;
- Comparison with other calibration methods based on approximation formulae, like, for instance, the work by [34].

Author Contributions: Conceptualization, E.R. and L.O.-G.; methodology, E.R. and L.O.-G.; software, E.R.; investigation, E.R. and L.O.-G.; funding acquisition, L.O.-G. All authors have read and agreed to the published version of the manuscript.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A. Gradient Complimentary Formulas

Partial derivatives for gradient computation of the Heston model ChF from [5],

$$\frac{\partial d}{\partial \rho} = \frac{\xi \sigma i u}{d}, \tag{A1}$$

$$\frac{\partial A_2}{\partial \rho} = \frac{\sigma i u (2 + \xi \tau)}{2 d v_0} \left(\xi \cosh \frac{d\tau}{2} + d \sinh \frac{d\tau}{2} \right), \tag{A2}$$

$$\frac{\partial B}{\partial \rho} = \frac{e^{\kappa \tau / 2}}{v_0} \left(\frac{1}{A_2} \frac{\partial d}{\partial \rho} - \frac{d}{A_2^2} \frac{\partial A_2}{\partial \rho} \right), \tag{A3}$$

$$\frac{\partial A_1}{\partial \rho} = \frac{i u (u^2 - i u) \tau \xi \sigma}{2 d} \cosh \frac{d\tau}{2}, \tag{A4}$$

$$\frac{\partial A}{\partial \rho} = \frac{1}{A_2} \frac{\partial A_1}{\partial \rho} - \frac{A}{A_2} \frac{\partial A_2}{\partial \rho}, \tag{A5}$$

$$\frac{\partial A}{\partial \kappa} = -\frac{i}{\sigma u} \frac{\partial A}{\partial \rho}, \tag{A6}$$

$$\frac{\partial B}{\partial \kappa} = -\frac{i}{\sigma u} \frac{\partial B}{\partial \rho} + \frac{B\tau}{2}, \tag{A7}$$

$$\frac{\partial d}{\partial \sigma} = \left(\frac{d}{\sigma} - \frac{1}{\xi}\right)\frac{\partial d}{\partial \rho} + \frac{\sigma u^2}{d}, \tag{A8}$$

$$\frac{\partial A_1}{\partial \sigma} = \frac{(u^2 - iu)\tau}{2}\frac{\partial d}{\partial \sigma}\cosh\frac{d\tau}{2}, \tag{A9}$$

$$\frac{\partial A_2}{\partial \sigma} = \frac{\rho}{\sigma}\frac{\partial A_2}{\partial \rho} + \frac{2 + \tau\xi}{v_0\tau\xi iu}\frac{\partial A_1}{\partial \rho} + \frac{\sigma\tau A_1}{2v_0}, \tag{A10}$$

$$\frac{\partial A}{\partial \sigma} = \frac{1}{A_2}\frac{\partial A_1}{\partial \sigma} - \frac{A}{A_2}\frac{\partial A_2}{\partial \sigma}. \tag{A11}$$

Appendix B. Strike and Maturity Test Sets

Set 1 and set 2 are provided in Tables A1 and A2, respectively. The goal of set 3 is just to check the behavior of KSWIFT in the worst configuration possible for its speeding up techniques. This scenario is only be applied to KSWIFT, and consists of the same strikes and maturities as set 2. The code implementation of KSWIFT generated for this work receives as inputs a vector of expiry-defined-data (EDD). Each element of the vector of EDD contains a single expiry and a vector of strikes. Thus, set 2 will have all the strikes with the same expiry grouped in a single EDD, and set 3 will have an EDD consisting on a single strike. This will enforce full recomputation of the density and payoff coefficients for each strike.

Table A1. Set 1 of strikes and expiries. All the strikes have the same expiry $\tau = 0.119047619047619$.

Strike	Strike	Strike	Strike	Strike
0.9371	0.9956	1.0427	1.2287	1.3939
0.8603	0.9868	1.0463	1.2399	1.4102
0.8112	0.9728	1.0499	1.2485	1.4291
0.7760	0.9588	1.0530	1.2659	1.4456
0.7470	0.9464	1.0562	1.2646	1.4603
0.7216	0.9358	1.0593	1.2715	1.4736
0.6699	0.9175	1.0663	1.2859	1.5005
0.6137	0.9025	1.0766	1.3046	1.5328

Table A2. Set 2 of strikes and expiries.

Expiry	Strike	Strike	Strike	Strike	Strike
0.119047619047619	0.9371	0.9956	1.0427	1.2287	1.3939
0.238095238095238	0.8603	0.9868	1.0463	1.2399	1.4102
0.357142857142857	0.8112	0.9728	1.0499	1.2485	1.4291
0.476190476190476	0.7760	0.9588	1.0530	1.2659	1.4456
0.595238095238095	0.7470	0.9464	1.0562	1.2646	1.4603
0.714285714285714	0.7216	0.9358	1.0593	1.2715	1.4736
1.07142857142857	0.6699	0.9175	1.0663	1.2859	1.5005
1.42857142857143	0.6137	0.9025	1.0766	1.3046	1.5328

References

1. Ortiz-Gracia, L.; Oosterlee, C.W. A highly efficient Shannon wavelet inverse fourier technique for pricing European options. *SIAM J. Sci. Comput.* **2016**, *38*, B118–B143. [CrossRef]
2. Leitao, A.; Ortiz-Gracia, L.; Wagner, E.I. SWIFT valuation of discretely monitored arithmetic Asian options. *J. Comput. Sci.* **2018**, *28*, 120–139. [CrossRef]
3. Maree, S.C.; Ortiz-Gracia, L.; Oosterlee, C.W. Pricing early-exercise and discrete barrier options by Shannon wavelet expansions. *Numer. Math.* **2017**, *136*, 1035–1070. [CrossRef]
4. Moré, J.J. The Levenberg-Marquardt algorithm: Implementation and theory. In *Numerical Analysis*; Springer: Berlin/Heidelberg, Germany, **1978**.
5. Cui, Y.; del Baño Rollin, S.; Germano, G. Full and fast calibration of the Heston stochastic volatility model. *Eur. J. Oper. Res.* **2017**, *263*, 625–638. [CrossRef]
6. Heston, S.L. A Closed-Form Solution for Options with Stochastic Volatility with Applications to Bond and Currency Options. *Rev. Financ. Stud.* **1993**, *6*, 327–343. [CrossRef]
7. Clark, I. *Foreign Exchange Option Pricing: A Practitioner's Guide*. Wiley: Chichester, UK, 2011.
8. Gatheral, J. The volatility surface: A practitioner's guide. In *Finance*; Wiley: Chichester, UK, **2006**.
9. Gilli, M.; Schumann, E. Calibrating option pricing models with heuristics. In *Natural Computing in Computational Finance. Studies in Computational Intelligence*; Brabazon, A., O'Neill, M., Maringer, D., Eds.; Springer: Berlin/Heidelberg, 2011; Volume 380.
10. Janek, A.; Kluge, T.; Weron, R.; Wystup, U. FX smile in the Heston model. In *Statistical Tools for Finance and Insurance*; Cizek, P., Härdle, W., Weron, R., Eds.; Springer: Berlin/Heidelberg, 2011; pp. 133–162.
11. Bin, C. Calibration of the Heston Model with Application in Derivative Pricing and Hedging. Master's Thesis, TU Delft, Delft, The Netherlands, 2007.
12. Mikhailov, S.; Nögel, U. Heston's stochastic volatility model implementation, calibration and some extensions. *Wilmott Mag.* **2003**, *4*, 74–79.
13. Gerlich, F.; Giese, A.M.; Maruhn, J.H.; Sachs, E.W. Parameter identification in financial market models with a feasible point SQP algorithm. *Comput. Optim. Appl.* **2012**, *51*, 1137–1161. [CrossRef]
14. Lagarias, J.C.; Reeds, J.A.; Wright, M.H.; Wright, P.E. Convergence properties of the Nelder-Mead simplex method in low dimensions. *SIAM J. Optim.* **1998**, *9*, 112–147. [CrossRef]
15. Gilli, M.; Schumann, E. Heuristic optimisation in financial modelling. *Ann. Oper. Res.* **2012**, *193*, 129–158. [CrossRef]
16. Fernández, J.L.; Ferreiro, A.M.; García-Rodrxixguez, J.A.; Leitao, A.; López-Salas, J.G.; Vxaxzquez, C. Static and dynamic SABR stochastic volatility models: Calibration and option pricing using GPUs. *Math. Comput. Simul.* **2013**, *94*, 55–75. [CrossRef]
17. Kahl, C.; Jäckel, P. Not-so-complex logarithms in the Heston model. *Wilmott Mag.* **2006**, *19*, 94–103.
18. Albrecher, H.; Mayer, P.; Schoutens, W.; Tistaert, J. The little Heston trap. *Wilmott* **2007**, *1*, 83–92.
19. Schoutens, W.; Simons, E.; Tistaert, J. A perfect calibration! now what? *Wilmott* **2004**, *2*, 66–78. [CrossRef]
20. del Baño Rollin, S.; Ferreiro-Castilla, A.; Utzet, F. On the density of log-spot in the Heston volatility model. *Stoch. Process. Their Appl.* **2010**, *120*, 2037–2063. [CrossRef]
21. Mallat, S. *A Wavelet Tour of Signal Processing: the sparse way*; Academic Press: Cambridge, MA, USA, 2009.
22. Daubechies, I. *Ten Lectures on Wavelets*; CBMS-NSF Regional Conference Series in Applied Mathematics; Society for Industrial and Applied Mathematics: Philadelphia, PA, USA, 1992.
23. Cattani, C. Shannon wavelets theory. *Math. Probl. Eng.* **2008**, *2008*, 164808. [CrossRef] [PubMed]
24. Ortiz-Gracia, L.; Oosterlee, C.W. Robust pricing of European options with wavelets and the characteristic function. *SIAM J. Sci. Comput.* **2013**, *35*, B1055–B1084. [CrossRef]
25. Fang, F.; Oosterlee, C.W. A novel pricing method for european options based on fourier-cosine series expansions. *SIAM J. Sci. Comput.* **2008**, *31*, 826–848. [CrossRef]
26. Frigo, M.; Johnson, S.G. The design and implementation of FFTW3. *Proc. IEEE* **2005**, *93*, 216–231.
27. Floc'h, F.L. Notes on the SWIFT method based on Shannon Wavelets for Option Pricing. *arXiv* **2020**, arXiv:2005.13252.
28. Christoffersen, P.; Jacobs, K. The importance of the loss function in option valuation. *J. Financ. Econ.* **2004**, *72*, 291–318. [CrossRef]
29. Lourakis, M.I.A. Levmar: Levenberg–Marquardt Nonlinear Least Squares Algorithms in C/C++. 2004. Available online: https://github.com/jturney/levmar (accessed on 2 March 2020).
30. Planitz, M.; Anderson, E. LAPACK Users Guide. *Math. Gaz.* **1995**, *79*, 210. [CrossRef]
31. Glasserman, P.; Kim, K.K. Gamma expansion of the Heston stochastic volatility model. *Financ. Stochastics* **2011**, *15*, 267–296. [CrossRef]
32. Andersen, L. Simple and efficient simulation of the Heston stochastic volatility model. *J. Comput. Financ.* **2008**, *11*, 1–42. [CrossRef]
33. Willmore, F.T. Debugging with gdb. In *Introduction to Scientific and Technical Computing*; CRC Press: Boca Raton, FL, USA, 2016.
34. Alòs, E.; Santiago, R.D.; Vives, J. Calibration of stochastic volatility models via second-order approximation: The Heston case. *Int. J. Theor. Appl. Financ.* **2015**, *18*, 1550036. [CrossRef]