A Comparison of Neural Networks and Bayesian MCMC for the Heston Model Estimation

(Forget Statistics – Machine Learning is Sufficient!)

Jiří Witzany

Milan Fičura

FFA Working Paper 7/2023

FACULTY OF FINANCE AND ACCOUNTING

Bibliographic information:

Witzany J., Fičura M. (2023). *A Comparison of Neural Networks and Bayesian MCMC for the Heston Model Estimation (Forget Statistics – Machine Learning is Sufficient!)*. FFA Working Paper 7/2023, 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

A Comparison of Neural Networks and Bayesian MCMC for the Heston Model Estimation

(Forget Statistics – Machine Learning is Sufficient!)

Jiří Witzany[1], Milan Fičura[2]

Abstract

The main goal of this paper is to compare the classical MCMC estimation method with a universal Neural Network (NN) approach to estimate unknown parameters of the Heston stochastic volatility model given a series of observable asset returns. The main idea of the NN approach is to generate a large training synthetic dataset with sampled parameter vectors and the return series conditional on the Heston model. The NN can then be trained reverting the input and output, i.e. setting the return series, or rather a set of derived generalized moments as the input features and the parameters as the target. Once the NN has been trained, the estimation of parameters given observed return series becomes very efficient compared to the MCMC algorithm. Our empirical study implements the MCMC estimation algorithm and demonstrates that the trained NN provides more precise and substantially faster estimations of the Heston model parameters. We discuss some other advantages and disadvantages of the two methods and hypothesize that the universal NN approach can in general give better results compared to the classical statistical estimation methods for a wide class of models.

AMS/JEL classification: C45, C63, G13

Keywords: Heston model, parameter estimation, neural networks, MCMC

1. Introduction

Recently, there has been a considerable interest in machine learning methods applications to financial models calibration, i.e. unknown parameter estimation based on observable market data. For example, in case of advanced stochastic volatility models used for valuation of options and other derivatives, it is important to perform a relatively fast calibration based on available plain vanilla option prices or other market information. It turns out that the calibration performed using classical statistical or optimization methods is often substantially slower compared to a calibration based on neural networks (see e.g. Horvath et al., 2021, or Cao et al., 2022). Another approach to the calibration, in case of unavailable option prices, is to estimate

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

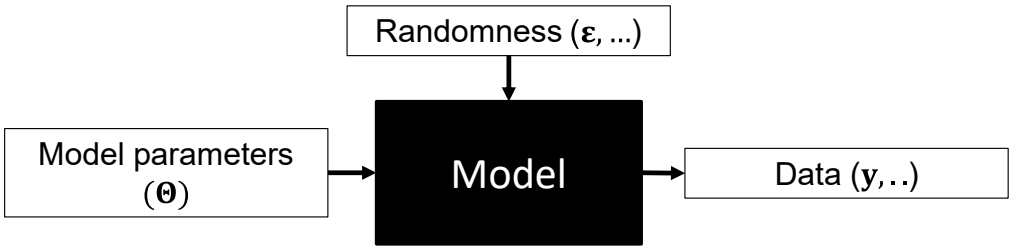
[2] Milan Fičura, Faculty of Finance and Accounting, Prague University of Economics and Business, Czech Republic, (milan.ficura@vse.cz).

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

the model parameters based on available historical returns of the underlying assets. Hutchinson et al. (1994) in their seminal paper tested this approach on the relatively simple Black-Scholes model. However, in case of stochastic volatility or jump models, this is known to be a difficult estimation problem since the stochastic volatility (and/or jump occurrence) represent latent state variables that need to be estimated, or integrated out, at the same time with the parameters' estimation (Shephard, 2004). Witzany and Ficura (2023) proposed a neural network (generalized) moment-based approach to the Heston model calibration and option valuation based on historical underlying asset prices. The trained neural network (NN) models showed promising results but with estimation errors relatively large compared to the classical calibration using quoted option prices (unavailable in this case). The estimation error can be generally decomposed into the irreducible error due to limited information content of the input in the form of an observed returns dataset and to the error due to the estimation method itself. The main motivation of this paper is to implement the Bayesian MCMC method for the Heston model in order to show that the former type of error represents the key contribution to the overall estimation error, and that the neural network estimation approach, in fact, performs quite well. This can be done looking on the posterior distribution of the parameters conditional on the input data, or in a simpler way, comparing the estimation error of MCMC point estimates and the NN estimates.

The motivation of this paper can be actually formulated in more general terms. A key problem of statistics is to analyze collected data by describing an appropriate probabilistic data-generating model (Figure 1) with parameters that are somehow estimated from the observed data. The classical estimation approaches include the method of moments (MM), the generalized method of moments (GMM), the maximum likelihood estimation (MLE), or Bayesian methods such as the Markov Chain Monte Carlo (MCMC) with Gibbs, Metropolis-Hastings, or Particle Filter sampling (see e.g. Lynch, 2007, Johannes and Polson, 2009, or Speekenbrink, 2016). The Bayesian MCMC approach turns out to be superior in case of latent state-space models like the stochastic volatility models. However, the implementation of the Bayesian estimation methods requires a tedious analysis of the marginal densities, proposal densities, and prior distributions. In addition, the estimation procedure is usually quite slow and computationally-intensive with certain level of uncertainty regarding the speed of convergence of the sampled chain to the theoretical distribution.

Figure 1 - A data generating model scheme



The machine learning (NN) based model estimation offers a relatively straightforward approach where the NN learns the relationship between the observed data and the set of parameters Θ on a large synthetically generated dataset. A possible architecture of a feed-forward NN with a fixed number of features is outlined in Figure 2. In this case, the general dataset can be very large and it needs to be transformed to a limited set of extracted characteristics $\mathbf{m} = \langle m_i \rangle$ that can be called "generalized moments" in an analogy with GMM. The training dataset is obtained by sampling the model parameters Θ_j from an appropriate distribution, sampling the dataset of a given size and conditional on the model (Figure 1), and finally calculating the generalized

moments $\mathbf{m}_j$. The synthetic dataset $\{(\mathbf{m}_j, \Theta_j); j = 1, \ldots, N\}$ is then used for training of the NN as outlined in Figure 2. The advantage of this approach is that the training dataset can be as large as we wish and computational resources allow. Moreover, the NN training procedure, which might be computationally more intensive, can be implemented offline, while the trained NN itself, used for model estimation, would be usually very fast.

Figure 2 - A neural network estimating model parameters from historical return moments

We will show that in case of the Heston model the NN parameter estimation approach gives quite good results compared to the MCMC approach. Note that NN estimation is very flexible and straightforward since it does not require a decomposition of the data-generating model into marginal densities etc. as in case of MCMC. However, one can object that in case of the Figure 2 NN architecture we need to manually find and define an appropriate set of generalized moments requiring a deeper analysis of the model. This part of the process can be eliminated employing other architectures such as the Convolutional or Long-Short-Term-Memory (LSTM) NN where the full dataset can be used on the input of the network. We hypothesize that this universal approach is quite efficient and applicable to a wide class of models, in particular to the more complex latent state variable models that are difficult to estimate using classical methods.

The paper is organized as follows: after the introduction, Section 2 describes the MCMC method applied to the Heston model, Section 3 specifies the NN training dataset construction, Section 4 reports and discusses the empirical results, and Section 5 concludes.

2. MCMC Estimation of the Heston Model

2.1. The Heston Model

The Heston (1993) model describing a price process S (or log-price $s = \ln S$ process) in continuous time with stochastic volatility (variance) V can be described by the following stochastic differential equations

$$ds = \mu dt + \sqrt{V} dW_1,$$
$$dV = \kappa(V_{LT} - V)dt + \sigma_V \sqrt{V} dW_2,$$
$$corr(dW_1, dW_2) = \rho.$$

The stochastic variance will stay (almost surely) positive provided the Feller (1991) condition $2\kappa V_{LT} > \sigma_V^2$ is satisfied.

Since the prices and returns are in reality observed over T discrete time intervals of length Δt (e.g. one trading day, i.e. $\Delta t = 1/250$) we discretize the model as follows: set $y_i = s(i\Delta t) - s\big((i-1)\Delta t\big)$, $i = 1, \ldots, T$ and

$$v_i = \alpha + \beta v_{i-1} + \gamma \sqrt{v_{i-1}} \varepsilon_i^V, \qquad (1)$$
$$y_i = m + \sqrt{v_i} \varepsilon_i,$$

$$\varepsilon_i \sim N(0,1), \varepsilon_i^V \sim N(0,1),\ corr(\varepsilon_{i-1}, \varepsilon_i^V) = \rho \text{ for } i \geq 2, \text{ i.e.}$$

$$\varepsilon_i^V = \rho\varepsilon_{i-1} + \sqrt{1-\rho^2}u,\ u \sim N(0,1), iid.$$

The relationship between the continuous and discrete time model parameters and the latent stochastic volatility is as follows: $m \approx \mu\Delta t$, $v \approx V\Delta t$, $\beta \approx 1 - \kappa\Delta t$, $\alpha \approx \kappa V_{LT}\Delta t^2$, and $\gamma \approx \sigma_V \Delta t$. Note that γ must satisfy the Feller condition: $2\alpha > \gamma^2$.

Therefore, given observed returns $\boldsymbol{y} = \langle y_1, \ldots, y_T \rangle$, the main goal is to estimate the vector of parameters $\Theta = \langle m, \alpha, \beta, \gamma, \rho \rangle$. An auxiliary goal might be to estimate the latent variances $\boldsymbol{v} = \langle v_1, \ldots, v_T \rangle$, or at least the last one, i.e. v_T, that allows us to analyze and simulate the future development of the return and volatility processes.

2.2. The MCMC Estimation Algorithm

In the Bayesian approach, the unknown parameters and state variables are estimated based on the equation

$$p(\Theta, \mathbf{v}|\mathbf{y}) \propto L(\mathbf{y}|\mathbf{v}, \Theta) \times prior(\mathbf{v}, \Theta). \tag{2}$$

The likelihood function is given by the discrete-time Heston model (1) is expressed as a product of bivariate normal densities of the pairs $\langle v_i, y_{i-1} \rangle$ conditional on v_{i-1} for $i = 2, .., T$, and of the univariate normal density of y_T conditional on v_T (as the value of v_{T+1} is unknown), i.e.

$$L(\mathbf{y}|\mathbf{v}, \Theta) = \varphi(y_T; m, v_T) \prod_{i=2}^{T} \varphi_2 \left(\langle v_i, y_{i-1} \rangle; \langle \alpha + \beta v_{i-1}, m \rangle, \begin{pmatrix} \gamma^2 v_{i-1} & \rho\gamma v_{i-1} \\ \rho\gamma v_{i-1} & v_{i-1} \end{pmatrix} \right). \tag{3}$$

The multivariate prior distribution in (2) is specified below based either on noninformative or appropriate wide distributions reflecting some basic limitations that we need to put on the parameters. The main principle of the MCMC algorithm is to iteratively sample the parameters and latent variables one-by-one (or in blocks) conditional on all the other already sampled parameters and variables from the conditional distributions that are typically simpler than the joint posterior distribution given by (2). According to Clifford-Hammersley theorem (see e.g. Johannes and Polson, 2009) the sequence (chain) of sampled parameters and variables converges to the distribution given by (2). It should be noted that there are different approaches to the HM Bayesian estimation that can be found in literature (see e.g. Frühwirth-Schnatter and Sögner, 2003), but our goal is to propose a rather straightforward MCMC estimation algorithm that can be empirically compared with the NN approach.

The MCMC sampling is typically performed by the Gibbs sampler if the marginal distribution has a known parametric form, or otherwise, typically, by the Metropolis-Hastings accept-reject sampler based on a proposal distribution. We will use the latter possibility since the HM marginal distributions are difficult or impossible to find in parametric form due to relative complexity of the bivariate densities in the likelihood function (3). The following summarize the individual MCMC sampling steps of the unknown parameters $\Theta = \langle m, \alpha, \beta, \gamma, \rho \rangle$ and volatilities $\boldsymbol{v} = \langle v_1, \ldots, v_T \rangle$ given observed returns $\mathbf{y} = \langle y_1, \ldots, y_T \rangle$:

(i) Sampling of the mean return m, given $\mathbf{y} = \langle y_1, \ldots, y_T \rangle$ and $\mathbf{v} = \langle v_1, \ldots, v_T \rangle$,

based on $p(y_i|m, v_i) = \varphi(y_i; m, v_i)$, and with the non-informative prior sample the proposal:

$$q(m|\mathbf{y},\mathbf{v}) = \varphi\left(m;\frac{B}{A},\frac{1}{A}\right), \text{where}$$

$$A = \sum_{i=1}^{T}\frac{1}{v_i}, B = \sum_{i=1}^{T}\frac{y_i}{v_i}.$$

The sampled value m cannot be used as in the Gibbs sampler since the proposal density does not consider the leverage correlation ρ. Therefore, we need to accept/reject the proposal m_1 against the previous value m_0 with the probability $\min(R, 1)$, where

$$R = \frac{p(m_1|\dots)q(m_0|\dots)}{p(m_0|\dots)q(m_1|\dots)},$$

$$p(m|\dots) = L(\mathbf{y}|\mathbf{v},\Theta), \Theta = \langle m, \dots\rangle,$$

$$q(m|\dots) = \varphi\left(m;\frac{B}{A},\frac{1}{A}\right).$$

(ii) Sampling of the SV equation parameters α, β, γ, given $\mathbf{v} = \langle v_1, \dots, v_T\rangle$,

based on $p(v_i|\alpha,\beta,\gamma,v_{i-1}) = \varphi(v_i;\alpha+\beta v_{i-1},\gamma^2 v_{i-1})$, and with the non-informative prior sample from the proposal:

$$q(\alpha|\dots) = \varphi\left(\alpha;\frac{B}{A},\frac{1}{A}\right), \text{where}$$

$$A = \sum_{i=2}^{T}\frac{1}{\gamma^2 v_{i-1}}, B = \sum_{i=2}^{T}\frac{v_i-\beta v_{i-1}}{\gamma^2 v_{i-1}}.$$

And, again, accept/reject based on $p(\alpha|\dots) = L(\mathbf{y}|\mathbf{v},\Theta)$ and $q(\alpha|\dots) = \varphi\left(\alpha;\frac{B}{A},\frac{1}{A}\right)$.

Similarly, accept/reject the proposal for β:

$$q(\beta|\alpha,\gamma,\mathbf{v}) \propto \varphi\left(\beta;\frac{B}{A},\frac{1}{A}\right)\times prior(\beta), \text{where}$$

$$A = \sum_{i=2}^{T}\frac{v_{i-1}}{\gamma^2}, B = \sum_{i=2}^{T}\frac{v_i-\alpha}{\gamma^2 v_{i-1}}.$$

In this case we will use an informative $prior(\beta) = \varphi(\beta; 0.985, 0.02^2)$ since in case of daily data the parameter $\beta \approx 1-\kappa\Delta t$ is expected to be around 0.99.

And finally, accept/reject the proposal for $G = \gamma^2$, with the Jeffreys prior $p(G) \propto 1/G$:

$$q_0(G|\alpha,\beta,\mathbf{v}) \propto IG\left(G;\frac{T-1}{2},C\right), \text{where}$$

$$C = \sum_{i=2}^{T}\frac{\left(\alpha-(v_i-\beta v_{i-1})\right)^2}{2v_{i-1}}.$$

In this case, we need to use $q(\gamma|\dots) \propto f_{IG}\left(G;\frac{T-1}{2},C\right)\cdot\gamma$.

(iii) Sampling of the correlation parameter ρ given m,α,β,γ, $\mathbf{y} = \langle y_1, \dots, y_T\rangle$, and $\mathbf{v} = \langle v_1, \dots, v_T\rangle$.

Calculate ε_i and ε_i^V for $i = 2, \dots, T$ from the model (1) equations, and calculate the sample correlation $\hat{\rho} = corr(\varepsilon, \varepsilon^V)$. For sampling use the Fisher transformation $F(\rho) = artanh(\rho)$, i.e. sample $x \sim N(artanh(\widehat{\rho}), \frac{1}{T-4})$ and set the proposal $\rho_1 = \tanh(x)$.

Finally, use the Metropolis-Hastings accept-reject algorithm choosing between the previously sampled correlation estimate ρ_0 and the new proposal ρ_1, which is accepted with probability $\min(R, 1)$, where

$$R = \frac{p(\rho_1|\dots)q(\rho_0|\dots)}{p(\rho_0|\dots)q(\rho_1|\dots)},$$

$$p(\rho|\dots) = L(\mathbf{y}|\mathbf{v}, \Theta),$$

$$q(\rho|\dots) = \varphi\left(artanh(\rho); artanh(\hat{\rho}), \frac{1}{T-4}\right)\frac{1}{1-\rho^2}.$$

Sampling of the stochastic variance v_i given v_{i-1} , v_{i+1} , y_i, y_{i+1} and the SV equations parameters $m, \alpha, \beta, \gamma, \rho$:

Estimate the expected variance using a naïve GARCH-like model, e.g. as (for $1 < i < T$, analogously for $i = 1$, or $i = T$)

$$EV = w\frac{v_{i-1} + v_{i+1}}{2} + (1-w)(y_i - m)^2,$$

where w is a GARCH-like weight, for example 0.95. Then sample from the gamma distribution with the mean EV and sampling standard deviation $SV = \gamma\sqrt{v_{i-1}}$, i.e. $q(v|\dots) \sim Gamma(v; a, b)$, where $b = \frac{SV^2}{EV}$ and $a = \frac{EV}{b}$. For $i = 1$ and $i = T$ the proposal must be modified based only on the single neighboring variance. Finally, use Metropolis-Hastings with the acceptance ratio

$$R = \frac{p(v_1|\dots)q(v_0|\dots)}{p(v_0|\dots)q(v_1|\dots)},$$

where $p(v|..) \propto L(\mathbf{y}|\mathbf{v}, \Theta)$ with the remark that only the two (or one if $i = 1$ or $i = T$) terms involving v_i in the product defining the likelihood (3) need to be evaluated.

Initial values of the parameters and volatilities: In order to start the MCMC sampling procedure the values and latent variables of the parameters need to be appropriately initialized. Since we are going to use daily data with $\Delta t = 1/250$ we will start with "normally" observable annualized parameters of the continuous-time volatility equation, $V_{LT} = 0.01, \kappa = 2.5, \sigma_V = 50\%$, and transform them using the formulas given in Section 2.1. The initial mean parameter in (1) can be simply set equal to the mean of the observed returns $m = mean(\mathbf{y})$. Similarly, the initial variance can be estimated from the return series based on the simple EWMA (Exponential Weighted Moving Average) model setting $v_1 = var(\mathbf{y})$, and recursively $v_{i+1} = wv_i + (1-w)y_i^2$ for $i = 2, \dots, T$.

3. The Neural Network Training Dataset for the Heston Model

The neural network training is relatively straightforward once we define an appropriate dataset where features are selected historical returns' generalized moments and the vector of the Heston model parameters is the target.

We will generate daily returns and sample the vector of parameters $\Theta = \langle m, \alpha, \beta, \gamma, \rho \rangle$ by specifying certain "usual" ranges of the annualized parameters sampled from the uniform distribution:

- $\mu \in [0,0.3]$, the mean of returns,
- $\theta_V \in [0.1^2, 0.3^2]$, long term variance,
- $\kappa \in [1,5]$, speed of mean reversion,
- $\sigma_V \in \left[0.1, \min(0.5, \sqrt{2\kappa\theta_v} - 0.05\right]$, volatility of volatility where we need to ensure the Feller's condition, and
- $\rho_{SV} \in [-0.9,0.3]$, leverage effect parameter.

The sampled annual parameters are again transformed into the daily model parameters with $\Delta t = 1/250$ based on the formulas given in Section 2.1. In order to sample the volatility $\langle v_i \rangle_{i=1}^T$ and return time series $\langle y_i \rangle_{i=1}^T$, we also need to sample the initial variance: $V_0 \in [0.1^2, 0.3^2]$ in annualized terms. In our empirical study we will work with $T = 1000$ daily returns, but of course longer or shorter periods can be used as well.

Since we are going to apply a feed-forward NN the number of inputs needs to be limited by extracting appropriate characteristics (generalized moments) from the return time series. The generalized moments below follow Bollerslev and Zhou (2002) estimating the Heston model parameters with GMM, but the list of features is substantially more extensive. Unlike GMM where the weights of the moments are implicitly set by the researcher, the NN should utilize the important features only and assign them weights automatically during the training process. We are going to analyze the most important features in the empirical section using the LIME (Local Interpretable Model-Agnostic Explanations) method (Ribeiro et al., 2016). We propose to use the following 45 features that can be viewed as empirical proxies or characteristics related to the target variables:

- variance, skewness, kurtosis, and the 5th moment of $\langle y_i \rangle_{i=1}^T$,
- autocorrelations of $\langle y_i^2 \rangle_{i=1}^T$ (with lags 2-5, and the mean of autocorrelations with lags 6-10),
- autocorrelations of $\langle |y_i| \rangle_{i=1}^T$ (with lags 2-5),
- correlations $corr\left(y_{i-1}, y_i^2 - y_{i-1}^2\right)$ and $corr(y_{i-1}, |y_i| - |y_{i-1}|)$,
- calculated realized variances $RV_j = \sum y_i^2$, $RA_j = \sum |y_i|$, and cumulative returns $Rc_j = \sum y_i$ over non-overlapping periods of $D = 10$ and 20 days,
- mean, variance, skewness, kurtosis, and autocorrelations (with lags 2-3) of RV,
- correlations $corr(RV_j, Rc_j)$, $corr(RV_j, RA_j)$, $corr(RV_j - RV_{j-1}, Rc_j)$, $corr(RV_j - RV_{j-1}, Rc_{j-1})$, $corr(RV_j - RV_{j-1}, RA_j)$, and $corr\left(RV_j - RV_{j-1}, RA_{j-1}\right)$,
- AR(1) intercept of RV (proxy of alpha),
- autocorrelation of RV^2, i.e. $corr(RV_{j-1}^2, RV_j^2)$.

Finally, we will normalize the features and the target variables (using the min-max linear method) based on the minimum and maximum of the variables in the dataset so that each variable range is between 0 and 1. This is in particular important for the target variables since otherwise the standard sum-of-squared-differences error function would implicitly assign different weights to the target variables depending on their magnitudes. Regarding the size of the training dataset, we will use $N = 50\,000$, but smaller or larger datasets can be certainly sampled depending on the computational resources. The training of a feed-forward neural network will be performed in Matlab (2021b) using the standard Levenberg-Marquardt algorithm (Levenberg, 1944, or Marquardt, 1963).

Figure 3 - A neural network training summary report produced by Matlab

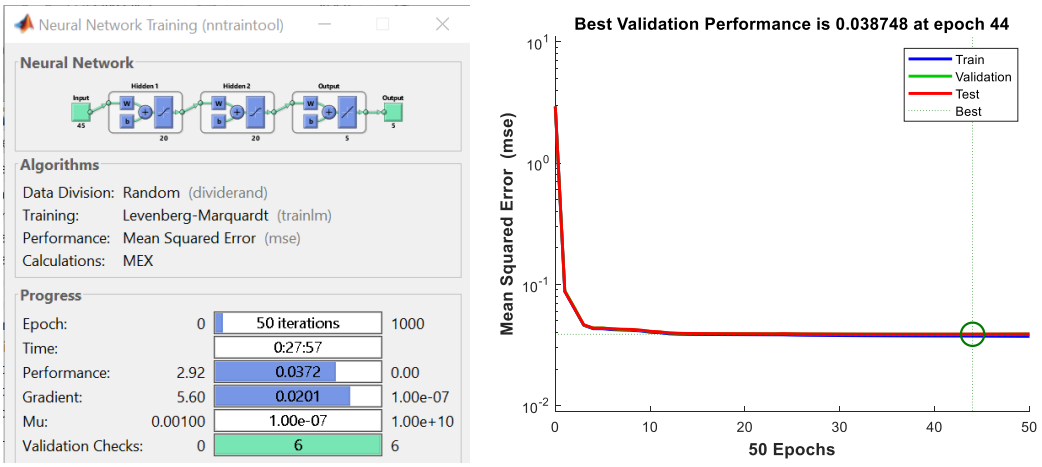


Figure 3 shows an example of a summary report for training of a neural network with two hidden layers, each with 20 neurons (with the sigmoid activation function). The reported performance 0.0372 in terms of the mean-square-error of the standardized target can be interpreted (after taking the square root) as an average relative error of the estimation model on the testing dataset. Note that the training procedure took approximately 28 minutes on a standard notebook computer with the Intel i7 processor. This is a one-time larger computational cost, in fact much smaller than in case of the MCMC algorithm, and then the evaluation of the estimates on a given set of inputs takes just a fraction of a second. The algorithm randomly splits the original training dataset into the "Train", "Validation", and "Test" datasets (in the proportion 70:15:15), and optimizes the error on the "Train" dataset periodically checking the performance on the "Validation" dataset. The final performance is reported on the "Test" dataset. However, to double check the performance we shall also sample an additional testing dataset (e.g. with 2 000 observations) that will not be used in the training procedure at all, and in the empirical section we report the performance on this pure out-of-sample dataset.

4. Empirical results

4.1. The methodology of comparison between the MCMC and NN estimation methods

As outlined in the introduction, the advantage of the MCMC method is that it gives an empirical approximation of the full posterior distribution $p(\Theta|data)$ of the parameters given the data, i.e. the time series of returns in our case. In practice, the mean, median, or mode of the distribution can be used as the point estimates $\widehat{\Theta} = f_{MCMC}(data)$ and, in addition, the empirical density gives us a full picture of the Bayesian error distribution. Therefore, the variance of $p(\Theta|data)$ can be interpreted as the irreducible error that cannot be eliminated by any statistical estimator.

Generally, 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*** given by $p(Y|X)$, the estimator $\hat{f}(X)$ ***variance,*** and the ***bias***:

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

Note that the estimator $\hat{f}_T(X)$ depends also on the training dataset T, implicitly assuming that (X,Y) drawn from the testing dataset is independent on the set T (see Hastie et al., 2009).

If the target Y was a function of X, then the irreducible error would be equal to zero! However, if Y is not uniquely determined by X incorporating some noise or some unknown information, then the irreducible error becomes positive, i.e. it depends on the "information content" of the vector of features!

Therefore, the goal might be to compare the error of the NN estimator with the standard deviation of $p(\Theta|data)$. However, there are at least two issues with this approach. Firstly, the NN model (feed-forward NN), gives us only point estimates $f_{NN}(data)$ and it is not obvious how to approximate the posterior distribution $p(\Theta|data)$ or at least its variance using a NN model. Secondly, the MCMC point estimates based on a feasible computational time often turn out to have a substantial error with respect to the true values. Hence, the MCMC error should incorporate not only variance of $p(\Theta|data)$, but also variance of the point estimate $\widehat{\Theta} = f_{MCMC}(data) \approx E[\Theta|data]$.

We shall report the standard deviation of $p(\Theta|data)$ or its average value over a collection of data (sampled time series) used as input of the MCMC algorithm. However, to conclude the discussion, we will simply focus on RMSE, R-squared, or the bias based on the sampled parameters Θ_j (and return series $data_j$) and the point estimates $\widehat{\Theta}_j$, i.e.

$$RMSE = \sqrt{\frac{1}{K}\sum_1^K \left(\widehat{\Theta}_j - \Theta_j\right)^2},$$

for both estimation methods. Alternatively, we will also report the results with a fixed "typical" parameter vector Θ and sample only the time series of returns $data_j$.

4.2. MCMC estimation results

To illustrate the performance of the MCMC algorithm, we have firstly fixed a set of "true" HM parameters (see the 1st row of Table 1), repeated (100 times) sampling of the daily return time series (of length 1000), and run (for sampled return series) the MCMC estimation algorithm with 3000 iterations. Figure 4 gives example of the iteratively sampled estimates of the most problematic parameter ρ fluctuating around its true value set to -0.5. It indicates that the convergence to the posterior distribution is relatively fast, and, at the same time that the irreducible error for this parameter (posterior variance) will be quite large. The acceptance ratio of the ρ parameter proposals (and similarly for the other parameters) is around 60-90% indicating that the algorithm works well. We have also tested implementation of the MCMC algorithm with a larger number of iterations, e.g. 6 000 or 10 000, and achieved only a small improvement against a substantially higher computational time.

Figure 4 - An example of the parameter ρ MCMC estimates convergence

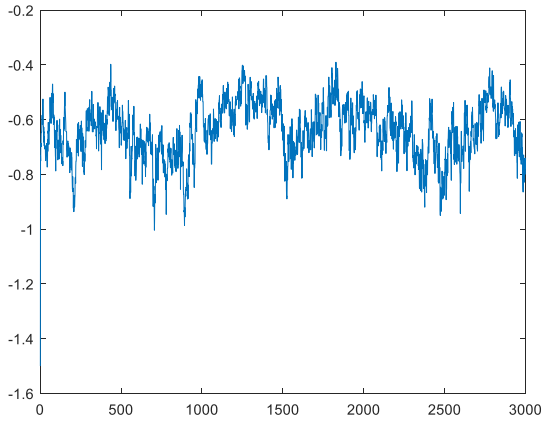


Table 1 summarizes the results of the 100 MCMC estimations where the point estimates are defined as means of the sampled values with the burnout period set to 1 499, i.e. as the means of values sampled in the 1500th to 3000th iteration. The table gives the mean of the 100 MCMC estimates and the mean of the 100 posterior standard deviations of the individual parameters. In addition, it shows the RMSE and bias of the point estimates with respect to the true values. For example, focusing on ρ, the estimated average posterior standard deviation corresponding to the theoretical irreducible error is relatively large 0.0962, but in addition the RMSE with respect to the true value is even larger (0.1974) caused mainly by the bias (0.1294). The bias and the RMSE should theoretically disappear if the length of the chain converged to infinity, which is, of course, impossible in practice. The errors of the other parameters appear to be much smaller, but this is to large extent caused by their low levels or narrow ranges sampled in the training dataset (see Section 3). For a better relative comparison, one should look at the values normalized with respect to the minimum and maximum values of the parameters in the training dataset. Then it turns out that the relative errors (RMSE_n and Bias_n) are also quite large, between 10% to 60%, with the overall normalized RMSE being approximately 28%.

Table 1 – MCMC results with fixed parameters and based on 100 estimation runs

	m	alpha	beta	gamma	rho
True val.	0.0012	0.0000	0.9900	0.0013	**-0.5000**
Mean est.	0.0011	0.0000	0.9824	0.0013	**-0.3706**
RMSE	0.0004	0.0000	0.0101	0.0002	**0.1974**
Bias	-0.0001	0.0000	-0.0076	0.0001	**0.1294**
Post std	0.0003	0.0000	0.0061	0.0001	**0.0962**
RMSE_n	0.2938	0.1119	0.6294	0.1324	**0.1645**
Bias_n	-0.0504	0.1119	-0.4733	0.0300	**0.1079**

The performance of the MCMC method on 500 parameter vectors sampled from the ranges specified in Section 3 is reported in Table 2. Again, focusing firstly on the parameter ρ, we can notice that the average posterior standard deviation is approximately the same (0.1022) as in the case with fixed parameters, but that the point estimate $RMSE = \sqrt{\frac{1}{500}\sum_1^{500}\left(\hat{\rho}_j - \rho_j\right)^2}$ with variable Θ_j turns out to be substantially larger (0.3316) which is caused mainly by the bias

(0.1872). The R^2 of the ρ estimate is not too large, but at least positive (0.2012). It is interesting to note that the R^2 of all the other parameter are negative which is caused by relatively large estimation biases. A better comparison of the estimation precision in terms of RMSE and bias can be obtained looking rather on the normalized values in the last two rows of Table 2. The mean normalized RMSE across all parameters appears very high at around 55%.

The calculation took over 20 hours of Matlab 2021b four workers parallel computing on a Core i7 desktop computer. As mentioned above the results improved only slightly when the number of MCMC iterations was increased to 10 000, but the computational time went to more than 60 hours.

Table 2 – MCMC results with variable parameters sampled 500 times

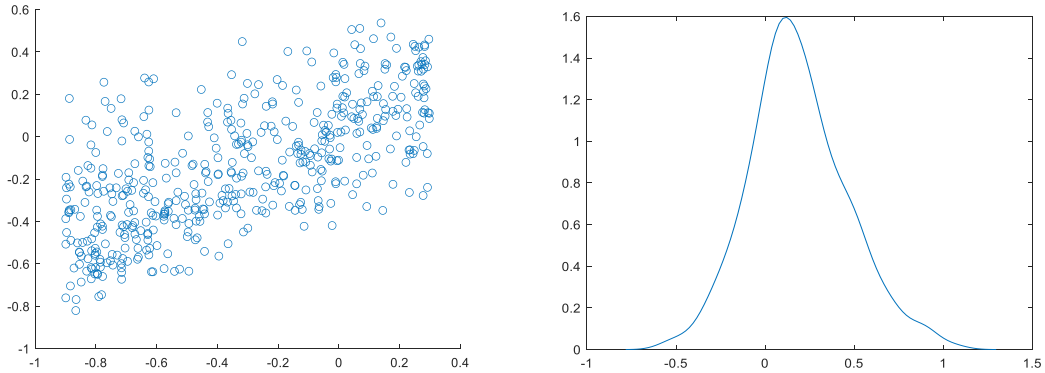
	m	alpha	beta	gamma	rho
RMSE	0.0004	0.0000	0.0197	0.0005	**0.3316**
Bias	-0.0000	0.0000	-0.0156	0.0004	**0.1872**
R^2	-0.5944	-8.1504	-17.5456	-0.2145	**0.2012**
Post std	0.0004	0.0000	0.0082	0.0001	**0.1022**
RMSE_n	0.3643	0.5802	1.2288	0.2875	**0.2764**
Bias_n	-0.0396	0.3917	-0.9637	0.2078	**0.1560**

The performance of the estimations can be further analyzed, for example, with the Mincer-Zarnowitz test regressing the true parameter values against the estimates, $\theta = \alpha + \beta\hat{\theta} + \varepsilon$. For example, in case of ρ, the Mincer-Zarnowitz regression correcting for the bias and slope (α and β) shows a relatively good R-squared of 0.467. The issue with the quite low slope (that should be optimally equal to 1) is also illustrated by the scatter-plot in the left part of Figure 5. The right-hand part of the figure shows the wide distribution and a large bias of the estimation errors, i.e. of $\hat{\rho}_j - \rho_j$, $j = 1, \ldots, 500$.

Table 3 – Mincer-Zarnowitz regression of true versus estimated values of ρ (MCMC)

	Estimate	s.e.	t-statistics	p-value
$\boldsymbol{\alpha}$	-0.2063	0.0134	-15.376	6.09e-44
$\boldsymbol{\beta}$	0.8624	0.0413	20.871	5.698e-70
Number of observations: 500, Error degrees of freedom: 498				
RMSE: 0.271, **R-squared: 0.467**, Adjusted R-squared: 0.466				

Figure 5 - The true vs. estimated ρ scatter plot (left figure) and the estimation error distribution (MCMC)



4.3. NN estimation results

Next, let us look at the performance of the NN estimator. As described in Section 3 the NN with two hidden layers (both with 20 neurons) was trained on a dataset of 50 000 synthetically generated observations, i.e. sets of generalized moments (features) calculated from daily return time series of length 1000 conditional on sampled parameter vectors (targets). The training itself took less than 30 minutes with results given in Figure 3. The reported performance (0.0372) should be interpreted as the mean-squared error of the normalized parameter estimates corresponding to the average normalized RMSE around 0.193.

Let us firstly look at the NN out-sample performance (Table 4) when we fix the set of parameters as above (compare with Table 1). Since one estimation takes less than a second, we can easily run it 500 times (sampling the time series and the generalized moments based on the fixed vector of parameters). Table 4 demonstrates substantially better performance of the NN estimator compared to MCMC with an overall normalized RMSE equal approximately to 19% (compared to 28% in case of MCMC). The better RMSE appears to be related mainly to much lower biases, for example just 0.0217 in case of ρ (compared to 0.1294 for the MCMC).

Table 4 – NN estimation results with fixed parameters and based on 500 times sampled return series

	m	alpha	beta	gamma	rho
True val.	0.0012	0.0000	0.9900	0.0013	**-0.5000**
Mean est.	0.0008	0.0000	0.9880	0.0013	**-0.4783**
RMSE	0.0004	0.0000	0.0030	0.0002	**0.1694**
Bias	-0.0004	0.0000	-0.0020	0.0000	**0.0217**
RMSE_n	0.3347	0.0968	0.1850	0.1227	**0.1412**
Bias_n	-0.2950	0.0681	-0.1254	0.0091	**0.0181**

Finally, Table 5 reports the NN performance on the full out-of-sample testing dataset with variable parameters and 2 000 observations. The overall normalized RMSE (19%) is again substantially lower than in case of MCMC (55%). It should be noted that in case of the variable parameter testing dataset the average biases turn out to be almost negligible, both in absolute

and relative terms, and unlike in case of MCMC, the R-squared values are positive and relatively high for all the parameters.

Table 5 – NN estimation results with variable parameters on the testing dataset with 2 000 observations

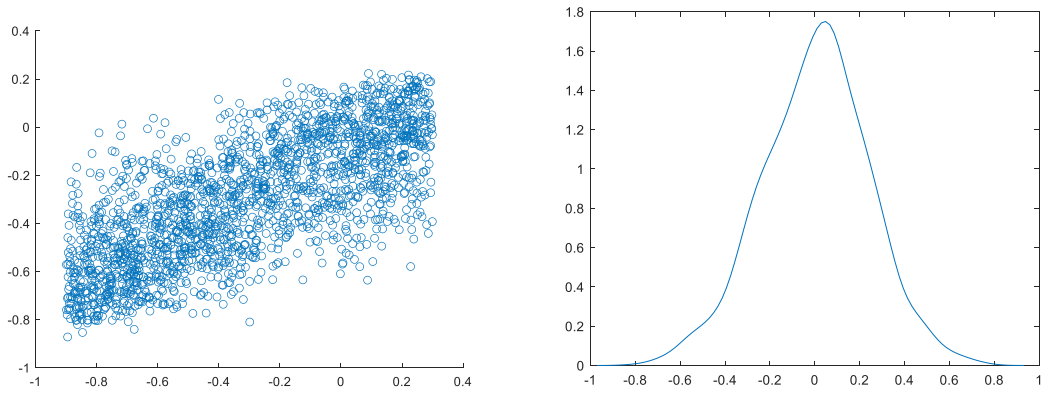
	m	alpha	beta	gamma	rho
RMSE	0.0003	0.0000	0.0043	0.0002	**0.2389**
Bias	0.0000	0.0000	-0.0000	0.0000	**0.0025**
R^2	0.3701	0.6328	0.1566	0.7631	**0.5325**
RMSE_n	0.0003	0.0000	0.0043	0.0002	**0.2389**
Bias_n	0.0000	0.0000	-0.0000	0.0000	**0.0025**

For the sake of completeness, Table 6 reports the Mincer-Zarnowitz regression results. In this case, the NN estimator passes the test with the intercept not being significantly different from 0 and the slope not significantly different from 1. Figure 6 also visually indicates superiority of the NN estimator compared to MCMC (see Figure 5).

Table 6 – Mincer-Zarnowitz regression of true versus estimated values of ρ (NN)

	Estimate	s.e.	t-statistics	p-value
α	-0.0025	0.0083	-0.3021	0.76255
β	1.0021	0.0210	47,711	0
Number of observations:2000, Error degrees of freedom: 1998				
RMSE: 0.239, **R-squared: 0.533**, Adjusted R-squared: 0.532				

Figure 6 - The true vs. estimated ρ scatter plot (left figure) and the estimation error distribution (NN)



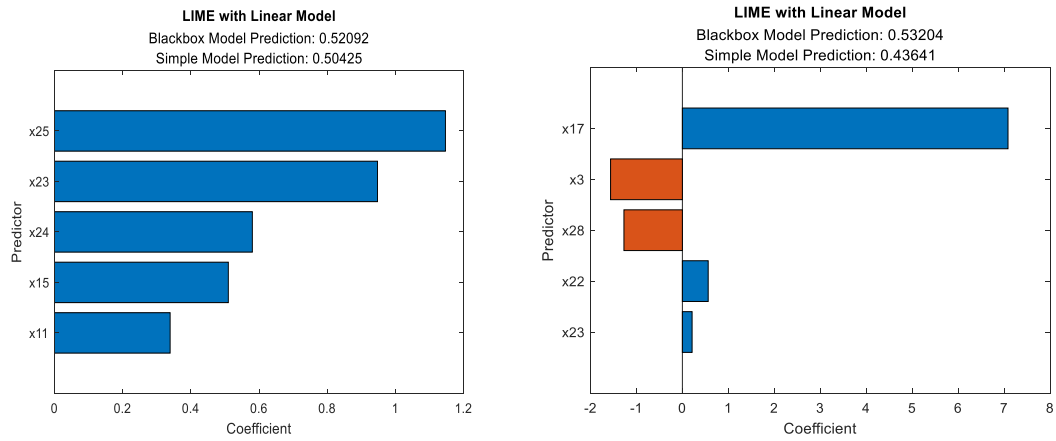
4.4. Interpretability of the NN model

One of the disadvantages of neural networks, often discussed by researchers and practitioners (see e.g. Hull, 2021), is their Blackbox character, i.e. their difficult interpretability in terms of dependence of the output targer values on the input feature values. The most well-known methods to handle the issues are the LIME (Local Interpretable Model-Agnostic Explanations) and the method of Shapley values.

Shapley values calculate average impact of each feature value of a query point with respect to its expectation considering all possible combinations of the other features values at the query point. On the other hand, Lime simply interpolates the data locally at a query point by a linear model (or tree for a categorical model), selects a number of variables and shows the prediction of the approximate simple model. In case of a higher dimensional target, the analysis needs to be done separately for individual coordinates of the output vector.

In our case, we consider the Lime method as easier to apply and interpret in order to check that the most important explanatory variables (features) and their impact correspond with our intuition.

Figure 7 - The most important 5 variables and their coefficients estimated by LIME for ρ and β



For example, Figure 7 shows the five most important features selected by Lime and the estimated coefficients of the approximating model predicting ρ (left) and β (right). The query point at which the models are evaluated is set to the mode of the (testing) dataset, however the estimated Lime model is, in fact, global, i.e. it is the linear model estimated on the full dataset (with the min-max standardized feature and target values). It turns out that the selected features and their signs are consistent with our intuition and document importance of the concept of realized volatility.

The five selected features for ρ in the order shown on the figure (left part) are: $corr(RV_j, Rc_j)$, $corr(RV_j - RV_{j-1}, Rc_j)$, $corr(RV_j - RV_{j-1}, Rc_{j-1})$, $corr(y_{t-1,}|y_t| - |y_{t-1}|)$, and $corr(y_{t,}y_{t-2})$. The five selected features for β (right part) are: $var(RV)$, $kurtosis(RV)$, AR (1) intercept of RV, and lag 1-2 autocorrelations of RV, where RV is the 20-day realized variance.

5. Discussion and Conclusions

We have outlined a flexible neural network application to estimation of unknown parameters of a model based on a large synthetically generated dataset conditional on the data-generating model. Our specific hypothesis was that this approach can provide more precise and computationally efficient estimates of the Heston model parameters compared to state-of-the-art probabilistic methods such as the Bayesian MCMC. We have implemented and compared both methods on relatively large testing datasets. The empirical results have confirmed our hypothesis in both directions: the NN parameter estimation approach is substantially faster compared to the MCMC method, and in addition the estimations are more precise. To mention some advantages of MCMC, this method gives an empirical approximation of the posterior parameter distribution and estimates the latent state variables (i.e. stochastic volatilities in case

of the HM) together with the model parameters. On the other hand, an important advantage of the NN approach is its relative simplicity compared to the MCMC method requiring a careful formulation of marginal and proposal densities. We have applied the simplest feed-forward NN requiring to condense a large time series to a limited number of features defined as a set of selected generalized moments. However, we believe that even this "manual" part of the estimation procedure could be eliminated applying the convolutional or similar NN, which is a direction of our further research.

Regarding the provocative subtitle "Forget Statistics – Machine Learning is Sufficient!" of the paper, we do believe that the NN estimation approach might indeed outperform the classical methods for a large class of models. This general hypothesis has been confirmed in case of the HM, but, of course, it remains to be tested on many other models, in particular on more complex latent state space financial models that are difficult to estimate using the classical methods. To conclude in more moderate and realistic terms, we believe that the NN approach can provide more efficient estimates compared to the classical statistical methods, but we agree that the classical probability and statistics theory is of course needed in order to study and assess properties of such novel machine learning methods.

6. References

[1] Bollerslev, T., & Zhou, H. (2002). Estimating stochastic volatility diffusion using conditional moments of integrated volatility. Journal of Econometrics, 109(1), 33-65.

[2] 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.

[3] Feller, W. (1991). An introduction to probability theory and its applications, Volume 2 (Vol. 81). John Wiley & Sons.

[4] Frühwirth-Schnatter, S., & Sögner, L. (2003). Bayesian estimation of the Heston stochastic volatility model. In Operations Research Proceedings 2002: Selected Papers of the International Conference on Operations Research (SOR 2002), Klagenfurt, September 2–5, 2002 (pp. 480-485). Springer Berlin Heidelberg.

[5] 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.

[6] 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

[7] Hull, J. (2021). Machine Learning in Business: An Introduction to the World of Data Science, Independently Published, p. 287.

[8] Johannes, M., Polson, N. (2009). “MCMC Methods for Financial Econometrics.” In Ait-Sahalia, Hansen, L. P., eds., Handbook of Financial Econometrics. pp. 1-72.

[9] Levenberg, Kenneth (1944). "A Method for the Solution of Certain Non-Linear Problems in Least Squares". Quarterly of Applied Mathematics. 2 (2): 164–168. doi:10.1090/qam/10666.

[10] Lynch, S. M. (2007). Introduction to Applied Bayesian Statistics and Estimation for Social Scientists. Springer, pp. 359.

[11] Marquardt, Donald (1963). "An Algorithm for Least-Squares Estimation of Nonlinear Parameters". SIAM Journal on Applied Mathematics. 11 (2): 431–441. doi:10.1137/0111030. hdl:10338.dmlcz/104299.

[12] Ribeiro, M. T., Singh, S., & Guestrin, C. (2016). " Why should i trust you?" Explaining the predictions of any classifier. In Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining (pp. 1135-1144).

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

[14] Speekenbrink, M. (2016). A tutorial on particle filters. Journal of Mathematical Psychology, 73, 140-152.

[15] 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

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.
2. David Mazáček, Jiří Panoš: Key determinants of new residential real estate prices in Prague.
3. Milan Fičura: Impact of size and volume on cryptocurrency momentum and reversal.
4. Masood Tadi, Jiří Witzany: Copula-Based Trading of Cointegrated Cryptocurrency Pairs.
5. Pavel Jankulár, Zdeněk Tůma: Changes to Bank Capital Ratios and their Drivers Prior and During COVID-19 Pandemic: Evidence from the EU.
6. Teona Shugliashvili: The words have power: the impact of news on exchange rates.
7. Jiří Witzany, Milan Fičura: A Comparison of Neural Networks and Bayesian MCMC for the Heston Model Estimation *(Forget Statistics – Machine Learning is Sufficient!)*

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