Machine Learning Applications for the Valuation of Options on Non-liquid Option Markets

Jiří Witzany, Milan Fičura

FFA Working Paper 1/2023

FACULTY OF FINANCE AND ACCOUNTING

Bibliographic information:

Witzany J., Fičura M. (2023). *Machine Learning Applications for the Valuation of Options on Non-liquid Option Markets*. FFA Working Paper 1/2023, FFA, Prague University of Economics and Business, Prague.

This paper can be downloaded at: wp.ffu.vse.cz

Contact e-mail: ffawp@vse.cz

Winston Churchill Sq. 1938/4, CZ-13067 Prague 3, Czech Republic, ffu.vse.cz

Machine Learning Applications for the Valuation of Options on Non-liquid Option Markets

Authors

Jiří Witzany[1], Milan Fičura[2]

Abstract

Recently, there has been considerable interest in machine learning (ML) applications for the valuation of options. The main motivation is the speed of calibration or, for example, the calculation of credit valuation adjustments (CVA). It is usually assumed that there is a relatively liquid market with plain vanilla option quotations that can be used to calibrate (using an ML model) the volatility surface, or to estimate the parameters of an advanced stochastic model. In the second stage the calibrated volatility surface (or the model parameters) are used to value given exotic options, again using a trained neural networks (NN) or another ML model. The NNs are typically trained "off-line" by sampling many model and market parameter combinations and calculating the options´ market values. In our research, we focus on the quite common situation of a non-liquid option market where we lack a sufficient number of plain vanilla option quotations to calibrate the volatility surface, but we still need to value an exotic option, or just a plain vanilla option subject to a more advanced stochastic model, as that is typical for energy and carbon derivative markets. We show that it is possible to use selected moments of the underlying historical price return series complemented with a volatility risk premium estimate to value such options using the ML approach.

AMS/JEL classification: C45, C63, G13

Keywords: derivatives valuation, options, calibration, neural networks

1. Introduction

Modern approaches to the valuation of financial, energy, and commodity options, or other derivatives, are based on various stochastic models describing the dynamics of the underlying asset prices. In more advanced models, there are additional directly unobservable stochastic variables such as volatility, jumps, or convenience yield in the case of commodities. The simpler models (such as geometric Brownian motion) allow us to obtain analytical formulas to value plain vanilla European style options (e.g., the Black-Scholes-Merton formula). However, the valuation becomes less and less tractable, with more complex models requiring partial numerical calculations or a general Monte Carlo simulation approach. The advantage of the complex stochastic models is their ability to capture better the return dynamics and to provide better evaluation of exotic derivatives. Their disadvantage lies in more

[1] Jiří Witzany, Faculty of Finance and Accounting, Prague University of Economics and Business, Czech Republic, (jiri.witzany@vse.cz, corresponding author).

[2] Milan Fičura, Faculty of Finance and Accounting, Prague University of Economics and Business, Czech Republic, (milan.ficura@vse.cz).

Note: This paper has been prepared under financial support of a grant GAČR 22-19617S "Modelling the structure and dynamics of energy, commodity and alternative asset prices", which the authors gratefully acknowledge.

difficult and slower computations, including the model calibration, risk quantification, or credit valuation adjustment (CVA) calculation.

Evaluation of a derivative contract, besides its parameters, requires knowledge of the model parameters that are typically obtained by the process of calibration. The usual calibration procedure is based on quotations of plain derivative contracts (typically European options) and on finding a vector of the model parameters that makes the model prices as close as possible to the market quotes. The model and its estimated parameters can then be used to evaluate an exotic derivative contract. The optimization requires repeated evaluation of a grid of options conditional on the model and its parameters, which means that if the single option evaluation is slow, then the calibration procedure itself is much slower. Since the calibration and valuation of derivatives needs to be done by traders "online" during a trading session, a valuation procedure that takes hours is not acceptable. A possible solution to the computational problem is to find a fast approximating function evaluating the plain vanilla derivatives with sufficient precision. It turns out that the standard feed-forward neural network (NN) trained on a synthetic model-generated dataset can serve this purpose quite well. The dataset can be generated "off-line" by sampling the parameters and evaluating the options. It can be as large as time and computational resources allow. Since the evaluation of a feed-forward NN based on matrix multiplication and elementary functions' application is relatively fast and efficient, the on-line calibration procedure becomes much more efficient. The NN pricing function approximation can also be used for CVA calculations requiring, in general, simulations of the underlying asset prices' dynamics and the evaluation of the conditional derivative values in all the scenarios and at different points in time. If the derivative evaluation involves a Monte Carlo simulation, then we face a problem of nested Monte Carlo simulations. Again, the fast NN approximation can make the CVA calculations manageable, similarly as in the case of derivative portfolios VaR or CVaR calculations.

The motivation of this research is the pricing of (exotic) options on markets requiring more complex stochastic models where, in addition, the plain vanilla option market is not sufficiently liquid to calibrate a model in the way described above. The energy commodity markets present a typical example. The behaviour of energy prices shows spikes and changing volatility, and the stochastic models need to be rich enough to capture not only these phenomena, but also the complex initial forward term structure. In addition, the energy markets are segmented due to limited or costly transportation. For example, the electricity power markets are segmented into national grids and the natural gas markets are segmented by the hubs from which the gas flows. While some segments might be liquid, even in terms of option derivatives trading, most segments have lower liquidity. In spite of that, options are occasionally traded on these markets, and so the stochastic models need to be somehow calibrated and the options evaluated.

The contribution of our paper is a proposal of an NN training and valuation methodology based on the utilization of the information from historical asset returns data in the form of moments and other selected characteristics. In order to value an option conditional on a stochastic volatility model, the volatility premium parameter is also needed. We propose a way to estimate it from available quotations on the market, or from a more liquid, but similar market. In our empirical study, we will demonstrate that the performance of the valuation approach is comparable to NN-based valuation using the implied volatilities.

We provide a review of literature related to NN applications for derivative pricing in the following section. Section 3 outlines the pricing and calibration NN based on option prices and our proposed approach. We also discuss the problem of irreducible error, variance, and bias decomposition. Section 4 summarizes the main properties of several stochastic models, in particular of the Heston model, and discusses the problem of the volatility premium (price of risk). Details of the synthetic dataset used for

training and out-of-sample testing are given in Section 5. The results of our empirical tests comparing the option based NN performance with our proposed moment-based pricing and calibration NNs are reported and discussed in Section 6. Finally, Section 7 summarizes the main results, conclusions, and further possible research directions.

2. Literature review

Hutchinson et al. (1994) is one of the first academic studies of NN applications for derivatives pricing. The study proposes a non-parametric approach to the valuation and hedging of options based on historical option prices that is compared to an NN trained to approximate the Black-Scholes (BS) formula on synthetically generated data. However, the NN model has the underlying asset price and the option parameters as the only inputs, while the interest rate and volatility are either assumed to be given or estimated in a simple way from the historical data.

Interestingly, there is a long gap in the literature on NN applications for derivatives pricing, with more papers coming at the end of the last decade. Culkin and Das (2017) show that a deep NN trained to approximate the BS model with volatility, interest rate, and income yield among the NN inputs (features) gives good precision (relative pricing error around 1% of the strike price and R^2 over 99.8%). Ferguson and Green (2018) train an NN to value basket options. The NN, which has volatilities and correlations among its features, is trained on a synthetically generated dataset where options are valued using a Monte Carlo simulation, possibly with a lower number of simulations. They argue that the NN learns to remove random Monte Carlo (MC) noise, and in the end performs better than an MC valuation with a relatively larger number of simulations.

Regarding the model calibration task, Hernandez (2017) proposes a direct approach where option market prices are features, and the vector of a stochastic model parameters is the target of an NN to be trained. This means that the market option prices, or the implied volatilities, must be available on a fixed grid. The empirical study trains an NN on a dataset generated in a way that is based on implied volatilities and yield curves used as features and the calibrated vectors of the (two-factor) Hull-White model parameters as targets.

Liu et al. (2019) propose a two-step model calibration approach: forward-pass when a pricing NN is trained and backward-pass when the pricing NN is used in a calibration optimization procedure. Büchel et al. (2021) advocate the two-step calibration approach, mainly from the perspective of validation requirements, and test it on market data. The performance of the NN approach is benchmarked against a real-life calibration framework that is used at a large financial institution. Horvath et al. (2021) show that a two-step approach can be used as a flexible tool to calibrate a wide range of 2nd generation stochastic models (e.g. rough volatility). Cao, Chen, Hull, & Poulos (2022) compare the two-step calibration approach with the volatility feature approach, where the volatility surface data and the option parameters are the features and an exotic option price is the output (target) of an NN.

In this study, we focus on the situation when plain-vanilla option market prices are not available and the parameters of a stochastic model need to be estimated from historical data and possibly from option prices quoted on related markets. Estimation of the Black-Scholes model parameters, in particular of the constant model volatility, is relatively simple using a sample estimate or a more advanced model such as GARCH etc. However, the estimation of parameters of models with stochastic volatility, jumps, and possibly other stochastic variables becomes much more challenging since only the asset returns are observable, while the stochastic volatility or jumps are latent. There is a vast literature on this subject mostly applying Bayesian methods such as MCMC or particle filters (see e.g. Shephard, 2004, Jacquier et al., 2007, or Fičura and Witzany, 2016). The problem with these methods,

similarly to the calibration from option prices, is their computational cost; in fact, it is typically much higher than in the case of the classical calibration. Many of the Bayesian methods are not appropriate when the pricing needs to be performed on-line, and so there is a strong motivation to research alternative, computationally more efficient machine learning methods.

3. Calibration of Models and Pricing of options with Neural Networks

3.1. Model Calibration

The derivatives pricing models typically describe the stochastic dynamics of returns and possibly of other stochastic variables such as the stochastic volatility. A particular model $\mathcal{M}$ is specified by a vector of parameters Θ and is assumed to be the data-generating process of the historical data as well as of their future probability distribution. Therefore, the model $\mathcal{M}$ can be used for the valuation of a given derivative with an analytical formula (such as the Black-Scholes formula), or for empirical valuation based on a forward-looking Monte Carlo simulation of the asset prices. In any case, for a given derivative with specification given by a vector of parameters OP (e.g. the strike price K, maturity T, and call/put option indicator, etc.) the value of the derivative is given by a function of Θ, OP, and the actual market prices MV (the underlying asset price, the risk-free interest rate, etc.), i.e.

$$c = f_{\mathcal{M}}(OP, MV, \Theta).$$

By calibrating the model $\mathcal{M}$ we understand the finding of a vector of parameters Θ consistent with the actual market prices c_j of derivatives with specifications $OP_j, j = 1, \dots, n$, with the other actual market prices MV, and with the series of historical returns of the underlying assets (let us denote the series as $data$). The common calibration approach appropriate, for example, for the pricing of exotic options is based on available plain vanilla (European style) option quotations c_j. Optimally, we would like to find Θ solving the equations $c_j = f_{\mathcal{M}}(OP_j, MV, \Theta)$ for all j. However, this is rarely possible, and we must implicitly accept that, firstly, the model is only an approximation of the market reality, and secondly, even if the model described the market reality perfectly, the market prices would contain noise with respect to the prices implied by the theoretical model. Therefore, the calibration procedure entails the minimization of a total loss function measuring the deviation of the model implied prices and the market quotations, i.e.

$$\widehat{\Theta} = \arg\min_{\Theta} \sum L(c_j, c_j^{\mathcal{M}}), \tag{1}$$

where $c_j^{\mathcal{M}} = f_{\mathcal{M}}(OP_j, MV, \Theta)$ and the loss function L might, for example, be the weighted squared difference between the market and option price $L(c_j, c_j^{\mathcal{M}}) = w_j(c_j - c_j^{\mathcal{M}})^2$, or the weighted squared difference between the Black-Scholes implied volatilities $L(c_j, c_j^{\mathcal{M}}) = w_j(\sigma_{BS}(c_j) - \sigma_{BS}(c_j^{\mathcal{M}}))^2$, etc. In the empirical study, further on, we will use uniformly weighted squared differences between the option prices, but another definition could easily be used as well.

An alternative calibration approach can be based on the history of market prices only. In this case, the goal is to estimate $\widehat{\Theta} \approx E[\Theta|data]$, or optimally, in the Bayesian set up, the posterior probability distribution of the parameters $p(\Theta|data)$.

It should be noted that the parameters calibrated based on the option prices are "risk-neutral", allowing us to price derivatives based on the risk-neutral principle, while the parameters estimated from historical data correspond to the "physical" probability measure. In order to transform the physical parameter estimates to risk-neutral, additional parameters such as the market or volatility

risk premiums need to be estimated. This is not an issue if we assume that the underlying process is the Black-Scholes model (geometric Brownian motion) where the volatility is assumed to be constant, but it becomes more problematic once we admit that the volatility is stochastic, and a volatility risk premium needs to be estimated. We are going to discuss this issue in more detail in Section 4. in the context of the Heston model, but let us firstly outline possible machine learning (ML) applications for derivative pricing and model calibration.

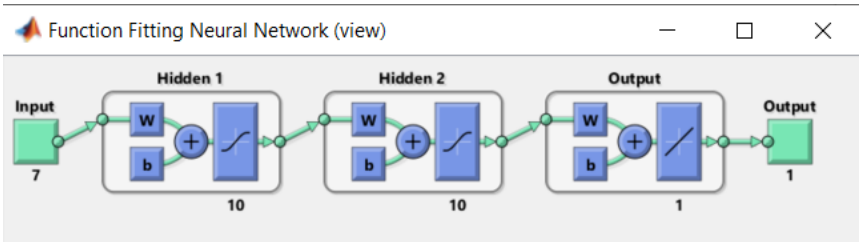
3.2. Neural Networks

Our goal will be to approximate a multivariable pricing function, or to estimate unknown model parameters given historical data using an ML model. There are many possibilities such as the neural networks, polynomial regression, regression support vector machines, gradient boosting regression, etc. (see e.g. Hastie et al., 2009). However, in line with the literature on ML applications for financial derivatives pricing, we will focus on neural networks that serve as flexible universal approximators: according to Hornik's (1991) well-known theorem, any continuous multivariable function can be approximated arbitrarily well by a single (or multi-layer for any given number of layers) hidden layer feed-forward NN.

We also do not focus too much on the various types of NN, but rather on their performance on the validation and testing datasets. We argue that the main issue, achieving higher precision of our approximating valuation functions, is in the "information content" of the vector of input variables (features) rather than the NN approximating performance itself. We will use the standard feed-forward architecture where the NN is specified by the number of neurons in the hidden layers (see e.g. Hastie et al., 2009). The default activation function will be the sigmoid function (i.e. $tanh$), and the training procedure will use the Levenberg-Marquardt optimization algorithm.

The NN and computations will be implemented in Matlab, which allows us to train, visualize and report the performance of the networks in an efficient and convenient way (see Figure 1 as an example of a two-hidden-layer NN with 10+10 hidden neurons and 1 output neuron).

Figure 1: A neural network with 2 hidden layers created in Matlab

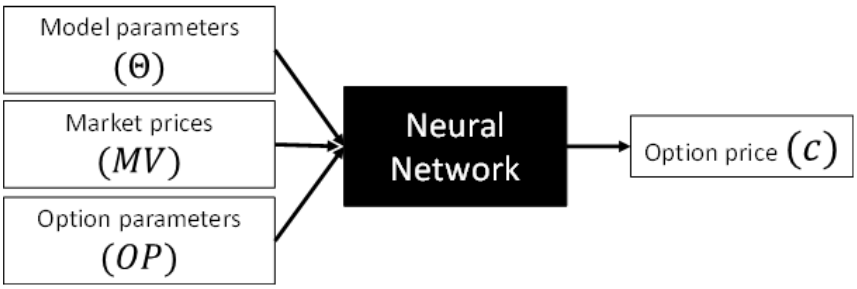


3.3. NN Pricing and Calibration Approach

Let us first outline the most straightforward NN pricing model. The goal is to train an NN approximating the function $c = f_{\mathcal{M}}(OP, MV, \Theta)$, i.e. with the features and the target as shown in Figure 2. The advantage of the model-based approach is that the synthetic training set can be generated by sampling the model parameters Θ_i, the market prices MV_i, and the option parameters OP_i. The derivative values c_i are then calculated using the valuation function, or its numerical (e.g. Monte Carlo) approximation. Numerical experiments (Ferguson and Green, 2018) show that the neural networks are able to remove the noise from the derivative calculation in the training set as long as the approximations are not biased. The off-line generated dataset can be as large as the computational resources and time allow. The parameters need to be sampled from appropriate distributions covering sufficiently well the range

of values for which the model is to be used. We will discuss our approach to dataset building in Section 5.

Figure 2: Option pricing Neural Network architecture



The trained NN defines a function approximating the theoretical valuation function

$$f_{NN}(OP, MV, \Theta) \approx f_{\mathcal{M}}(OP, MV, \Theta),$$

which can be used in the process of calibration, i.e.

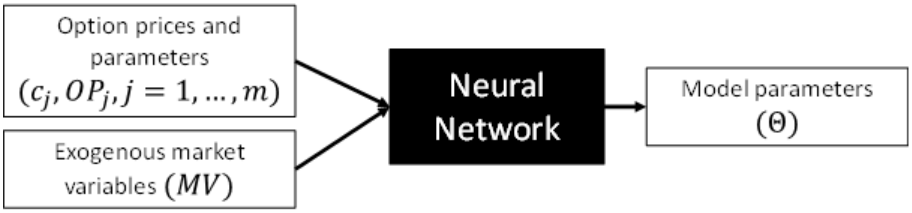
$$\widehat{\Theta} = \underset{\Theta}{\arg\min} \sum_{j} \left(f_{NN}(OP_j, MV, \Theta) - c_j\right)^2$$

where c_j is a set of actual option quotes with specifications $OP_j, j = 1, \dots, n$ and using the sum of squared errors as the loss function. The main advantage of the NN approximating function is its computational speed, which allows much faster and possibly online calibration of the model parameters. The speed of the optimization itself depends on the selected procedure. For example, Liu et al. (2019) interpret the calibration procedure as the backward-pass when the input layer of the NN used in the forward pass is trained using the data that enter the target layer. They propose using a population-based optimization algorithm (Differential Evolution) and show that it is able to estimate the calibrated parameters of the Heston model relatively fast.

3.4. Direct Calibration Approach and Volatility Feature Approach

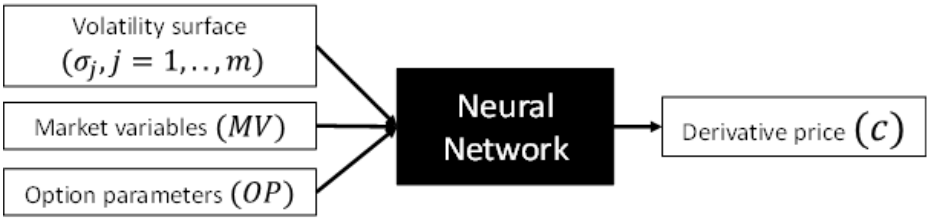
Another possibility (Hernandez, 2017) is the direct calibration approach with the NN architecture outlined in Figure 3. In this case, a set of market option prices is given on the input, and the model parameters define the target. It means that the synthetic training set needs to be generated in a slightly different way: for a given sampled set of model parameters Θ_i and market variables MV_i the option prices $Q_i = \langle c_{ij}, OP_{ij}, j = 1, \dots, m \rangle$ are calculated (or, alternatively, implied volatilities) typically on a grid of strike prices and maturities. The NN trained on the dataset defines a function $\widehat{\Theta} = f_{NN}(Q, MV)$ that estimates the model parameters in a fraction of seconds. Subsequently, the derivatives, for example exotic options, can be valued using the model valuation function or with a separate pricing NN having the parameters $\widehat{\Theta}$ on the input. The disadvantage of this approach is the lower precision of the estimates and a less transparent validation compared to the two-step model calibration approach (Büchel et al., 2021).

Figure 3: Direct calibration Neural Network architecture



Cao et al. (2022) suggest merging all the estimation steps into one, i.e. training a single NN that implicitly calibrates the model and estimates an exotic derivative value given the volatility surface (i.e. a set of implied volatilities on a grid of strike prices and maturities), market prices, and the derivative's parameters on the input (Figure 4). The NN is trained on a synthetic dataset generated conditional on a model (e.g. Heston) similarly to the direct calibration approach. In order to compare the volatility feature approach (VFA) with the two-step model calibration approach, Cao et al. (2022) pick another model (e.g. Bates) to generate a testing set of "true" volatility surfaces and prices of exotic options with varying parameters. The empirical results indicate that VFA performs slightly better compared to MCA.

Figure 4: Volatility feature option pricing Neural Network architecture



3.5. Realized Moments Feature Approach

Finally, let us formulate the calibration and pricing approach that represents the main contribution of this study. The problem encountered on many markets, in particular on the fragmented energy markets (fragmented by region and energy commodity types), is how to value options (possibly exotic) if there are no or just a few plain vanilla quotations. The natural suggestion is to use the only available information, i.e., besides the actual market prices and rates, the history of the underlying asset returns. The asset return history can be used to estimate stochastic model parameters (including stochastic volatility models) using various maximum likelihood or Bayesian methods that are too slow to be used for online calibration and pricing, as discussed above. Since we want to stick to the feed-forward NN that can efficiently process only a limited number of inputs, we propose to use selected moments and other characteristics $M = \langle m_j, j = 1, \dots, m \rangle$ as informative features that can be relatively efficiently calculated from the historical return series. The direct calibration or pricing NN architecture can then be designed similarly to the approaches based on the volatility surface as outlined in Figure 5 and Figure 6.

The training dataset for the direct calibration model (Figure 5) can be generated similarly as for the model shown in Figure 3. However, as explained in more detail in Section 4, the model parameters Θ_P of the physical data generating process differ from the risk-neutral parameters Θ_{RN}, and the difference is expressed by one or more risk premium parameters Λ. Therefore, to generate the training dataset we need to sample the physical model parameters Θ_j, simulate a series of historical returns and the corresponding vector of moments and other characteristics. Note that, for stochastic volatility models, the simulation involves also sampling of the stochastic volatility history, but the characteristics used as

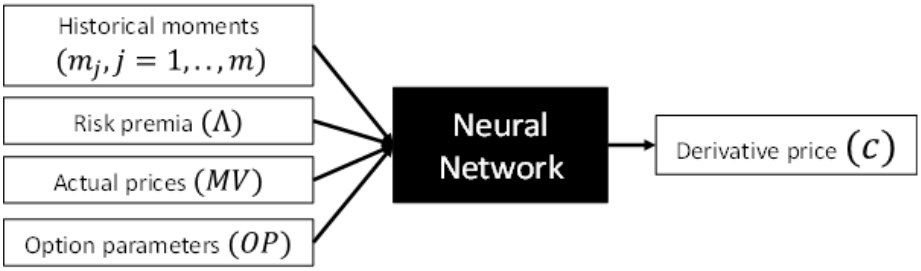
the NN input moments cannot use the stochastic volatility series, since it is not observable on the market.

Figure 5: Model calibration based on historical returns NN architecture

The vector of physical process parameters $\widehat{\Theta} = f_{NN}(M)$ estimated by a trained NN is not, unfortunately, sufficient for derivatives valuation, in particular if the model involves stochastic volatility. In this case, the volatility risk premium needs to be estimated (implied) from existing option quotation or from a related, but more liquid market (e.g., in the case of the energy markets).

Since the volatility risk premium (or other risk premia) cannot be inferred from the underlying asset return series, the pricing NN (Figure 6) input needs to include the vector of risk premia Λ. Hence the training dataset is in this case generated by sampling the physical process parameters Θ_i, risk premia Λ_i, then by simulating the physical process historical returns and their characteristics M_i, and finally calculating the derivative price $c_i = f_{\mathcal{M}}(OP_i, MV_i, \Theta_i, \Lambda_i)$ for some sampled option parameters OP_i and market variables MV_i (for $i = 1, \dots, N$). Strictly speaking, the risk premia are themselves model parameters, but since we need to distinguish the risk-neutral and the physical processes, the vector of risk premia Λ will be recorded separately.

Figure 6: Derivative pricing based on historical returns NN architecture



The flexibility of the NN allows us to experiment with other possible solutions to the risk premium estimation problem. We will also test an NN where one or only a few known quoted option prices complement the historical moments. In this case, the risk premiums can be inferred from the option prices, so the risk premiums do not need to be given on the input of the NN.

3.6. Estimation Error Decomposition

It is important to realize that there is a substantial difference between training an NN approximating a given function such as $c = f_{\mathcal{M}}(OP, MV, \Theta)$, and training an NN approximating a variable that is only statistically related to the inputs, as in the case of moment-based pricing.

As for any statistical learning algorithm trying to estimate a target Y given a vector of features X, the error can be decomposed into the ***irreducible error*** $\mathrm{var}[\mathrm{Y}|\mathrm{X}]$ given by the conditional probability distribution $p(Y|X)$, and the estimator $\hat{f}(X)$ ***variance*** and ***bias*** (see Hastie et al., 2009). Specifically,

$$E\left[\left(Y - \widehat{f_T}(X)\right)^2 |X\right] =$$
$$= \mathrm{var}[\mathrm{Y}|\mathrm{X}] + \mathrm{E}\left[\left(\mathrm{E}[\hat{f}_T(X)] - \hat{f}_T(X)\right)^2 |X\right] + \left(\mathrm{E}[\mathrm{Y}|\mathrm{X}] - \mathrm{E}[\hat{f}_T(X)|X]\right)^2 \quad (2)$$

where the estimator $\hat{f}_T(X)$ is also interpreted as a random variable depending on the training set T given a particular approximating model and its meta-parameters. The irreducible error depends on the relationship between X and Y, or, intuitively speaking, on the information content of X that can be used to estimate Y, and so it cannot be eliminated whatever the ML algorithm is or how complex it is. At the same time, the remaining two components of the error represent the key ML dilemma between, on the one hand, model complexity causing larger variance due to the effect of overfitting, and, on the other hand, robustness (simplicity) that typically causes a bias.

In the case of the NN pricing based on a model and set of parameters, the irreducible error will equal zero provided the valuation function $Y = f_{\mathcal{M}}(X)$ is implemented precisely. Nevertheless, even if the valuation function is implemented (when the training set is generated) with noise, i.e. $Y = f_{\mathcal{M}}(X) + \varepsilon$ where $E[\varepsilon|X] = 0$, then the irreducible error remains very small provided that the NN is trained with the quadratic loss function, i.e. targeting $E[Y|X] = f_{\mathcal{M}}(X)$, and the variance of noise $\mathrm{var}[\varepsilon|X]$ is relatively small. This fact explains the numerical observation of Ferguson and Green (2018) that a NN can achieve high precision even if it is trained on a training set with approximate derivative valuations.

However, in the volatility feature valuation approach, and in particular in the realized moment approach, the irreducible error becomes more significant – it depends on the "information content" of the vector of features. Firstly, focusing on the problem of model-calibration based on option market prices, we have to realize that the options are quoted with an error (or within a bid-ask spread interval) and not as theoretical option values. Therefore, for a given set of option market prices X there are many different parameter vectors Y that are theoretically consistent with the quotes, but of course with different conditional probabilities (likelihood). Moreover, the model we are calibrating only approximates the market reality. This is also reflected in the fact, that we only optimize the loss function (1) and do not seek an exact fit. Secondly, the "loose" statistical relationship between the input (features) X and the output (target) Y is even more pronounced in the case of moment-based calibration, or pricing when the synthetically simulated historical time series by definition contain a noise. In addition, when the model is implemented in practice, the historical prices will contain the market noise as well. Part of the "information" contained in the asset return series can also be lost by defining a limited set of moments and characteristics. Hence, their definition and selection will be a key problem in the design of both calibration and valuation NNs.

4. Selected Stochastic Models

Let us recall the standard **Black-Scholes (BS) model** characterized by the stochastic differential equation for the asset price S with constant drift μ and constant volatility σ, i.e.

$$dS = S\mu dt + S\sigma dW.$$

In order to value a European style option with the strike price K and maturity T, the drift parameter is replaced by the risk-free interest rate r, i.e. $c = f_{BS}(S_0, r, \sigma; K, T)$. Practically, the volatility σ is a key market parameter, in addition depending on K and T forming the volatility surface $\sigma = \sigma(K, T)$. However, in order to sample the historical returns under the physical probability measure, the drift μ is needed. It is typically expressed in the form $\mu = r + \lambda\sigma$, where λ is a given price of market risk. Or vice versa, if the parameters μ and σ are estimated from the historical returns then, the market price of risk can be calculated as $\lambda = \frac{\mu - r}{\sigma}$.

The BS model is sufficient to value European style options given the volatility surface $\sigma = \sigma(K, T)$ In fact, the (BS formula implied) volatility surface is just a conventional way to express the market prices of options with different strikes and maturities. To value more complex exotic options, the general Monte Carlo simulation approach can be used. The drift μ can be again replaced by the risk-free

interest rate, but it is not clear which constant volatility σ from the volatility surface should be used. The empirical phenomenon of the volatility surface is an expression of the fact that the market does not, in fact, see the asset log-returns normally distributed and the volatility constant. As a consequence, there are many alternative stochastic models trying to capture jumps in returns, stochastic volatility, or other properties and variables empirically observed on the market.

In our empirical analysis, we will focus on **the Heston (1993) model**, which includes an additional stochastic volatility (variance) V equation:

$$\begin{aligned} \frac{dS}{S} &= \mu dt + \sqrt{V} dW_1, \\ dV &= \kappa(\theta_V - V)dt + \sigma_V \sqrt{V} dW_2, \\ E[dW1, dW2] &= \rho dt, \end{aligned} \tag{3}$$

where ρ is the leverage correlation parameter (typically negative), κ is the speed of mean reversion of the stochastic variance to its long-term level θ_V, and σ_V the volatility of volatility. The advantage of the Heston model is that the European options can still be valued by a relatively fast formula $c = f_{HM}(S_0, r, \Theta; K, T)$ using numerical integration or the fast Fourier transformation. Here, $\Theta = \langle V_0, \kappa, \theta_V, \sigma_V, \rho, \lambda_V \rangle$ is the vector of parameters that includes two not explicitly shown parameters in the stochastic differential equations (3). Firstly, the instantaneous initial variance V_0 needs to be given as a model parameter, since the stochastic variance V is not directly observed on the market. Secondly, the price of volatility risk λ_V is another parameter that is needed to value a derivative, unlike the market price of risk, and this deserves some more detailed comments.

The derivation of the HM model formula can be obtained as a solution of the partial differential equation (PDE) based on the similar argument as in the BS model. Let $\Pi = c - \frac{\partial c}{\partial S} S$ be the value of a delta hedged portfolio where the long option is hedged by the underlying asset. The portfolio return does not depend on the (instantaneous) return of the underlying asset, but it does depend on the stochastic volatility (variance) return (change). The problem is that the variance is not a price of a traded asset and the portfolio cannot be hedged against the volatility risk by a position in another asset (it can be hedged only by another derivative sensitive to V). Therefore, the expected return of the portfolio $E[d\Pi] = r\Pi dt + \lambda_V V \frac{\partial c}{\partial V} dt$ includes the volatility premium $\lambda_V V \frac{\partial c}{\partial V} dt$, where $V \frac{\partial c}{\partial V}$ can be interpreted as the volatility exposure and λ_V as the price of volatility risk (Hull and White, 1987). The corresponding PDE then takes the form

$$\frac{\partial c}{\partial t} + \frac{1}{2}\frac{\partial^2 c}{\partial S^2} V S^2 + \frac{1}{2}\frac{\partial^2 c}{\partial V^2} V \sigma_V^2 + \frac{\partial^2 c}{\partial s \partial V} \sigma_V V S \rho = rc - \frac{\partial c}{\partial S} rS - \frac{\partial c}{\partial V} (\kappa(\theta_V - V) - \lambda_V V)$$

Note that the initial parameters of (3) are physical, but can be easily adjusted to risk-neutral, setting $\mu = r$, $\kappa^* = \kappa + \lambda_V$, and $\theta_V^* = \theta_V \frac{\kappa}{\kappa^*}$. Therefore, the parameters (κ^*, θ_V^*) correspond to the risk-neutral probability measure that sets equal to zero not only the price of the market risk $\lambda_M = 0$, but also the price of volatility risk, $\lambda_V = 0$. It turns out that the empirical price of volatility risk is usually negative, since the systematic risk of volatility is negative (volatility tends to go up when the market goes down). It means that the implied volatility systematically overestimates the physical volatility, and similarly, in terms of the Heston model, we usually have $\kappa^* < \kappa$ and $\theta_V^* > \theta_V$.

Therefore, when the model is calibrated from option price data (volatility surface), then the price of volatility risk needs to be set equal to a constant, e.g. $\lambda_V = 0$, and the same price of volatility risk needs to be used for the purpose of valuation. In particular, if $\lambda_V = 0$, then the fitted $\hat{\kappa}^*$ and $\hat{\theta}_V^*$ are risk-neutral. On the other hand, when the parameters of (3) are estimated from historical returns, then the estimates $\widehat{\Theta}_{\mathrm{P}} = \langle \hat{V}_0, \hat{\kappa}, \hat{\theta}_V, \hat{\sigma}_V, \hat{\rho} \rangle$ correspond to the physical parameters, and this information is not

sufficient to estimate the price of volatility risk λ_V. Given the physical model parameter estimates $\widehat{\Theta}_{\mathrm{P}}$ and an option (with specification OP) market price c_{mkt}, the price of volatility risk λ_V can, for example, be estimated (implied) by solving the equation

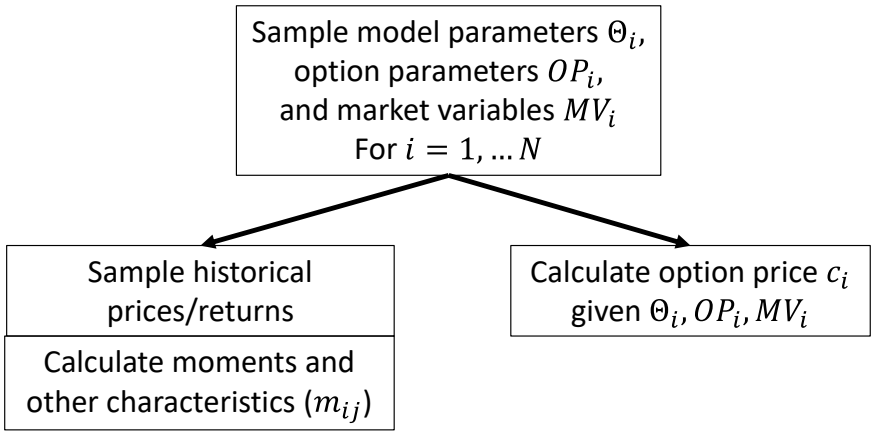
$$c_{mkt} = f_{HM}(S_0, r, \widehat{\Theta}_{\mathrm{P}}, \lambda_V; OP).$$

Let us conclude this section by mentioning a few other advanced stochastic models. Another relatively simple possible specification of the stochastic volatility model is the Hull and White (1987) model, where the variance V follows the geometrical Brownian motion, or the log-variance model suggested in Melino and Turnbull (1990). The models become less tractable and more computationally intensive when we add jumps in returns and in volatility (Bates, 1996, Eraker et al., 2003), or when the standard Brownian motion is replaced with the long memory fractional Brownian motion (see e.g. Horvath et al., 2021 or Cao et al.,2022). In order to model the term structure of commodity forward prices, additional parameters and stochastic variables need to be considered. For example, the Yan (2002) model, besides the stochastic volatility, considers stochastic jumps, interest rates and the convenience yield. In spite of its complexity, the futures prices can be obtained analytically and the option prices by using numerical integration or MC simulation.

5. Synthetic Training and Testing Dataset Construction

Let us describe in more detail how we are going to generate the synthetic datasets for training and testing the NN described in Section 3. The training, pricing, and calibration procedure is subject to a theoretical model that needs to be fixed at the beginning. In the empirical study, we will stick to the Heston model (HM), but the method can be applied to more complex models such as the rough volatility models. Generally, we will follow the scheme shown in Figure 7. However, there is a slight difference between the dataset for pricing and calibration based on market option prices (where the model parameters will be risk-neutral, setting $\lambda_M = \lambda_V = 0$) and the dataset for the realized moment-based pricing and calibration, where we need to sample the price of market risk λ_M and the price of volatility risk λ_V as well.

Figure 7: Synthetic training and testing dataset generation



Let us first note that due to the scalability of options the actual underlying asset price S_0 can be set at a fixed value, for example $S_0 = 100$. Since HM assumes a fixed interest rate r, we can set $r = 0$ without loss of generality. Hence the market variables MV_i do not have to be sampled at all.

Regarding the (European) option parameters $OP = \{I, K, T\}$, we will sample an indicator $I \in \{call, put\}$, the strike price $K \in [80,120]$, and the maturity $T = \frac{d}{252}, d \in \{1, \dots, 252\}$, in all cases with the uniform distribution.

The risk-neutral parameters $\Theta_{\mathrm{rn}} = \langle V_0, \kappa, \theta_V, \sigma_V, \rho \rangle$ of the Heston model are also sampled from the uniform distributions on intervals corresponding to the "normal" empirical values of the parameters as follows: $V_0 \in [0.05^2, 0.4^2], \kappa \in [0.3,1.3], \theta_{\mathrm{V}} \in [0.05^2, 0.4^2]$, $\sigma_V \in [0.01,0.2]$, and $\rho \in [-0.9,0.1]$. The parameters also need to satisfy the inequality $2\kappa\theta_V > \sigma_V^2$. For a sampled set of model parameters Θ_i and option parameters OP_i, the model option price is calculated as $c_i = f_{HM}(\Theta_i; OP_i)$ where we implicitly set $\lambda_V = 0$ and the market variables $S_0 = 100$, $r = 0$ are fixed as explained above. For the purpose of direct calibration (Figure 3) and volatility feature pricing (Figure 4) a set of option prices $c_{ij}, j = 1, \dots, m$ needs to be calculated on a grid of option parameters (strike price and maturity) conditional on the sampled parameters Θ_i.

For the purpose of the moment-based calibration (Figure 5) and pricing (Figure 6), NN models' physical parameters $\Theta = \{\theta_{\mathrm{V}}, \kappa, \sigma_V, \rho_{SV}, \lambda_M, \lambda_V, V_0\}$ need to be sampled. We will use the same uniform distributions for the option parameters and for the parameters $\kappa, \theta_V, \sigma_V, \rho$ as above, and in addition sample the market price of risk $\lambda_M \in [0.1,1.5]$, and the volatility premium parameter $\lambda_V \in [-2,0.1]$. In this case, we cannot independently sample the instantaneous variance V_0, since it has to be the last stochastic variance of the historical time series sampled according to (3). Here, we have to use a discretization based on a basic time step and a length of the time series (e.g. 252 daily returns). In order to initialize the series returns and stochastic volatilities, we need to sample $V_{ini} \in [0.05^2, 0.4^2]$ and the last simulated variance value then defines V_0. The historical return series will be used to calculate of set of moments, for example variance, skewness, and kurtosis in the 10, 21, 63, 126, and 252-day backward-looking time windows.

Our basic training and testing datasets (generated by the same underlying process) will have $N_{train} = 10\,000$ and $N_{test} = 2\,000$ observations. Hence, the total will be $N = N_{train} + N_{test} = 12\,000$, but we will also experiment with larger datasets to analyze the impact of the dataset size on the performance of the NN estimation model. Similarly, our basic dataset for the moment-based pricing calibration will use 252 daily historical returns, but we will also experiment with higher frequencies, longer periods, and additional historical return characteristics (for example based on the concept of realized volatility).

Note that the variables of the generated dataset can be used in the roles both of features and targets as indicated by Table 1. For example, the dataset with sampled model parameters, option prices, and historical moments can be used to train the calibration NN, where the vector of model parameters plays the role of the target, while in the case of the pricing NN the target is the option price (and the model parameters are not explicitly estimated by the NN).

Table 1: Training dataset structure

NN architecture	Features	Target
Standard pricing model	Model parameters, option parameters	Option price
Direct Calibration	Option prices (grid)	Model parameters
Volatility feature pricing	Option prices (grid), option parameters	Option price
Moments based calibration	Historical moments	Model parameters
Moments based pricing	Historical moments, option parameters	Option price

6. Synthetic Training and Testing Dataset Construction

Before focusing on the empirical results of the moments-based calibration and pricing using NN, which is the main contribution of this paper, we will show some examples of the results of the two-step and direct calibration based on the option prices.

6.1. Estimation Error Decomposition

Table 2 shows in-sample (IS) performance in terms of the RMSE of the pricing NN (Figure 2) trained on a dataset with 10 000 observations (synthetically generated as explained in the previous section). In addition, we show out-of-sample (OOS) performance on the (2 000) observations not used for training at all. The RMSE can be compared to the spot price $S_0 = 100$ or the strike price K (around 100) or the average option price in the sample, which is around 7, and we can see that the best reported out-of-sample RMSE is better than 0.2% of the average option price. It is interesting to see that the performance of the approximating NN dramatically improves when we add more neurons and hidden layers. Given the dataset with 10 000 observations, it seems that the NN with 3 hidden layers, each with 20 hidden neurons, provides almost optimal performance. The OOS performance is slightly improved when we increase the number of neurons in one layer to 30, but it should be noted that in this case there is a large difference between the reported IS and OOS performance, indicating increasing variance of the NN. The training time also increases substantially (since the number of weights in the NN with three hidden layers, each with n neurons, is proportional to n^2). As discussed in Section 3, there is almost no irreducible error (depending on the quality of the numerical integration used in the HM valuation function), and so the key problem of the NN approximation is the variance and bias which can be reduced by generating larger and larger training datasets and applying a more and more complex NN.

Table 2: Pricing NN performance with different number of neurons and hidden layers

NN # of neurons	Iterations	Training time	RMSE (IS)	RMSE (OOS)
10	65	0:00:01	0.2018	0.2031
20	181	0:00:05	0.0724	0.0840
10+10	1000	0:00:23	0.0572	0.0591
20+20	1000	0:02:35	0.0112	0.0139
20+20+20	964	0:06:19	0.0077	0.0156
30+30+30	1000	0:35:04	0.0026	0.0085

Figure 8 shows the performance (measured in terms of MSE) of the 20+20+20 network during the training procedure, which stops when an optimum on the validation subsample is achieved. The right panel illustrates the relatively narrow error histogram for the training, validation and test subsamples (of the dataset with 10 000 observations).

Figure 8: Performance of the NN (20+20+20) during the training procedure (left) and the error histogram of the trained NN

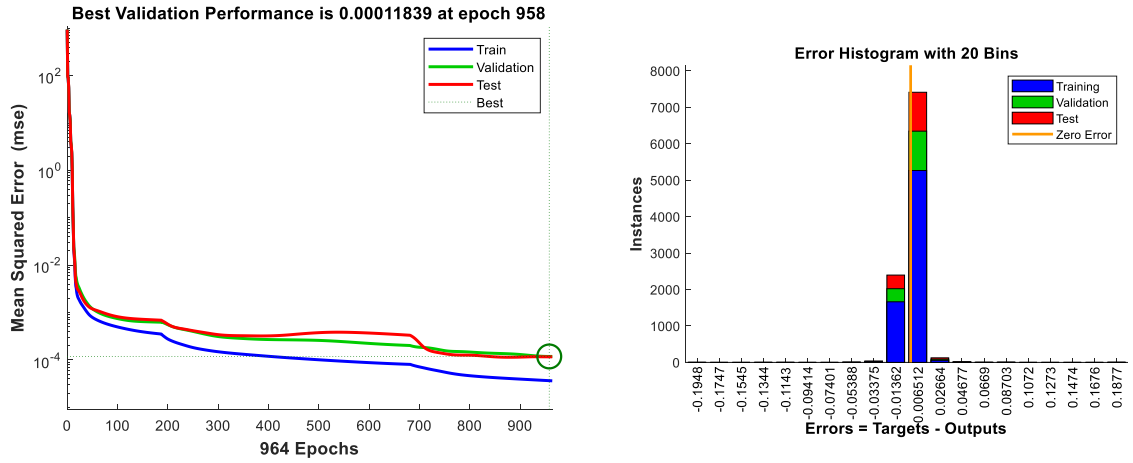
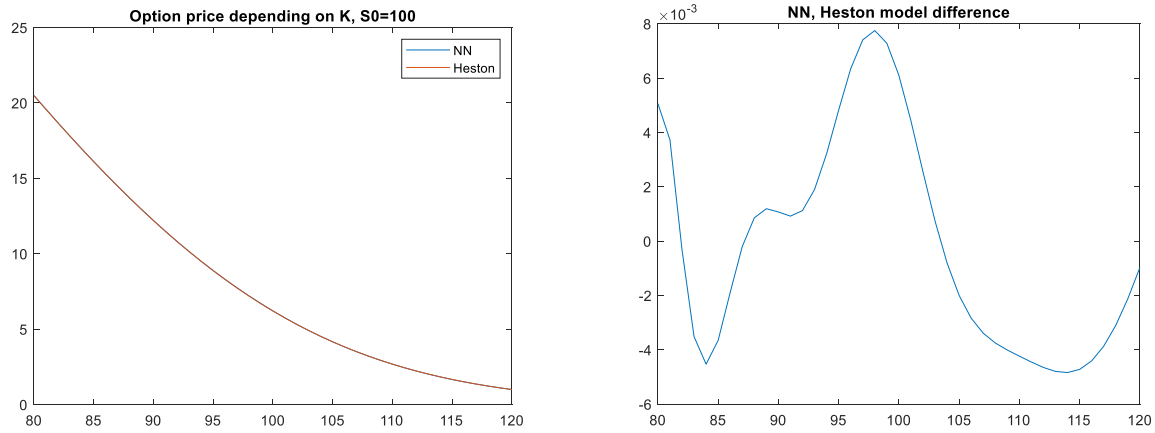


Figure 9 compares, for the sake of illustration, the Heston formula prices with the NN approximation for a call option depending on the strike prices and fixing the remaining (randomly selected) parameters, i.e. T and Θ. The left panel visually indicates that there is almost no difference between the HM function and the NN approximation, but the right panel shows that the difference between the two functions is in the range -0.006 to 0.008 corresponding to the reported RMSE.

Figure 9: A comparison between the Heston formula and NN approximated market values for call options with the strike price K=80,…,120



6.2. Classical Calibration using an approximating NN

To illustrate the calibration procedure in the two-step approach based on a trained approximating NN, we sample a vector HM parameters (from the distribution described in Section 5) and calculate (with the HM valuation function) a grid of option prices with strikes [90,95,100,105,110] and maturities [2W, 1M, 2M, 6M, 9M]. In order to make the grid of prices more realistic, we add random errors in the range ±0.5% (uniformly distributed) corresponding to market noise. We will assume that the grid represents a quoted volatility surface and perform the calibration procedure minimizing

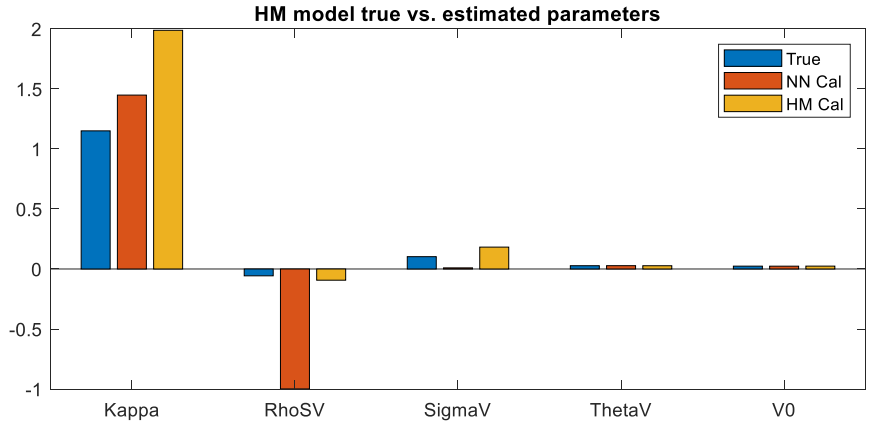
$$\arg\min_{\Theta} \sum \left(f\left(OP_{ij}, \Theta\right) - c_{ij} \right)^2,$$

where the valuation function is either the HM model function or the NN approximating function (trained as described above for the 20+20+20 network).

The NN based calibration converged relatively fast in approximately 1 min, showing RMSE = 0.44 with respect to the true parameters, while the HM function-based optimization took more than 17 min with only slightly better RMSE = 0.38. The calibration error (RMSE) with respect to the given grid of option prices was 0.13 for the NN calibration and 0.10 for the HM calibration. Figure 10 compares the true and the estimated parameters and indicates that the calibration error tends to be large, in particular for the leverage parameter ρ, the speed of reversion κ, and the volatility of volatility σ_V. The price of volatility risk λ_V is not shown, since it is implicitly set at zero when the calibration is based on the option prices. However, it should be noted that the optimization procedure fits the option prices including the noise for which the original "true" parameters do not necessarily provide the best fit. In any case, there are surprisingly large differences between the calibration results based on the two methods.

To conclude, the slightly lower precision of the NN based calibration seems to be compensated by its substantially better speed compared to the HM function-based calibration. Moreover, the precision of the NN can be further improved by generating a larger dataset and choosing a deeper NN as discussed above.

Figure 10: A comparison between the true and calibrated parameters using the HM function and the NN approximation



6.3. Direct Calibration

To illustrate the direct calibration performance, we have sampled 10 000 + 2 000 times (in + out) on a set of HM parameters (training target) and a grid of 25 corresponding call option prices for the maturities (10,20, 40, 60, 180) days and strike prices (90, 95, 100, 105, 110) with the spot price set at 100. Table 3 reports the calibration results in terms of the RMSE of the estimated parameters vs. the "true" parameters, and in terms of the pricing error when the estimated and "true" parameters are used to value the options on the grid. From both perspectives, the performance appears to be substantially better than the two-step calibration results shown above. However, the RMSE performance of the two-step approach and the reported NN calibration RMSE values cannot be directly compared, since in the two-step approach the option grid values included noise, while in Table 3 the RMSE is based on option grid values without any noise. When the comparison is done on a grid of "noisy" option values, then the performance is, in fact, similar.

Creation of the training dataset takes substantially more time compared to the dataset for the pricing NN, since the number of options to be evaluated is 25 times greater. But this calculation and the NN training, which also takes some time (see Table 3) can be done off-line, while the evaluation of the

trained calibration NN itself is very fast. Overall, our empirical results indicate that the direct calibration approach should be preferred to the two-step calibration approach.

Table 3: Direct calibration NN performance with different number of neurons and hidden layers

NN # of neurons	Iterations	Training time	Parameter RMSE (IS)	Parameter RMSE (OOS)	Pricing RMSE (OOS)
20	448	0:03:42	0.0520	0.0519	0.0811
20+20	764	0:32:49	0.0087	0.0152	0.0496
20+20+20	567	0:36:15	0.0060	0.0078	0.0364

6.4. Moments Based Pricing

Finally, we are going to present the empirical results of our proposed approach to the valuation of options using NN in a situation when market volatility (option) quotes are not available. Initially, we again use the same dataset with 10 000 + 2 000 (in + out) synthetically generated observations where the features are the selected historical moments, the (true) volatility risk premium, and option parameters. The only target is the option price. In the first empirical experiment, the moments are defined as described in Section 5, i.e. as variance, skewness, and kurtosis in the 10, 21, 63, 126 and 252-day backward-looking time windows based on 252 daily returns (i.e., the number of features is 19 including the price of volatility risk and option specifications). The results shown in Table 4 are promising, but certainly not as good as in the case of pricing using model parameters or parameters calibrated from the market option prices. It is interesting to note that the performance in terms of IS and OOS RMSE does not improve when we add more hidden layers and neurons. This could be explained by the irreducible error in the decomposition (2), due to the fact that the information content of the selected moments is limited and, in addition, contains the noise inherent to the synthetic data-generating process.

Table 4: Moment based pricing NN performance with different number of neurons and hidden layers

NN # of neurons	Iterations	Training time	RMSE (IS)	RMSE (OOS)
20	83	0:00:03	0.583	0.629
20+20	26	0:00:04	0.579	0.617
20+20+20	14	0:00:07	0.585	0.647

The relatively promising performance can also be illustrated by the simple (Mincer-Zarnowitz) univariate regression where the "true" prices are explained by the estimated prices with R^2 = 99.1% and RMSE = 0.6 (intercept -0.02, slope = 1.001) on the OOS dataset. See also Figure 11.

Figure 11: True (y-axis) versus estimated (x-axis) scatterplot

In the following subsections, we will consider two possible ways to improve the performance. Firstly, we will assume that there is at least one available option quote which can be used as a feature of the NN. The assumption is that the option price implicitly improves the model specification and, in addition, the price of volatility risk can be removed from the set of features, since it can be implied from the market option price. In the second direction, we will try to extend both the set of moments, in particular using the concept of realized volatility, and the size of the training dataset.

6.5. Mixed Inputs Based Pricing

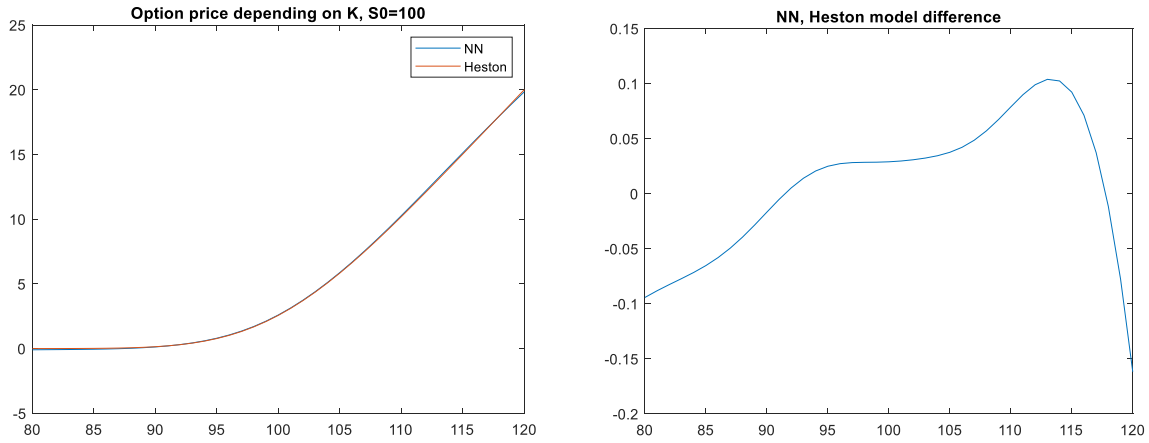
A straightforward solution to the missing volatility premium problem is to estimate the premium on a related market (e.g. the same or a similar product, or a different region in the case of energies). In this case, the model pricing function can be used, in particular if the estimation is off-line (assuming a constant volatility premium). Alternatively, if there are at least some option market quotes, we propose to use a mix of moments and available quotations. The second expected effect is a possible improvement in the approximating NN due to certain additional information contained in the option price. The results in Table 5 are based on the same dataset as for the moment-based pricing (i.e. 10 000 training observations with 15 moments based on 252 daily returns) where the volatility premium is replaced with a single ATM 1M call option price. It is interesting to note that the performance in terms of RMSE improved by almost one third and, in addition, the "true" value of the volatility price of risk could be eliminated. Again, the performance does not improve with increased NN complexity, indicating that the key issue is still the information content of the input features.

Table 5: Mixed input (moments and a market option price) based pricing NN performance

NN # of neurons	Iterations	Training time	RMSE (IS)	RMSE (OOS)
20	27	0:00:01	0.427	0.435
20+20	16	0:00:04	0.438	0.463
20+20+20	17	0:00:18	0.416	0.477

Figure 12 visually illustrates that the approximation quality of the option pricing function is similar to the case (pricing NN) when the model parameters are known. Regarding the Mincer-Zarnowitz regression, in this case R^2 improves to 99.6% and RMSE to 0.43 on the OOS dataset.

Figure 12: A comparison between the Heston formula and the mixed input NN approximated market values for call options with the strike price K=80,…,120



6.6. Improving the Performance of the Moment Based Pricing NN

As discussed in Section 6.3, it seems that the main obstacles to the improvement of the performance of the approximating NN is the limited information content of the set of features, which causes a relatively large irreducible error, and possibly also the size of the training data, which causes a relatively large variance of the approximating model (see Section 3). The former hypothesis could be directly verified by comparing the results of the moment-based estimation of the parameters with Bayesian (MCMC) estimations. This is a possible direction of future research. Indirectly, the hypothesis can be confirmed by extending the set of moments with additional historical return series characteristics and comparing the results.

We suggest improving the performance by using high-frequency returns, a longer historical time window and moments of daily realized variance time series approximating the series of latent stochastic variance. To calculate the realized variance (Andersen and Bollerslev, 1998), we use 10 min return data, which are in practice a reasonable choice due to the market microstructure noise. Specifically, the daily realized volatility will be calculated by the standard formula

$$RV_d = \sum_{i=i_{d,1}}^{i_{d,2}} r_i^2$$

where $r_i = \ln \frac{S_i}{S_{i-1}}$ is the high-frequency asset return and the summation covers a daily period. The time series $RV_d, d = 1, \dots, D$ can then be used to define moments characterizing the stochastic volatility equation (3). In addition, the last day realized variance is, in fact, a good proxy for the instantaneous latent variance V_0. Therefore, we will use the high-frequency returns to calculate variance, skewness, kurtosis, and, in addition, the fifth moment in the backward-looking periods of 1, 5, 20, 121, and 252*4 days extending the maximum historical period to 4 years, i.e. $D = 4 * 252$. The moment features of the set of 20 returns will, in addition, be enriched with the following characteristics:

- the EWMA daily variance estimate based high-frequency returns with $\lambda = 0.95^{1/m}$ where $m = 48$ is the number of high-frequency periods in one trading day,
- daily RV autocorrelations (with lags 1-4) serving as a proxy to estimate the speed of mean reversion κ,

- the first five moments of the RV daily changes,
- and the first five moments of the realized covariance between daily RV increments and the daily asset returns serving as a proxy for the leverage parameter ρ.

Therefore, the total number of historical characteristics serving as features will be 35. In addition, as stated above, there will be the "true" price of volatility risk and the option parameters as 4 additional input features. We will test the performance of the NN with the extended set of features and trained on 10 000 observations (IS, with OOS performance calculated on an additional 2 000 observations), or alternatively on 40 000 observations (IS, and an additional 4 000 OOS observations). The results in Table 6 and Table 7 show that the performance in terms of RMSE has improved substantially compared to Table 4 and bears comparison with the direct calibration performance (Table 3) based on a grid of option market prices. Note that the performance does not improve significantly when we add more hidden layers and neurons. In our opinion, the NN model becomes applicable for the valuation of options on non-liquid markets where bid-ask spreads, if any, are by definition largely of low liquidity. It should also be noted that the training procedure is very fast, taking at most minutes, and the valuation itself can be done in fractions of a second. On the other hand, the creation of the full training dataset, which can be done off-line, is, of course, computationally costlier, taking several hours on an i7 Core 3GHz computer (in Matlab).

Table 6: Moment based pricing NN performance, extended set of features, dataset 10 000 + 2 000

NN # of neurons	Iterations	Training time	RMSE (IS)	RMSE (OOS)
20	41	0:00:07	0.229	0.250
20+20	26	0:00:06	0.211	0.243
20+20+20	14	0:00:11	0.213	0.242

Table 7: Moment based pricing NN performance, extended set of features, dataset 40 000 + 4 000

NN # of neurons	Iterations	Training time	RMSE (IS)	RMSE (OOS)
20	235	0:03:27	0.216	0.219
20+20	48	0:01:34	0.206	0.210
20+20+20	21	0:01:15	0.207	0.221

All Table 7 models based on the larger dataset show excellent performance in the Mincer-Zarnowitz regression on the OOS sample. For example, in the case of the 20+20 model, the true prices are explained by the estimated prices with R^2= 99.9% and RMSE = 0.21 (intercept -0.003 not significantly different from 0 and slope = 0.9999 not significantly differing from 1, see also Figure 13).

Figure 13: True (y-axis) versus estimated (x-axis) scatterplot

For the sake of completeness, Table 8 reports the performance of the mixed input pricing NN, where we use the extended set of moments and an option price (ATM with one-month maturity). On the other hand, the "true" price of volatility risk is removed from the features vector as in Section 6.5. The training dataset is the larger one with 40 000 observations, and so the results could be compared to Table 7. It is interesting to note that in this case (compared to Section 6.5), the performance of the mixed input pricing NN is not better than the performance of the purely moment-based pricing NN. It appears that the loss of information due to the removal of the price of volatility risk from the input is not outweighed by adding the option market price to the features.

Table 8: Mixed input (moments and a market option price) based pricing NN performance with an extended set of moments and a large training dataset (40 000 + 4 000)

NN # of neurons	Iterations	Training time	RMSE (IS)	RMSE (OOS)
20	95	0:01:37	0.342	0.355
20+20	21	0:00:27	0.341	0.354
20+20+20	18	0:00:43	0.339	0.389

6.7. Moment Based Direct Calibration

Finally, we can also look at the moments-based direct calibration NN, where the option prices on the input are replaced by a set of moments. It is interesting to note that the pricing performance in terms of the RMSE of the moments-based calibration with the basic dataset reported in Table 9 is better than the performance of the pricing NN in Table 4. On the other hand, with the extended set of features and the larger training dataset, the performance improves (Table 10), but not as much as in the case of the pricing NN (Table 7). The pricing performance again does not improve with the NN complexity. In fact, it turns out to be the best for the one hidden layer NN with 20 neurons. Figure 14 shows that the parameters κ and ρ have the largest contribution to the overall parameter RMSE.

Table 9: Moment based calibration NN performance (basic set of features, 10 000 training observations)

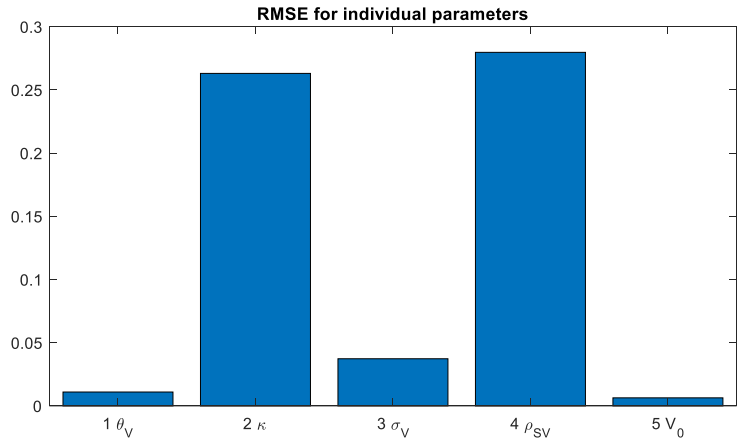
NN # of neurons	Iterations	Training time	Parameter RMSE (IS)	Parameter RMSE (OOS)	Pricing RMSE (OOS)
20	14	0:00:03	0.178	0.177	0.423
20+20	12	0:00:09	0.178	0.178	0.377

20+20+20	11	0:00:18	0.178	0.178	0.437

Table 10: Moment based calibration NN performance (extended set of features, 40 000 training observations)

NN # of neurons	Iterations	Training time	Parameter RMSE (IS)	Parameter RMSE (OOS)	Pricing RMSE (OOS)
20	17	0:01:00	0.174	0.173	0.273
20+20	21	0:03:17	0.173	0.173	0.296
20+20+20	21	0:05:20	0.173	0.172	0.356

Figure 14: OOS RMSE for individual HM parameters (moment based NN calibration, extended set of features, 40 000 training observation, one hidden layer with 20 neurons)



7. Conclusion

The applications of NN in option pricing are driven by several factors. Firstly, the valuation of (exotic) options based on advanced stochastic models necessarily involves the problem of calibration, which is often computationally too slow to be used for real-time trading. This problem arises when the valuation function is not analytical and involves a computationally intensive numerical procedure such as a Monte Carlo simulation. Secondly, the valuation of options and other derivatives on the OTC (over-the-counter) markets involves calculation of the credit valuation adjustment (CVA), or other analogous adjustments (XVA), which is again usually too slow if the valuation function is not very fast. This problem can be dealt with if the NN approximates the pricing function well, or by performing the calibration exercise directly.

Our contribution is, in addition, motivated by the situation of a non-liquid option market where the normal inputs of the calibration procedure, i.e. plain vanilla option prices (implied volatility surface) are not available, but we still need to price an option or calibrate a stochastic model. Our proposal is to train an NN on a synthetically generated dataset where we not only sample certain model parameters and evaluate options with randomly selected parameters, but, in addition, simulate backward- looking time series of returns, stochastic volatilities, and possibly other stochastic variables too. This approach allows us to train an NN that can estimate model parameters from the time series of historical returns. We have empirically tested this approach with the Heston Model and feed-forward neural networks. The results of the empirical experiments show that the moment-based pricing and calibration NNs can be applicable in practice with a performance lower than, but still comparable to the option-based NNs. We have shown that the performance (in terms of RMSE) can

be substantially improved when high-frequency historical data, allowing us to apply the concept of realized volatility, are available.

However, there are many possible directions of further research. It appears that the irreducible error (the limited “information content” of a historical return series with respect to unknown model parameters) is the main limitation of the moment-based pricing and calibration NN applications. This hypothesis can be verified, or more precisely analyzed, by comparing the NN errors with a Bayesian approach such as MCMC or particle filter estimations. Furthermore, other types of NN or ML algorithms could be tested. For example, the LSTM (long-short-term-memory) or convolutional (CNN) networks allow us to process the full return time series on the input instead of designing a limited set of moments or other characteristics that are needed for a feed-forward NN. Last but not least, the numerical testing should be implemented for more advanced models that capture not only the stochastic volatility, but also other stochastic variables, such as the convenience yield, or the term structure of forward prices that should be considered on the commodity markets.

8. References

[1] Andersen, T. G., Bollerslev, T. (1998). Answering the sceptics: yes standard volatility models do provide accurate forecasts. International Economic Review. 39 (4): 885–905.

[2] Bates, David S. (1996). Jumps and Stochastic volatility: Exchange Rate Processes Implicity in Deutsche Mark Options. The Review of Financial Studies, 9(1), 69–107.

[3] Büchel, P., Kratochwil, M., Nagl, M., & Rösch, D. (2021). Deep calibration of financial models: turning theory into practice. Review of Derivatives Research, 1-28.

[4] Cao, J., Chen, J., Hull, J., & Poulos, Z. (2022). Deep Learning for Exotic Option Valuation. The Journal of Financial Data Science, 4(1), 41-53.

[5] Culkin, R., & Das, S. R. (2017). Machine learning in finance: the case of deep learning for option pricing. Journal of Investment Management, 15(4), 92-100.

[6] Eraker, B., Johannes M., & Polson, M. (2003). The Impact of Jumps in Volatility and Returns, The Journal of Finance, 58(3), 1269-1300

[7] Ferguson, R., & Green, A. (2018). Deeply learning derivatives. arXiv preprint arXiv:1809.02233.

[8] Ficura, M., & Witzany, J. (2016). Estimating Stochastic Volatility and Jumps Using High-Frequency Data and Bayesian Methods. Czech Journal of Economics and Finance (Finance a uver), 66(4), 278-301.

[9] Hastie, T., Tibshirani, R., & Friedman, J. H. (2009). The elements of statistical learning: data mining, inference, and prediction (Vol. 2, pp. 1-758). New York: springer.

[10] Hernandez, A. (2017). Model calibration with neural networks. Risk.

[11] Heston, S. L. (1993). A Closed-Form Solution for Options with Stochastic Volatility with Applications to Bond and Currency Options. The Review of Financial Studies.,6 (2), 327–343.

[12] Hornik, K. (1991). Approximation capabilities of multilayer feedforward networks. Neural networks, 4(2), 251-257.

[13] Horvath, B., Muguruza, A., & Tomas, M. (2021). Deep learning volatility: a deep neural network perspective on pricing and calibration in (rough) volatility models. Quantitative Finance, 21(1), 11

[14] Hull, J., & White, A. (1987). The pricing of options on assets with stochastic volatilities. The Journal of Finance, 42(2), 281-300.

[15] Hutchinson, J. M., Lo, A. W., & Poggio, T. (1994). A nonparametric approach to pricing and hedging derivative securities via learning networks. The Journal of Finance, 49(3), 851-889.

[16] Jacquier, E., Johannes, M., & Polson, N. (2007). MCMC maximum likelihood for latent state models. Journal of Econometrics, 137(2), 615-640.

[17] Liu, S., Borovykh, A., Grzelak, L. A., & Oosterlee, C. W. (2019). A neural network-based framework for financial model calibration. Journal of Mathematics in Industry, 9(1), 1-28.

[18] Melino, A. and S. M. Turnbull (1990). Pricing foreign currency options with stochastic volatility, Journal of Econometrics, 45, 239-265.

[19] Shephard, N. (2004). Stochastic Volatility: Selected Readings. Oxford: Oxford University Press, p. 534.

[20] Yan, X. S., (2002). Valuation of commodity derivatives in a new multi-factor model, Review of Derivatives Research, Vol.5, No.3, pp.251–271.

FFA Working Paper Series

2019

1. Milan Fičura: Forecasting Foreign Exchange Rate Movements with k-Nearest-Neighbor, Ridge Regression and Feed-Forward Neural Networks.

2020

1. Jiří Witzany: Stressing of Migration Matrices for IFRS 9 and ICAAP Calculations.
2. Matěj Maivald, Petr Teplý: The impact of low interest rates on banks' non-performing loans.
3. Karel Janda, Binyi Zhang: The impact of renewable energy and technology innovation on Chinese carbon dioxide emissions.
4. Jiří Witzany, Anastasiia Kozina: Recovery process optimization using survival regression.

2021

1. Karel Janda, Oleg Kravtsov: Banking Supervision and Risk-Adjusted Performance in the Host Country Environment.
2. Jakub Drahokoupil: Variance Gamma process in the option pricing model.
3. Jiří Witzany, Oľga Pastiranová: IFRS 9 and its behaviour in the cycle: The evidence on EU Countries.

2022

1. Karel Janda, Ladislav Kristoufek, Binyi Zhang: Return and volatility spillovers between Chinese and U.S. Clean Energy Related Stocks: Evidence from VAR-MGARCH estimations.
2. Lukáš Fiala: Modelling of mortgage debt's determinants: The case of the Czech Republic.
3. Oľga Jakubíková: Profit smoothing of European banks under IFRS 9.
4. Milan Fičura, Jiří Witzany: Determinants of NMD Pass-Through Rates in Eurozone Countries.
5. Sophio Togonidze, Evžen Kočenda: Macroeconomic Responses of Emerging Market Economies to Oil Price Shocks: An Analysis by Region and Resource Profile.

6. Jakub Drahokoupil: Application of the XGBoost algorithm and Bayesian optimization for the Bitcoin price prediction during the COVID-19 period.
7. Karel Janda, Binyi Zhang: Econometric estimation of green bond premiums in the Chinese market.
8. Shahriyar Aliyev, Evžen Kočenda: ECB monetary policy and commodity prices.
9. Sophio Togonidze, Evžen Kočenda: Macroeconomic implications of oil-price shocks to emerging economies: a Markov regime-switching approach.

2023

1. Jiří Witzany, Milan Fičura: Machine Learning Applications for the Valuation of Options on Non-liquid Option Markets.

All papers can be downloaded at: wp.ffu.vse.cz

Contact e-mail: ffawp@vse.cz

Faculty of Finance and Accounting, Prague University of Economics and Business, 2023

Winston Churchill Sq. 1938/4, CZ-13067 Prague 3, Czech Republic, ffu.vse.cz