


Deep learning the Hurst parameter of linear fractional processes and assessing its reliability

Dániel Boros¹  | Bálint Csanády² | Iván Ivkovic³ | Lóránt Nagy³ |
András Lukács² | László Márkus¹ 

¹Department of Probability Theory and Statistics, Institute of Mathematics, Eötvös Loránd University, Budapest, Hungary

²Department of Computer Science, Institute of Mathematics, Eötvös Loránd University, Budapest, Hungary

³Rényi Institute, Budapest, Hungary

Correspondence

László Márkus, Department of Probability Theory and Statistics, Institute of Mathematics, Eötvös Loránd University, Budapest, Hungary.
Email: laszlo.markus@ttk.elte.hu

Present address

László Márkus, Pázmány P. St. 1/C, H-1117, Budapest, Hungary

Abstract

This research explores the reliability of deep learning, specifically Long Short-Term Memory (LSTM) networks, for estimating the Hurst parameter in fractional stochastic processes. The study focuses on three types of processes: fractional Brownian motion (fBm), fractional Ornstein–Uhlenbeck (fOU) process, and linear fractional stable motions (lfsm). The work involves a fast generation of extensive datasets for fBm and fOU to train the LSTM network on a large volume of data in a feasible time. The study analyses the accuracy of the LSTM network's Hurst parameter estimation regarding various performance measures like root mean squared error (RMSE), mean absolute error (MAE), mean relative error (MRE), and quantiles of the absolute and relative errors. It finds that LSTM outperforms the traditional statistical methods in the case of fBm and fOU processes; however, it has limited accuracy on lfsm processes. The research also delves into the implications of training length and valuation sequence length on the LSTM's performance. The methodology is applied to estimating the Hurst parameter in li-ion battery degradation data and obtaining confidence bounds for the estimation. The study concludes that while deep learning methods show promise in parameter estimation of fractional processes, their effectiveness is contingent on the process type and the quality of training data.

KEYWORDS

accuracy of deep learning, fractional stochastic processes, Hurst parameter estimation, li-ion battery degradation analysis, LSTM network

1 | INTRODUCTION

Fractional processes play a pivotal role in the stochastic modeling of diverse phenomena, including the wear and tear of machinery and the valuation of financial instruments. A key parameter in these models is the Hurst exponent, which may reflect memory decay, self-similarity, fractal dimension (FD), or all of them simultaneously, as in fractional Brownian motion (fBm).

This is an open access article under the terms of the [Creative Commons Attribution-NonCommercial-NoDerivs](https://creativecommons.org/licenses/by-nc-nd/4.0/) License, which permits use and distribution in any medium, provided the original work is properly cited, the use is non-commercial and no modifications or adaptations are made.

© 2024 The Author(s). *Quality and Reliability Engineering International* published by John Wiley & Sons Ltd.

The model's accuracy is highly sensitive to the Hurst exponent, highlighting its criticality; even slight errors in estimating this parameter can cause significant discrepancies in the model's effectiveness and predictive capability. Consequently, it is vital to comprehend and precisely quantify the margin of error associated with this parameter's estimation. While conventional statistical methods for estimating this parameter have been thoroughly explored, the potential of deep learning techniques in this area warrants investigation. Our research focuses on three process types: fBm, fractional Ornstein–Uhlenbeck (fOU) process, and linear fractional stable motions (lfsm).

Degradation modeling and reliability analysis are fundamental to the Prognostics and Health Management of complex systems. With advancements in measurement technologies, a notable long-range dependence (LRD) has been observed in the degradation patterns of various assets, including turbofan engines, blast furnaces, lithium-ion batteries, and chemical catalysts.^{1–3} This LRD, also known as long-term memory effect or persistence, indicates that correlations of asset degradation increments extend over extensive periods. Therefore, future degradation trends are not only influenced by the current condition of the asset but are also deeply linked to its entire degradation history. The LRD's significance extends beyond asset performance degradation, being prevalent in fields like finance, hydrology, and biology. Consequently, numerous models integrating LRD have been developed for asset degradation and reliability analysis.

To monitor the long-term memory LRD in degradation patterns effectively, several researchers have incorporated fBm in asset degradation models. The memory structure of fBm is delineated by the Hurst parameter H , which lies within the range $0 < H < 1$. Specifically, when $0.5 < H < 1$ fBm exhibits LRD, making it increasingly relevant for analyzing LRD-integrated degradation data across various assets. Examples of its application include Xi et al.'s⁴ model that leverages the LRD for predicting the remaining useful life of turbo engines, Zhang et al.'s⁵ integration of multiple degradation modes in an fBm-based model, Si et al.'s⁶ application in accelerated degradation tests, and Zhang et al.'s⁷ development of an explicit probability density function for remaining useful life estimation within an LRD-integrated framework. These models exemplify the growing scope of fBm applications in asset degradation analysis.^{8–10} The fBm solely relies on the Hurst parameter to characterize the degradation process, which may not sufficiently capture complex time series data attributes. Nonetheless, practical degradation processes typically exhibit non-Gaussian traits, whereas the fBm process conforms to Gaussian distributions.

Conversely, Lévy stable motion exhibits non-Gaussian stable distributions and is characterized by parameters α , β , γ , and δ . The parameter α , known as the stability index, controls the tail thickness of the distribution; lower values of α suggest heavier tails, indicating a higher likelihood of significant deviations from the mean. The skewness parameter β dictates the asymmetry of the distribution, with $\beta = 0$ representing a symmetric distribution, positive β values indicating a right-skewed distribution, and negative β values a left-skewed one. The scale parameter γ determines the width of the distribution, influencing its dispersion, while the location parameter δ sets the center of the distribution, around which it is symmetric in the case of $\beta = 0$. This motion, simplifying to Brownian motion when $\alpha = 2$, is distinguished by its heavy-tailed nature, with a probability density that diminishes following a power law. The lfsm, as an extension of Lévy stable motion, encapsulates both non-Gaussian characteristics and heavy-tailed properties, as well as exhibiting LRD. In this model, positive LRD is indicated when $H > \frac{1}{\alpha}$, suggesting continuity in future trends with the present. Conversely, $H < \frac{1}{\alpha}$ denotes the absence of LRD, implying contrasting future trends (i.e., future trends that differ from or are opposite to current trends). When $H = \frac{1}{\alpha}$, the process is deemed independent of past trends. Notably, in the $\alpha = 2$ case, the lfsm model aligns with the fBm. Thus, lfsm presents a versatile framework capable of modeling a broad spectrum of stochastic processes, ranging from heavy-tailed to both Gaussian and non-Gaussian distributions.

In the realm of industrial production, where sensor data is abundantly collected, data-driven methodologies are increasingly vital for predicting performance degradation in complex mechanical systems. These methods are broadly categorized into statistical and deep learning-based approaches. Deep learning techniques, including convolutional neural networks, long short-term memory (LSTM), deep belief networks (DBN), deep autoencoders (DAE), and transfer learning (TL), have garnered significant attention in remaining useful life (RUL) prediction. For instance, Ding et al.¹¹ introduced an RUL prediction method for rolling bearings based on deep convolutional neural network (DCNN), enhancing feature learning capabilities. Qin et al.¹² developed a neural network with a gated dual attention unit for predicting the RUL of rolling element bearings. Yan et al.¹³ crafted an ON-LSTM network for gear RUL estimation, integrating tree structures within LSTM to improve prediction accuracy. Haris et al.¹⁴ combined DBN with Bayesian optimization for estimating supercapacitor RUL, while Wang et al.¹⁵ utilized DAE for feature extraction and LSTM for electric valve RUL prediction. Fan et al.¹⁶ proposed a feature-based TL approach to extend RUL prediction models from simpler to more complex domains.

While deep learning approaches have achieved notable successes in predicting the RUL of equipment, these algorithms' effectiveness is heavily reliant on the quality and volume of the training data, leading to inefficiencies in model training, particularly with combinatorial network models. Additionally, the determination of hyper-parameters in these algorithms

often involves laborious cross-validation or heuristic methods, limiting their practical application in industrial settings. A significant limitation of most deep learning methods is their focus on point-wise RUL predictions, which struggle to accurately quantify the uncertainty in RUL projections under varying conditions, thereby inadequately addressing risk assessment in predictions. In contrast, statistical data-driven methods, which do not depend on extensive degradation data, possess intrinsic strengths in accurately quantifying the uncertainties associated with RUL predictions, offering a more robust framework for risk assessment.

The primary aim of our research is the swift and accurate estimation or calibration of these fractional processes in complex models used across various sectors, such as finance and reliability engineering. In the domain of financial volatility modeling, the Hurst exponent frequently falls below 0.5, leading to rough paths. Conversely, in reliability engineering, particularly in degradation or RUL analysis, the Hurst exponent often exceeds 0.5, indicating a long memory effect.

It is not our objective to find a universal structure that works for all models in degradation modeling or in RUL analysis. Concerning battery degradation, which is our primary motivation in the application field examined in this article, there are two modeling practices, where either fBm/fOU or lfsm processes are utilized. Our intention was to highlight that if we think within one modeling framework and train the network within that scope, it will not be suitable for use in the other modeling framework. The same network structure cannot be appropriately applied in both contexts, and this is not remedied by mixing data or fundamentally training the network with data from the other structure.

We train neural networks using extensive and accurately simulated datasets, ensuring the Hurst exponent's variation covers its entire spectrum from 0 to 1. To generate our training samples, we implement a sophisticated Davies–Harte-type algorithm capable of efficiently producing sample paths from isonormal processes, encompassing the fBm and fOU processes. For estimating parameters of the fOU process, we utilize a unidirectional LSTM network, comprising two layers and enhanced with a normalization layer.

While current deep-learning-based research predominantly concentrates on *predicting* risks or losses, the dynamics of risk propagation and methods to impede this spread are comparatively underexplored. To enhance production safety, product reliability, or financial stability, a blend of targeted risk response strategies and an ambient, all-encompassing risk prevention framework is essential. In addressing this, we propose integrating the study of *propagation dynamics* with deep learning methodologies to thoroughly model and scrutinize the spread of risks. Keeping this perspective, our goal is to utilize deep learning to unravel the dynamic parameters of stochastic processes that underlie or stimulate risk propagation.

In recent years, a number of works have emerged that utilize neural networks for parameter estimation problems.¹⁷ It is particularly so in estimating the Hurst parameter of fBm. Those applying multilayer perceptions (MLPs) have to cope with the required fixed input size and hence have one of the following alternatives: either inference can be performed only on a fixed-length series,^{18,19} or inference is made on a set of process-specific statistical measures enabling a fixed size input to the neural networks.^{20,21} A more recent signature-based method, described in Bonnier et al. (2019),²² is also capable of estimating fBm's Hurst parameter. In executing that, the extracted statistical descriptors are processed by an LSTM.

The hybrid application of statistical descriptors and neural networks employed in the mentioned methods does not bring significant improvement compared to our purely neural network solution while increasing the computational time sometimes unbearably. Another common weakness of the recently published methods is that they do not address the possible limitations caused by scaled inputs.

A major advancement of our research lies in the creation of a neural network-based methodology, which markedly surpasses traditional statistical methods in determining the parameters of the fBm and fOU processes. This method excels in accuracy and speed compared to previous statistical and deep learning approaches. To achieve it, we need large-scale teaching of the network, and to do that, we have successfully invented a very fast generation of extensive fBm datasets for training purposes.

2 | TEACHING THE NEURAL NETWORK

2.1 | The processes

A **fBm**,²³ denoted as $B_H(t)$, is a Gaussian process starting from zero and characterized by continuous time, zero mean, and the autocovariance function

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

The fBm is notable for having stationary and **dependent** increments and is recognized for being a **self-similar** process with **fractal** paths. The Hurst exponent H is its sole parameter. According to Robert Adler's seminal work, **Brownian motion** and Stochastic Differential Equation driven **diffusion processes** invariably exhibit FDs of 1.5. That highlights the inadequacy of these processes for modeling phenomena with varying FDs, thus underscoring the significance of fBm. Notably, the FD of fBm paths varies with the Hurst parameter H , so that $FD = 2 - H$.

A **fOU** process is defined²⁴ by a fractional stochastic Langevin differential equation

$$dX(t) = \kappa(\theta - X(t))dt + \sigma dB_H(t), \quad (1)$$

where the process is driven by an fBm $B_H(t)$ with **Hurst** parameter $H \in (0, 1)$. Both the **drift** parameter κ and the **volatility** parameter σ are positive real constants. The solution to the fOU equation is known to exist and is unique, subject to an initial condition. It can be expressed explicitly, particularly for a 0 initialization and expectation, as follows:

$$X(t) = -\sigma \int_0^t e^{-\kappa(t-s)} dB_H(s).$$

The fOU process is an **isonormal**²⁵ (Definition 1.1.1)¹ and, therefore, a Gaussian process belonging to the first Wiener-Ito chaos driven by fBm. It achieves a unique stationary solution when initiated in a stationary state. The paths of the fOU process inherit their **FD** from the driving fBm. The fOU process is characterized by four parameters: H , κ , θ , and σ .^{24,26,27}

The definition of the fOU process²⁴ we use includes the parameter θ . This inclusion allows for the mean-reverting behavior, which is especially relevant in financial modeling contexts where such dynamics are prevalent. In the financial literature it is almost exclusively referred to as the rough Vasicek model,²⁸ and widely recognized for modeling interest rates.

When θ is set to zero, the model simplifies to a standard fOU process, which does not exhibit mean-reverting behavior. This flexibility in defining the fOU process allows for a broader application range, catering to various scenarios encountered in financial time series analysis and beyond. Since the application in the present paper is not of financial nature, we did not find appropriate to refer to the model by Vasicek's name.

The fBm has different extensions to the α -stable case. One of the most commonly used is the **linear fractional stable motion (lfsM)**. This process is also called linear fractional Lévy motion (lfLm) or fractional Lévy stable motion (fLsm).

The lfsM $L_H^\alpha(t)$ is defined as the stochastic process given by the integral²⁹ (formula: 7.7.4)

$$L_H^\alpha(t) = \int_{-\infty}^{\infty} \left(((t-x)_+)^{H-1/\alpha} - ((-x)_+)^{H-1/\alpha} \right) dM(x)$$

where $0 < \alpha < 2$, is the parameter of stability, $0 < H < 1$, $H \neq \frac{1}{\alpha}$ is the Hurst exponent and M is an α -stable random measure on \mathbb{R} with Lebesgue control measure. For any $z \in \mathbb{R}$ $(z)_+ = \max(z, 0)$.

The lfsM is a fractional **self-similar** stable process with **stationary increments** (H-sssi), where H is the Hurst exponent of fractionality (for more details, see Samorodnitsky & Taqqu, 1994²⁹). The parameter H characterizes the self-similarity property of lfsM. The marginal distribution of $L_H^\alpha(t)$ is α stable. It follows that $P(L_H^\alpha(t) > x) \propto t^{\alpha H} x^{-\alpha}$ as $x \rightarrow \infty$, and that the generalized dispersion defined in the quantile sense satisfies $D(L_H^\alpha(t)) \propto t^H$, which is similar to the behavior of fBm. If $1/\alpha < H < 1$, and $\alpha \in (1, 2)$, this process may be shown to present LRD in some extended sense given in Samorodnitsky and Taqqu (1994).²⁹ It has to be clearly differentiated from the stable time-changed Brownian motion described in Huillet.³⁰

Let us recall that the path properties of an lfsM strongly depend on the interplay between the parameters H and α . When $H > 1/\alpha$, the lfsM paths are Hölder continuous on compact intervals of any order smaller than $H - 1/\alpha$. If $H < 1/\alpha$, the lfsM explodes at jump times of the driving Lévy process; in particular, it has unbounded paths on compact intervals. Clearly, in this approach, the stable character of the resulting process has been favored, and this model is the natural extension of the fBm in this respect.

¹ We say that a stochastic process $W = \{W(h), h \in H\}$ defined in a complete probability space (Ω, \mathcal{F}, P) is an isonormal Gaussian process (or a Gaussian process on H) if W is a centered Gaussian family of random variables such that $E(W(h)W(g)) = \langle h, g \rangle_H$ for all $h, g \in H$.

2.2 | The neural network structure

There are three kinds of invariances that we might require from the network: shift, scale, and drift invariance. In order to make an fBm Hurst-estimator that works well in practice, we want to rely on all three of the above invariances. We can obtain shift invariance by transforming the input sequence to the sequence of its increments. Differentiating the input also turns drift invariance to shift invariance. By performing a standardization on the sequence of increments, we can ensure drift and scale invariance. The standardizing phase can also be considered as an additional layer to the network, applying the transformation $x \mapsto (x - \bar{x})/\hat{\sigma}(x)$ to each sequence of increments x in the batch separately, where $\hat{\sigma}(x)$ is the sample standard deviation over the sequence x , and \bar{x} is the arithmetic mean over x . Note that the sample standard deviation is a biased estimator of the true one because of the autocorrelated samples. The bias, however, is negligible because of the relatively long sample length.

In order to create an fBm Hurst estimator that is effective in practical applications, it is crucial to ensure the model exhibits three kinds of invariances: shift, scale, and drift invariance. Achieving shift invariance can be done by converting the input sequence into a sequence of its increments, which also transforms drift invariance into shift invariance when differentiating the input.

To address both drift and scale invariances, we perform a standardization on the sequence of increments. This is where our approach diverges from the conventional pre-processing methods. Given the nature of our model, which generates new data in every training epoch, we integrate the standardization process as an intrinsic part of the network. This involves applying the transformation $x \mapsto (x - \bar{x})/\hat{\sigma}(x)$ directly to each sequence of increments x within the batch. Here, $\hat{\sigma}(x)$ denotes the sample standard deviation across the sequence x , and \bar{x} represents the arithmetic mean of x .

Incorporating this standardization step as an additional layer within the network architecture allows for dynamic adjustment to the newly generated data's properties, ensuring that each batch is standardized based on its current statistics. This method not only maintains the required invariances for the fBm Hurst estimator but also enhances the model's adaptability and performance across varying data distributions. It is important to note, though, that while the sample standard deviation is a biased estimator due to the autocorrelated nature of the samples, this bias is negligible owing to the sufficiently long sample length, ensuring the effectiveness of our standardization approach.

By embedding the standardization within the network, we offer a robust solution to maintain shift, scale, and drift invariance in the face of dynamically generated training data, thereby improving the reliability and accuracy of the fBm Hurst estimator in practical scenarios.

After the layers to ensure invariances, the next layers constitute a sequential regressor. This part of the network first transforms the input sequence into a higher dimensional sequence, after which each dimension of the output is averaged out, resulting in a vector. Finally, a scalar output is obtained by an MLP. In the present analysis, we only consider an LSTM network.³¹ We found that the specific hyperparameter configuration does not have a significant effect on its performance. The following are the hyperparameters that we used in the experiments below.

The applied network consists of an unidirectional LSTM with two layers; its input dimension is 1, and the dimension of its inner representation is 128. In both models, we use an MLP of three layers (output dimensions of 128, 64, and 1), with a parametric rectified linear unit (PReLU) activation function between the first two layers. Adaptive moment estimation with weight decay (AdamW) optimization on the mean squared error (MSE) loss function was used for training the models.³²

The choice of the LSTM architecture was primarily influenced by its prevalent use in current applications. Our main objective was to critically evaluate its effectiveness in managing time-series data by exploring its operational reliability under various conditions.

Our experimental analysis led to the selection of rectified linear unit (ReLU) over other activation functions like Gaussian error linear unit (GELU) and exponential linear unit (ELU) due to its superior performance in the MLP layers. The experimentation also revealed that incorporating dropout layers negatively impacted model performance, leading to their exclusion from the final design.

We initially tested multiple hidden layers but settled on a two-layer LSTM structure. Adding a third layer provided marginal improvements, which did not justify the additional computational resources required.

Our trials with varying batch sizes and series lengths showed these factors had minimal impact on model performance. Similarly, adjustments to the alpha parameter mainly influenced training speed rather than outcome quality, while high values of the mu parameter detrimentally affected learning, suggesting issues with model fitting.

Among the different optimizers tested, stochastic gradient descent (SGD) was outperformed by ADAMW. Further tuning of ADAMW's parameters like learning rate and weight decay proved unbeneficial. The model consistently achieved optimal loss values by around 35 epochs, with no substantial gains observed up to 50 epochs.

In the MLP layers, no significant advantages were noted when using a pyramid-like configuration of neurons compared to uniform layers, leading to a simplified and more resource-efficient structure.

In sum, the architecture and parameters of our network were methodically determined through extensive testing to ensure optimal performance across various scenarios.

2.3 | The training procedure

When faced with limited real data, the use of synthetic data generators enables the training of neural networks on virtually unlimited datasets. In this approach, the loss calculated on new data batches serves as the validation loss, as each batch consists entirely of new, synthetically generated data. This method helps to prevent overfitting by ensuring the model does not specialize too heavily on the training data and also highlights the critical importance of the quality of the synthetic data generator.

In the literature, there are generally two types of generation procedures discussed: Exact methods and Approximate methods. During the data generation phase, we tested the method by Davies and Harte, as well as Cholesky from the exact methods category. Cholesky was quickly discarded due to its computational complexity of $O(n^3)$, which made it increasingly slower as we trained the model on longer sequences.

A **large and high-quality simulated sample** from both fBm and fOU processes is applied for effective training of the neural network. We utilized Kroese's method,³³ a variation of the Davies–Harte procedure, to generate fBm trajectories. The Hurst parameter for these trajectories is set randomly, according to the uniform distribution in the [0,1] interval.

For generating the fOU process, we used the Euler scheme, where the driving noise was derived from fractional Gaussian noise, since we cannot directly produce fOU using the Davies and Harte method. Additionally, for fractional Lévy stable motion, we used both the `rlfsm` package available in R and another [implementation](#) available in Python.

To ensure the quality of the simulation, we assess Gaussianity, compare the empirical autocovariance to the theoretical model, and re-estimate the parameters using classical statistical estimators, such as the R/S, Whittle, variogram, Higuchi, and Peng's detrended fluctuation analysis.

In addressing the issue of stationarity for the simulated sequences used in our study, specific measures were taken to ensure that the sequences exhibit the required statistical properties, especially for fOU and fBm processes.

For the fOU process, we initiated the simulation from zero and discarded the first 100 values. This burn-in period is considered sufficient to achieve stationarity in the generated sequence, thereby mitigating the transient effects from the initial condition. This approach is based on the characteristic of the fOU process where the increments tend to stabilize as the process evolves.

In the case of fBm, the focus is on the increments of the process, which are inherently stationary. This is a fundamental property of fBm, as the increments over equal time intervals are statistically identical, a characteristic that is crucial for our applications involving modeling long-range dependent data.

It is important to note that while statistical tests are employed to verify the empirical properties of these distributions, the theoretical underpinnings of the generation methods for both fBm and fOU have been rigorously proven.^{34,35} Theoretical proofs provide a solid foundation for the reliability of these methods in generating data that accurately reflects the intended stochastic properties.

Therefore, our approach integrates robust theoretical validation with practical statistical tests to ensure that the data generated for our simulations adheres to the required criteria of stationarity and other statistical properties. This strategy is employed specifically to verify that no inconsistencies or errors arise during implementation that would conflict with the theoretical foundations. By ensuring this compatibility, we can confidently utilize these simulations for further analysis and applications in modeling and forecasting tasks.

While the original Davies–Harte method for generation is accessible in existing Python packages, our application of Kroese's method necessitates a custom **implementation** using efficient Python framework tools. Our implementation strategy preserves the covariance structure, saving it to avoid redundant computations, thereby significantly enhancing computational speed. As the lfsm process does not belong to the class of isonormal processes, the generation has to rely on different principles. In this case, we were unsuccessful in developing a similarly efficient generator; therefore, we cannot train the network on lfsm trajectories. Instead, we use the package `rlfsm` in R to generate lfsm sample paths. Simulation

of the sample paths is done via Riemann-sum approximations of its symmetric α -stable stochastic integral representation while Riemann sums are computed efficiently using the Fast Fourier Transform algorithm. However, it is much slower, so creating a proper training set is not feasible this way. Therefore, we had to contend with analyzing the estimation of the Hurst parameter of fBm processes by the LSTM network trained on fBm or fOU processes.

Speed is a critical factor in this context, as we employ up to 10^7 trajectories of the length of 1600 or 6400 for training the network, meaning 16–64 billion data.

The neural network training is carefully designed to accommodate this complex setup. The model utilizes a learning rate of 0.0001 with the AdamW optimizer, a combination chosen for its precision in model tuning and effective convergence. In our research, we conducted multiple training sessions and evaluations using different loss functions to investigate their individual impacts on the model's performance. Each loss function was applied in its own separate training context, without combining them or using them simultaneously within a single session. For loss functions, both L1Loss (Mean Absolute Error [MAE]) and MSE are employed, crucial in refining the model's prediction accuracy by appropriately weighting different types of errors. The training spans various epoch lengths – 25, 50, and 100 epochs – with each epoch generating 100,000 sequences. This strategy allows the model to learn at multiple depths and assess the impact of different training durations. Moreover, the batch sizes are set at 32 for training and 128 for validation, ensuring efficient learning with smaller batches during training, and a broader assessment of data during validation. We execute several trainings with trajectory lengths of 400, 1600, and 6400 and analyze the obtained estimator.

The overall success of this method depends significantly on the generator's accuracy and reliability. If the generator fails to closely approximate the desired distribution, there's a risk that the model might learn the generator's errors, leading to distorted results.

3 | ANALYSIS OF THE HURST PARAMETER ESTIMATION

3.1 | Evaluation of the LSTM estimator on fBm and fOU samples

For benchmarking, we choose the estimator obtained by training the network on fBm trajectories of length 1600 with a training span of 100 epochs. Increasing the number of epochs or the length of the training series does not result in much decrease in the loss function; hence, the choice. We choose the MSE and the MAE for the loss function. The root mean squared error (RMSE) is minimized by the conditional mean, making it most useful when large errors are particularly undesirable. Given the constraint on the estimated values, large errors are of less concern in this case. The MAE is a linear score, which means that all the individual differences are weighted equally in the average. Minimizing MAE will make the fit closer to the median and eventually more biased.

The trained network is then used to estimate the Hurst parameter of 10,000 fBm and fOU trajectories of length 400, 1600, and 6400, respectively. We evaluate the estimation in terms of the RMSE, the MAE, and the mean relative errors (MRE). The MAE and the RMSE can be used together to diagnose the variation in the errors in a set of predictions. The RMSE will always be larger or equal to the MAE; the greater the difference between them, the greater the variance in the individual errors in the sample. The Hurst parameter is an exponent; therefore, the absolute or the relative deviation will heavily affect the modeling accuracy, necessitating their analysis. It is not just the overall variability of the errors that reflects the loss or risk in the model. Rarely occurring severe errors can be very much intolerable in some applications. Therefore, beyond the mean error values, we also calculate the 95% quantiles and the medians of the absolute and relative errors together with the maximum of the absolute error. The reason is that in real applications, we estimate the Hurst exponent trajectory-wise, and even if the estimation's significant inaccuracy occurs only in a small percentage of the observations (i.e., stock prices, degradation of sensitive machine parts, etc.), it still can cause unbearable loss or risk.

In the first instance, we analyze the evaluation of the estimator on samples of the same length, that is, of 1600, as the training was conducted. The visual representation of the results is displayed in Figure 1.

The upper right panel shows the estimated values versus those set for generating the trajectories. The overall performance of deep learning seems quite appealing in the graph. While the Hurst exponent values were chosen uniformly in the generation, the estimated values slightly deviate from that, as seen in the upper mid-panel histogram. The deviation indicates the presence of a bias at large – close to one – Hurst parameter values. With a few exceptions, the raw errors seem to follow normal law, as illustrated by the normal plot in the upper right corner.

Turning to absolute errors, its growth with the true Hurst value, as the lower left plot shows, is noteworthy. The histogram of the absolute errors in the middle of the lower row shows the skewed distribution close to that of an absolute

LSTM Estimator, trained on fBm 1600, evaluated on fBm1600

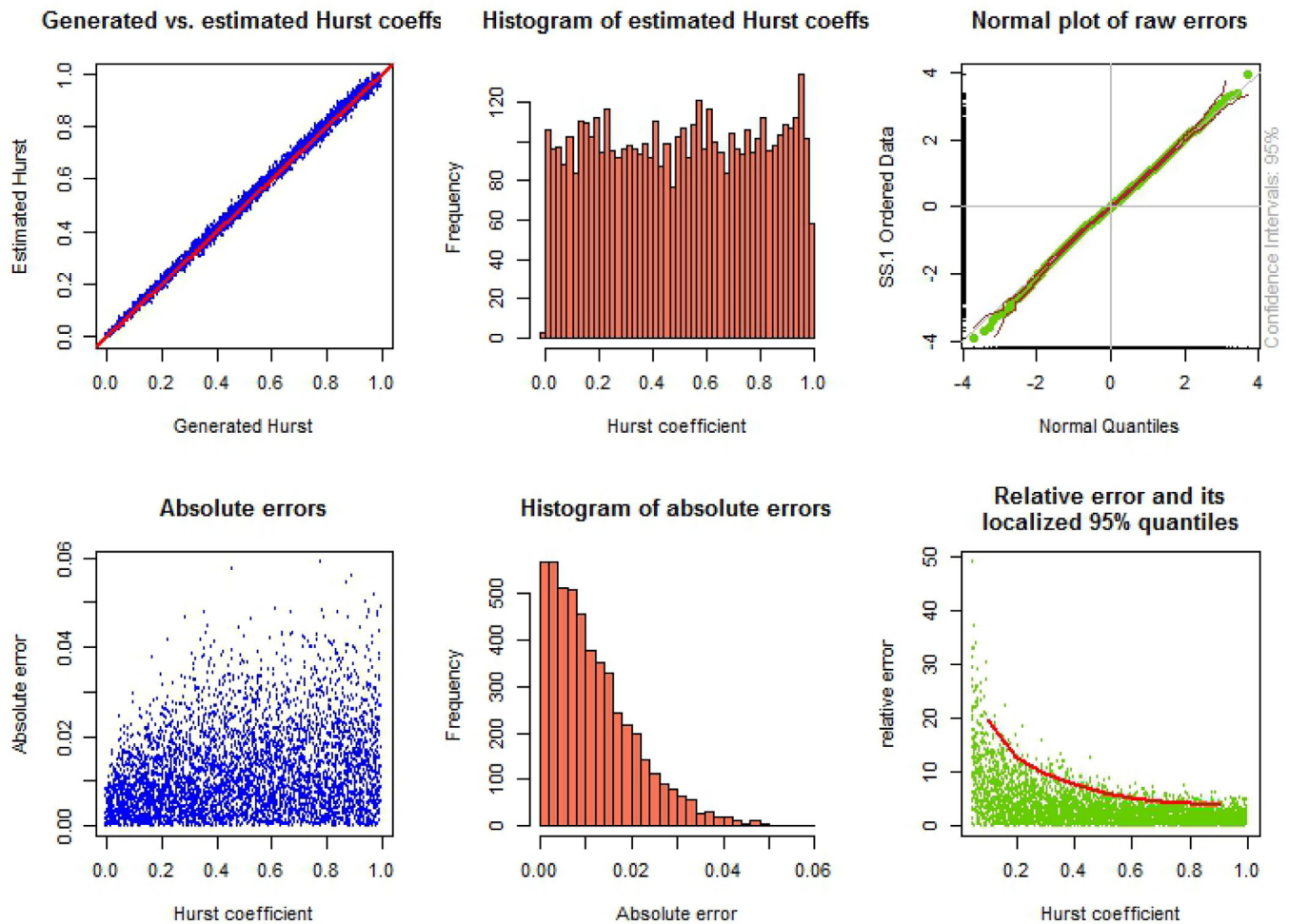


FIGURE 1 LSTM estimation and its errors. The network is trained and subsequently validated equally on 1600-length fractional Brownian motion samples. LSTM, long short-term memory.

value of a Gaussian variable. Remark, however, that two-sided deviations from the normal law add up, and the skewness of the absolute errors is 1.16, while the absolute value of a Gaussian variable has a skewness of 1. The maximum absolute error value is almost 0.06, more than four times the raw error's standard deviation (0.0147). That is regarded as extreme under the Gaussian law. It hints at a heavier-than-normal tail of the error distribution. The lower right plot exhibits the relative error. Understandably, when the “true” generated Hurst value is extremely low, it induces high relative error even from a small absolute error. For this reason, we cut the graph and do not display the relative error when the generated Hurst value is lower than 0.02; such low values are not plausible in real applications anyway. As can be seen, the relative error depends heavily on the true value of the Hurst parameter. For that reason, together with the overall 95% quantiles, we also calculate localized values. To a given generated Hurst value H , we take the $H - 0.1, H + 0.1$ interval around it and calculate the relative error quantiles from generated trajectories with true Hurst value from that interval. We present these quantiles, calculated in 0.1 steps, by the red line in the lower right plot.

We may change the loss function in the training from MSE to MAE, pushing the optimization to the conditional median. Not knowing what feature of the samples LSTM learns, it is difficult to understand the difference between the two optimizations. However, the difference in errors is really minuscule: RMSE: 0.014774 versus 0.014866, MAE: 0.011582 versus 0.011659 MRE% 3.694370 versus 3.631601 the first two to the advantage of the MAE, the last one to the MSE. Neither of the loss functions can be preferred, and the result indicates that well-balanced samples are generated for training.

For comparison, we present a similar analysis with the Higuchi estimator, noting that very similar results – with alternate asymmetries though – can be obtained by the detrended fluctuation analysis and the Whittle estimators, and results are significantly worse for the variogram-based estimator, not to mention the classical R/S estimator. We evaluate the

Higuchi Estimator, Evaluated on fBm of length 1600

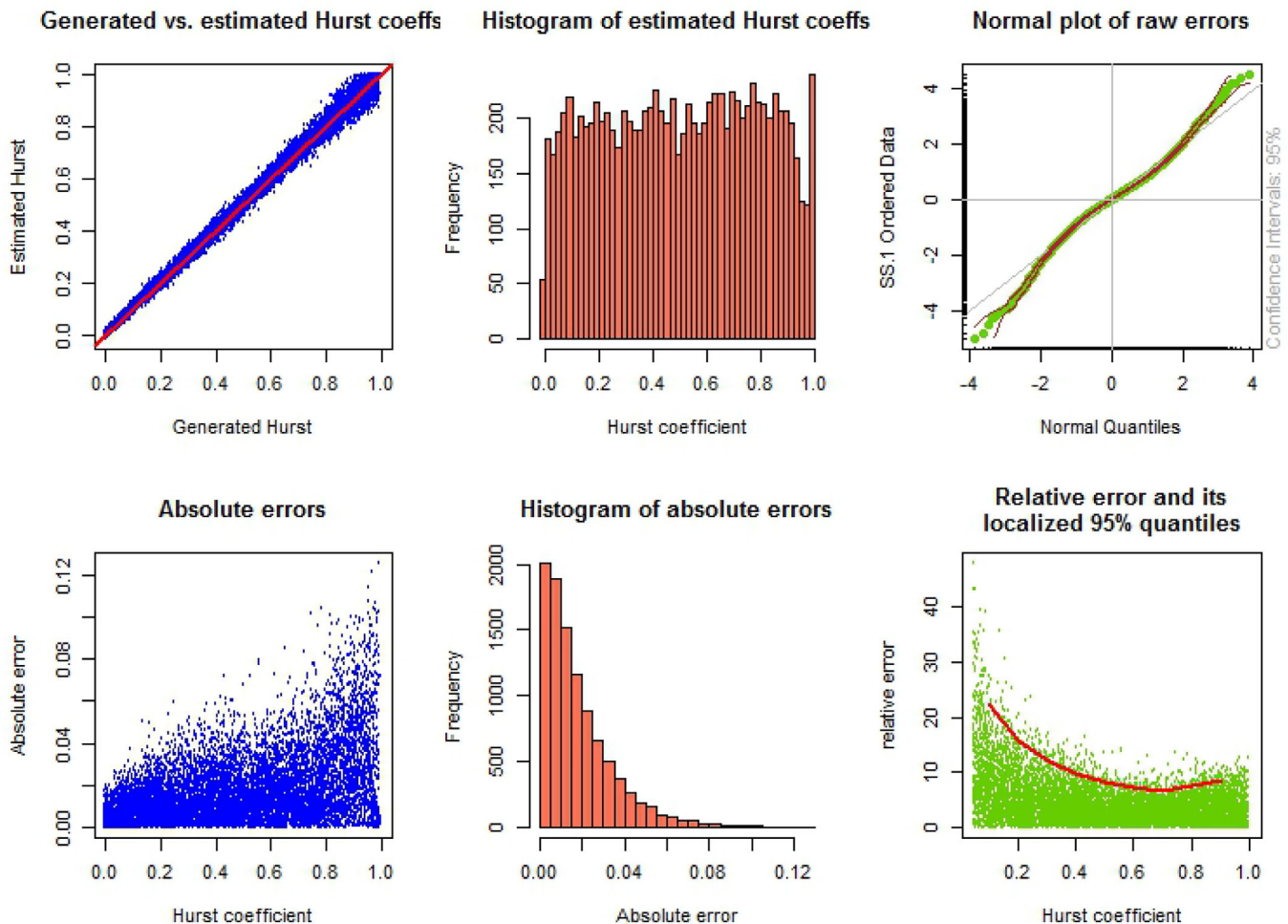


FIGURE 2 Higuchi estimation and its errors. The estimator was tested on 1600-length fractional Brownian motion samples.

Higuchi estimator on 10,000 simulated fBm trajectories of length 1600 with uniformly distributed random Hurst parameters. The plot of estimated versus “true” generated values is shown in the upper left corner of Figure 2. Bias is now observable for the smallest (close to 0) and the largest (close to 1) values, as the histogram in the middle of the upper row discloses. According to the normal plot in the upper right corner, the estimator does not have a normal distribution; it has heavier than normal tails.

The absolute errors slightly exceed that of the LSTM estimator and have heavier tails; this is how the first lower left corner plot and the histogram next to it show. The relative errors are not much higher than the ones for the LSTM estimator, though. That can be observed in the lower right plot. The quantiles are also similar in the low end of the Hurst parameters; however, they grow higher than that of the LSTM at high Hurst values.

Evaluating on fOU processes the LSTM estimator trained on fBm, we proceed similarly to the fBm case. The results are also very similar to the fBm case concerning both the LSTM and the Higuchi estimators. Therefore, we do not present a similar figure and detailed assessment separately; it would simply repeat what has been presented so far. In the first two blocks of rows in Table 1, we exhibit the performance indicators of the LSTM network trained on fBm samples of length 1600 and evaluated on 10,000 shorter (length 400), equally long, and longer (length 6400) samples of both fBm and fOU. The accuracy grows significantly by increasing the sample length; the errors almost halve when quadrupling the sequence length.

We also evaluate the LSTM on fOU processes when trained on 1600-long fOU processes. The generation of fOU processes is slower than fBm-s; hence, we are forced to contend with shorter epochs in training, consisting of 10,000 trajectories only. The third block of rows in Table 1 reveals that the network’s performance is almost as good as the one trained on fBm despite the shorter training on fOU. Compared to the second block of rows, we may notice that the evaluation is

TABLE 1 Performance metrics for LSTM and Higuchi Estimator evaluated on fractional Brownian motions and fractional Ornstein–Uhlenbeck processes.

Type	Evaluation length	RMSE	MAE	MRE (%)	Absolute error			Relative error (%)	
					Max.	q95%	q50%	q95%	q50%
LSTM trained on fBm 1600 and evaluated on fBm	400	0.0311	0.0241	7.32	0.1482	0.0630	0.0194	24.45	4.66
	1600	0.0149	0.0117	3.63	0.0592	0.0300	0.0094	11.73	2.21
	6400	0.0079	0.0066	2.43	0.0389	0.0165	0.0056	9.80	1.24
LSTM trained on fBm 1600 evaluated on fOU	400	0.0307	0.0238	7.06	0.1279	0.0620	0.0190	22.59	4.56
	1600	0.0156	0.0122	3.69	0.0710	0.0316	0.0097	11.95	2.34
	6400	0.0079	0.0062	2.02	0.0293	0.0157	0.0051	6.54	1.20
LSTM trained on fOU 1600 evaluated on fOU	400	0.0295	0.0229	7.02	0.1270	0.0599	0.0184	22.53	4.33
	1600	0.0157	0.0122	3.62	0.0708	0.0321	0.0098	11.51	2.31
	6400	0.0107	0.0085	2.28	0.1169	0.0233	0.0063	6.65	1.60
Higuchi Estimator on fBm samples	400	0.0445	0.0342	10.5	0.2507	0.0903	0.0268	33.69	6.57
	1600	0.0246	0.0182	4.93	0.1257	0.0515	0.0136	14.34	3.40
	6400	0.0152	0.0104	2.61	0.0895	0.0332	0.0071	7.51	1.84

Note: The LSTM network was trained on 1600-long fractional Brownian motion samples.

Abbreviations: fBm, fractional Brownian motion; fOU, fractional Ornstein–Uhlenbeck; LSTM, long short-term memory; MAE, mean absolute error; MRE, mean relative error; RMSE, root mean squared error.

slightly better on shorter (400) samples, but on longer (6400) samples, its performance does not improve as much as the fBm-trained network. When the training and the evaluating sample lengths are equal, the two training types perform almost equally well. The higher maximum and quantiles of absolute errors in the 6400-long evaluation are due to several outliers. We guess that those outliers are the result of the shorter training. The skewness of the absolute errors is 2.3558, highly different from 1, the skewness of the absolute value of a normal distribution. It is unprecedentedly high, considering all of our experiments.

In summary, the LSTM estimator of the Hurst coefficients trained on fBm samples performs better than the traditional statistical estimators in every aspect. True, the LSTM halves the root mean squared errors compared to statistical estimators; however, it cuts the MAE to 60%–70% and the MRE to 75%–90% only, depending on sequence length. LSTM also performs well on the maxima and the quantiles of the absolute errors; it almost halves those. However, RMSE and MAE are not sensitive to which range of the original parameter the errors occur. That points out the relevance of the relative errors, and in that terms, the performance of LSTM is not that excellent. In particular, when the 95% quantiles of localized relative errors are considered, in the low range of the Hurst parameter, LSTM, and the statistical estimators are almost equally bad, nearing or even exceeding 20%. On the high end of the parameter space, nearing one, however, LSTM again significantly outperforms the Higuchi and other statistical estimators, as the latter's relative errors start to grow even along the Hurst values.

3.2 | Evaluation on linear fractional α -stable motions

In our research on RUL estimations, specifically focusing on lithium-ion battery degradation, we have delved into the nuances of the two predominant modeling practices in this field. The first widely adopted approach involves the application of fBm and fOU processes. These stochastic models, frequently discussed in the literature for their effectiveness in modeling time-dependent deterioration, include significant contributions detailed in references such as Refs. [4, 36], and [37].

The second approach employs linear fractional stable Lévy motion (lfsM), which has been increasingly recognized for its capability to accurately represent non-Gaussian distributions and capture heavy-tailed phenomena and extreme events. Such characteristics are especially critical in modeling lithium-ion battery degradation, where unpredictable operating conditions and outlier events can drastically affect the battery's performance and lifespan. Essential studies on lfsM, such as those by [28] and [1], offer a comprehensive framework for understanding the complex dynamics typical of real-world battery degradation scenarios.

The fBm-trained LSTM and the Higuchi Estimators, evaluated on lfsm of length 1600 with alphas=1.8, 1.5, 1.2

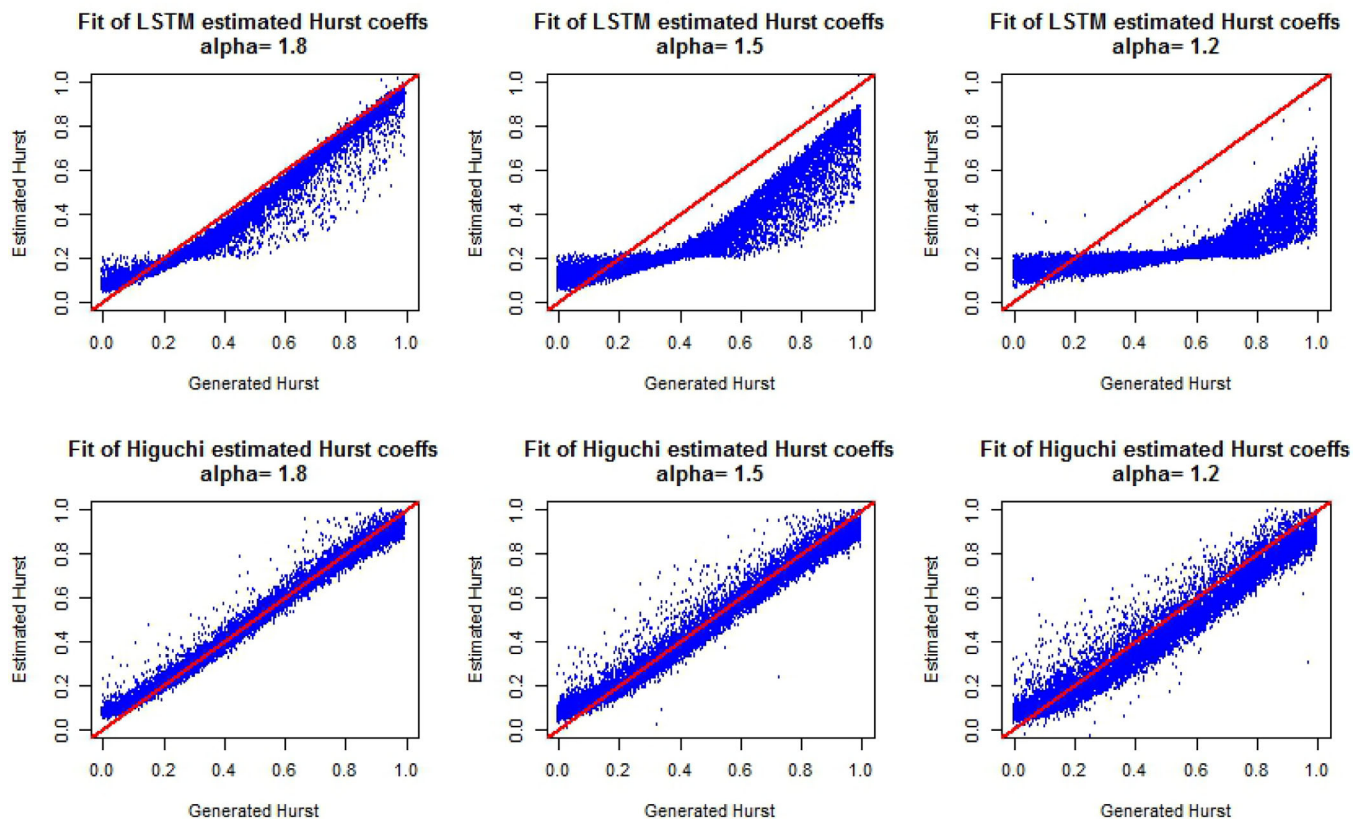


FIGURE 3 Hurst exponent estimation of linear fractional stable motions of $\alpha = 1.8, 1.5, 1.2$ by LSTM and Higuchi methods and its errors. The network is trained on 1600-length fractional Brownian motion samples. The evaluation length is also 1600. LSTM, long short-term memory.

In view of the explicit solution formula of the Langevin equation, the fOU process can be represented as a time-changed and scaled fBm, so it is not very surprising that the network trained on fBm samples performs well on fOU samples, too. Therefore, it is quite exciting to overstep this class and analyze the performance of LSTM on lfsm process samples.

As it has been said before, we cannot train the network on lfsm samples due to the lack of a fast generator at the moment. This is why we use the network trained on fBm and fOU samples. For the evaluatory analysis, we chose our benchmark, the LSTM network taught on 1600-long fBm samples in the span of 100 epochs. The evaluatory samples are generated by the `r1fst` package of R. We choose α from the range (1,2) with the values 1.8, 1.5, and 1.2. With these parameters, the lfsm features LRD. As Figure 3 demonstrates, the LSTM network cannot properly detect the true Hurst parameter value, and the higher the index of stability, the more LSTM underestimates it. To the contrary, the Higuchi method estimates the Hurst exponent correctly, albeit with higher variance than in the case of fBm or fOU.

Even though the result concerning LSTM is negative, it is still valuable, as it points out the limitations of the described setup and training when evaluating the fBm-trained network on lfsm samples.

3.3 | The effects of training cycles and training sequence length

In what follows, we consider the estimation quality depending on the training and evaluation length of the series. To better understand the sample-length dependence on the quality of learning, we refined the resolution. While keeping the methodology unchanged, we included further series of lengths 800 and 3200 in the training and 100, 200, 800, and 3200 long series in the evaluation. As it is partially indicative of the other error types, we restrict the evaluation to the RMSE only. The resulting errors are displayed in Table 2.

TABLE 2 Root MSE losses of LSTM-based models trained on different sequence lengths.

Training seq. length	Root MSE losses by evaluation seq. length						
	100	200	400	800	1600	3200	6400
400	0.0691	0.0449	0.0307	0.0218	0.0168	0.0140	0.0127
800	0.0693	0.0447	0.0302	0.0210	0.0152	0.0116	0.0094
1600	0.0708	0.0459	0.0311	0.0211	0.0149	0.0106	0.0079
3200	0.0738	0.0472	0.0312	0.0213	0.0149	0.0105	0.0080
6400	0.0748	0.0480	0.0318	0.0217	0.0151	0.0110	0.0083

Abbreviation: MSE, mean squared error.

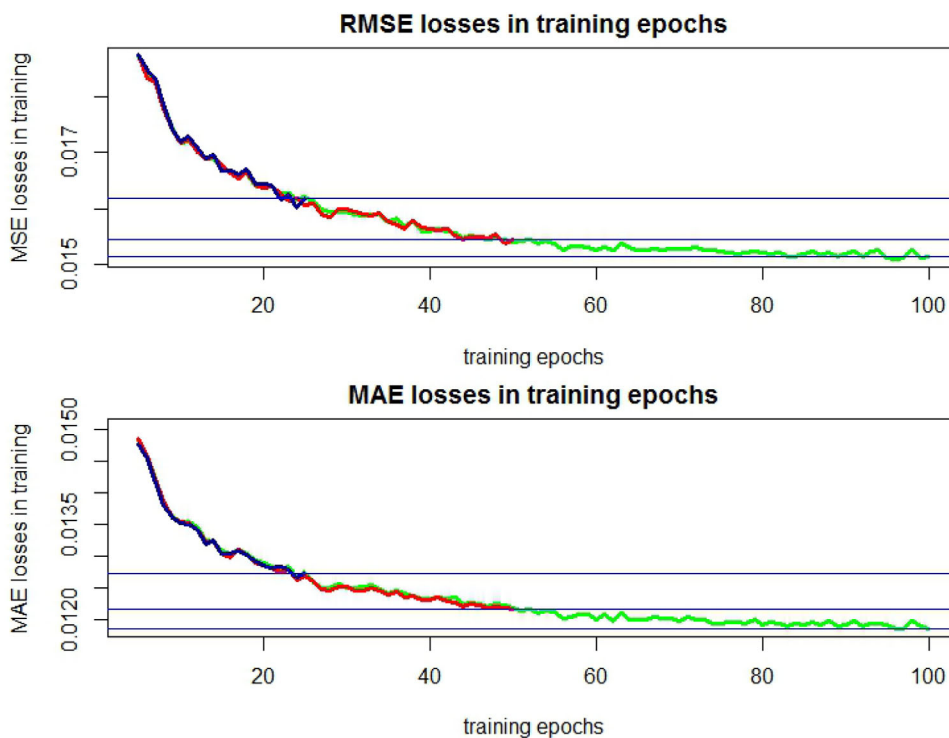


FIGURE 4 The changing of RMSE and MAE losses as training progresses. 25 epoch long training = blue, 50 epoch long training = coral, 100 epoch long training = green. MAE, mean absolute error; RMSE, root mean squared error.

Understandably, the performance is very weak when evaluated on the short series. Generally speaking, when *trained* on shorter sequences, the performance of LSTM improves when *tested* on longer sequences but is slower than LSTM variants *trained* on longer sequences. LSTM variants trained on longer sequences still performed well on shorter sequences but not as well as dedicated variants. The differences in these cases are marginal, except for the very short series. The LSTM trained on 1600-long samples has the best performance, hardly distinguishable from the LSTM of 3200-long samples. However, training time is considerably less for the 1600-long, which is why we have chosen that setup as the reference. Training on 6400-long sequences does not improve the RMSE loss; the network probably reached its limit of learning, given the current architecture.

We analyzed the 1600-long fBm trajectory training, studying the loss dependence on training length in terms of the spanned epochs. Figure 4 exhibits the loss changes during two series of 100, 50, and 25-long training, with MSE and MAE loss function choices, respectively. Although MSE is the loss function in the algorithm, we display the RMSE instead of the MSE in the figure and use it in the comparisons because of the very small MSE values. The graphs start from the 6th epoch. In the first 5 epochs, the loss is still so high that it would suppress the visibility of the changes in the remaining training epochs; hence, we omitted to display it. It is remarkable that the losses closely follow each other in independently started network teachings. Shorter teaching losses simply follow the beginning part of longer teaching. By the 50th training, the RMSE decreases to 95.1% that of the 25th training, and by the 100th training, it decreases significantly less, to 98.6% that of

TABLE 3 Detailed data generation times for various processes and data sizes.

Process/Data size	Time for 10,000 sequences (length 1600)	Time for large data sets
fBm in Python	6–10 s	N/A
fOU in Python	18–22 s	N/A
lfsm in Python	80–85 min	250k: 8–9 min
lfsm in R	20–25 s	1250k: 43–45 min
Pre-generated data from R (lfsm)	250k	8–9 min
	1250k	43–45 min
	2500k	86–90 min

TABLE 4 Validation errors of LSTM when training it on mixed datasets containing fractional stable motion (lfsm) in 0 (i.e., no mixing), 0.1, and 0.5 proportions.

	No lfsm mixed		Proportion lfsm 0.1		Proportion lfsm 0.5	
	fBm	fOU	fBm	fOU	fBm	fOU
RMSE	0.0149	0.0156	0.0325	0.0337	0.4087	0.4816
MAE	0.0117	0.0122	0.0244	0.0245	0.221	0.2552

Note: The evaluation was performed on fractional Brownian motions and fractional Ornstein–Uhlenbeck processes.

Abbreviations: fBm, fractional Brownian motion; fOU, fractional Ornstein–Uhlenbeck; MAE, mean absolute error; RMSE, root mean squared error.

the 50th training. At the 75th step, the RMSE is only 0.3% higher than the final RMSE loss. The same concerns the MAE losses in the respective teachings. The actual MAE loss percentages are: 50th/25th = 95.5, 100th/50th = 97.5%. At the 80th step, the MAE is only 0.5% higher than the final MAE loss. So, we may say that when using 1600-long fBm samples, there is no practical indication to increase the teaching epoch numbers above the 80th. Nevertheless, we used the 100 epochs in our analysis as the longest (and benchmark) teaching.

3.4 | Training and evaluation on mixed data

In the context of our experiments involving the training of neural networks on time-series data generated from fBm, fOU, and lfsm processes, we encountered significant challenges when including lfsm in the training datasets. Despite its potential theoretical advantages, incorporating lfsm data did not yield the expected performance improvements. On the contrary, it led to a degradation in model performance on fBm and fOU data.

Our training approach involved generating a substantial test set of 10,000 instances for each process, employing random Hurst parameters. This was facilitated by the PyTorch framework, which allowed efficient data handling within Python. Table 3 below illustrates the time required to generate this test data:

The significantly longer generation times for lfsm, especially in Python, highlight the computational challenge. Training durations extended markedly when lfsm data constituted a larger portion of the dataset. Specifically, training for 25 epochs on a dataset with 10% lfsm content required around 23 hours, and with 50% lfsm, it extended to 48 h.

We even considered the rlfsm package in R for its faster data generation capabilities, conducting model fittings with datasets containing varying proportions of lfsm to evaluate if improvements could be achieved under different conditions.

These extensive training periods were not justified by the results. Although there is an improvement in the Hurst estimation for lfsm data, in particular for small stability parameters, it is still far from the accuracy of the Higuchi estimator. Figure 5 illustrates these outcomes.

On the other hand, the accuracy of estimations for fBm and fOU processes worsened, thus leading to overall diminished model efficacy. It is demonstrated in Table 4.

These findings strongly suggest that while lfsm may theoretically enrich a mixed training dataset, in practice, it complicates training without significantly improving model accuracy, thereby necessitating further investigation into more suitable architectural adjustments or alternative strategies. Additionally, issues with data conversion significantly slowed down the training process, further contributing to its inefficiency and ineffectiveness. This highlights the need for optimized data handling methods to improve training dynamics and potentially enhance model performance under varying conditions.

LSTM Estimator, trained on a mixture of lfsm and fBm, evaluated on lfsm of length 1600
Generated vs. estimated Hurst coeffs

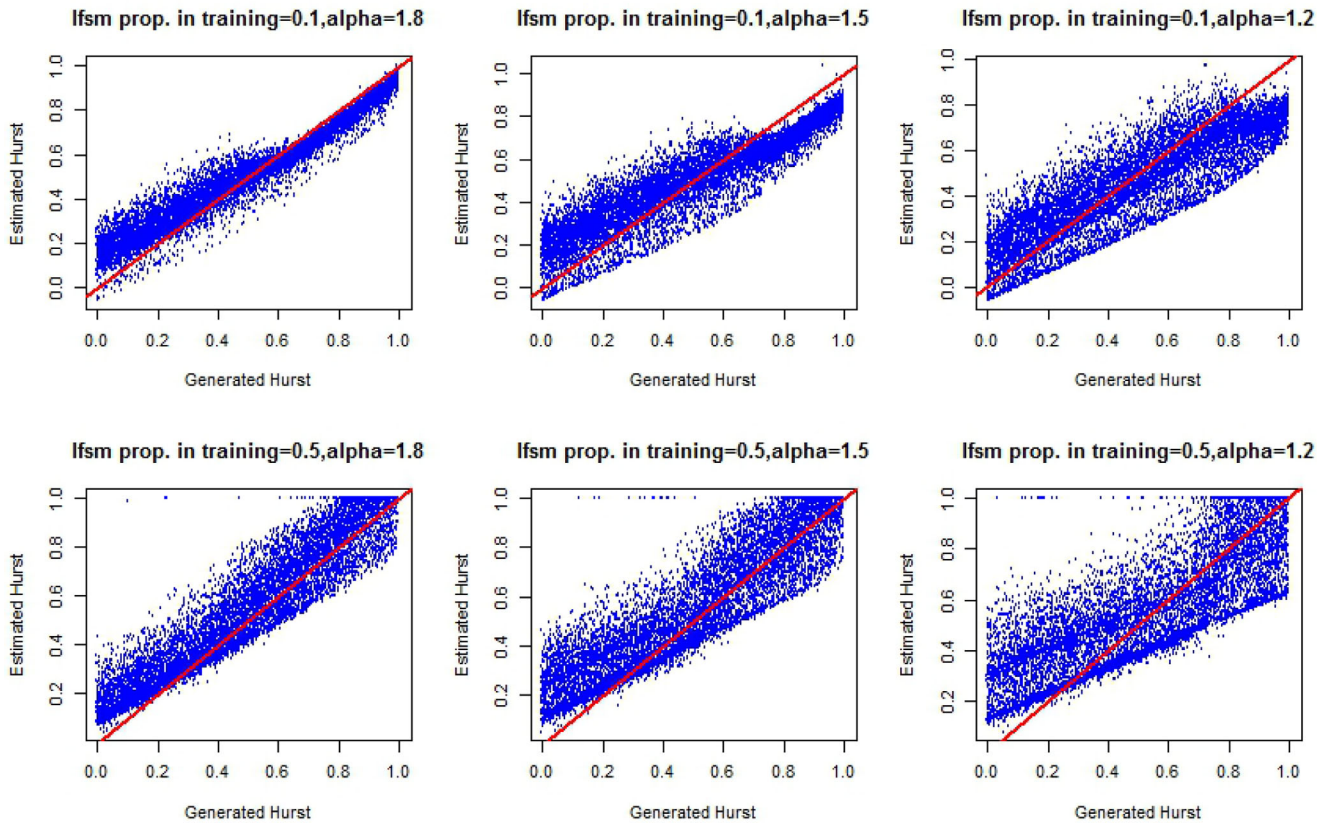


FIGURE 5 Generated versus estimated Hurst coefficients when the LSTM network is trained on a mixture of linear fractional stable motion (lfsm) and fractional Brownian motion processes. The proportion of lfsm processes in the training set is either 0.1 or 0.5. LSTM, LSTM, long short-term memory.

4 | AN APPLICATION TO LI-ION BATTERY DEGRADATION

In our study, we utilized a dataset provided by the NASA Prognostics Center of Excellence, which includes experiments on li-ion batteries involving charging and discharging at different temperatures and recording the impedance as the damage criterion. The dataset is available at <https://www.nasa.gov/intelligent-systems-division/discovery-and-systems-health/pcoe/pcoe-data-set-repository/>.

We determined the capacity associated with the cycles, applying a 70% threshold as the criterion for battery degradation. We present the obtained series in Figure 6. Our objective was to estimate the Hurst parameter of the process, assuming it follows fBm, based on the obtained data series.

We use the madogram to estimate the fractal dimension D , returning a Hurst exponent estimate by $2 - D$, and the Higuchi method for statistical benchmarking. The madogram yielded a value of 0.8558, while the Higuchi method resulted in 0.6189.

In contrast, among our trained LSTM models, the best results from models trained on fBm samples were obtained from the 100- and 200-long training length, which estimated Hurst parameter values of 0.9312 and 0.9641, respectively. In Subsection 3.3, we concluded that the best estimate can be achieved when the network is taught on the same length series as the sample, for which to estimate the Hurst parameter. It may explain why the short training lengths provide the best results in the sense of being closest to the benchmark. The closest result to the madogram estimation was obtained from a model trained on a 200-length fOU process, with a value of 0.8294, and the 400-length fOU-trained network provided 0.7726 as the estimated Hurst parameter. The advantage of our approach is that we can also come up with confidence intervals in terms of absolute deviation and relative error. For the 100-long fBm training, the 95% quantile of the absolute errors is 0.1280, whereas for the 200-long fBm training, the same 95% quantile is 0.0875. Considering that the Hurst

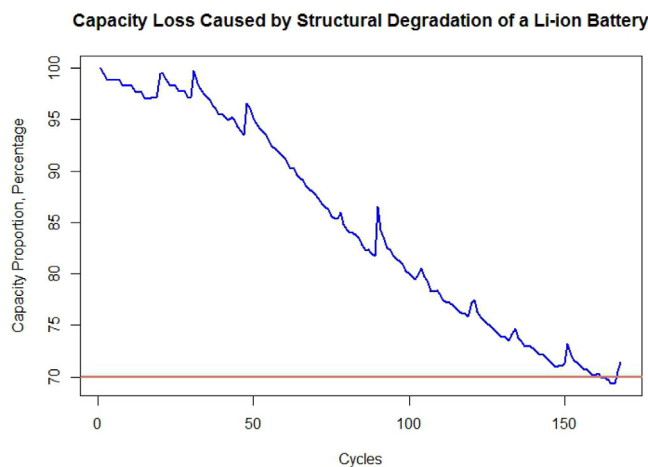


FIGURE 6 Capacity losses caused by structural degradation of li-ion batteries.

TABLE 5 Summary of Hurst parameter estimations and confidence intervals.

Method	Hurst parameter estimate	95% Confidence interval
Madogram	0.8558	N/A
Higuchi	0.6189	N/A
LSTM (100-long fBm)	0.9312	(0.8032, 1)
LSTM (200-long fBm)	0.9641	(0.8361, 1)
LSTM (200-long fOU)	0.8294	(0.744, 0.9148)
LSTM (400-long fOU)	0.7726	(0.7114, 0.8338), (0.7047, 0.8404)
Shao and Si Model	0.8 - 0.9	N/A
Liu, Song, and Zio Model (lfsm)	~0.6	N/A

Abbreviations: fBm, fractional Brownian motion; fOU, fractional Ornstein–Uhlenbeck; LSTM, long short-term memory.

parameter is less than one, these quantiles produce 95% confidence intervals (0.8032,1) and (0.8361,1) for the two estimations, respectively, both containing the madogram-based estimation. Turning to the fOU-trained LSTM estimates, the absolute error's 95% quantile is 0.0854 for the 200-long and 0.0612 for the 400-long training. These quantiles result in the 95% confidence intervals (0.744,0.9148) and (0.7114,0.8338) for the corresponding estimates, respectively. Considering relative errors, the 95% quantiles are far too large, being around 25%. The exception is the case of the 400-long fOU trained network when the 95% relative error quantile *localized* around the 0.8 Hurst value is 8.78%. It results in a confidence interval of (0.7047,0.8404). Summarizing, we may say that in all models, Hurst values between 0.8 and 0.83 belong to the 95% confidence interval as can be seen in Table 5.

In Shao and Si,³⁸ several results obtained values between 0.8 and 0.9 for the Hurst parameter estimation using their model, which was defined as $dX(t) = \mu dt + \sigma B_H(t)$, where $X(t)$ represents the degradation process. This way, they assume a linear trend in the process. When modeling by fOU, as we did by applying the fOU-trained network, we introduce a trend of exponential (and random) characters that may play a similar role to the linear one. That may explain why ours and their estimations of the Hurst parameter are close to each other. Remark for clarity that the fOU used in the modeling is not stationary.

In Liu, Song, and Zio,¹ an lfsm, called by them as fractional Lévy stable motion, was applied for modeling the degradation of lithium-ion batteries, where the Hurst parameter estimation yielded values around 0.6, aligning closely with our Higuchi estimation. The α parameter value in their modeling was consistently around 1.8–1.9. Given that the LSTM estimator does not perform satisfactorily in the lfsm case, there is no controversy in our and their findings. Note the well-known fact that when $\alpha = 2$, the lfsm process reduces to an fBm.

Both articles focused on modeling the degradation of lithium-ion batteries, similar to our study, providing valuable insights and benchmarks for our analysis.

TABLE 6 MSE losses of different Hurst-estimators for fractional Brownian motion by sequence length.

seq. len.	MSE loss ($\times 10^{-3}$)		
	M_{LSTM}^{SI}	DeeperSigNet ^{SD}	M_{LSTM}^{SD}
100	4.07	N/A (Error)	0.214
200	1.91	N/A (Error)	0.0826
400	0.917	0.131	0.0366
800	0.453	0.0870	0.0141
1600	0.224	0.199 (ep 34/100)	0.00715
3200	0.114	2.02 (ep 8/100)	0.00373
6400	0.0579	54.2 (ep 1.75/100)	0.00646
12,800	0.0297	87.5 (ep 0/100)	0.00318 (ep 37/100)

Note: "SI" indicates scale-invariant models, "SD" indicates non-scale-invariant models.
Abbreviations: ep, epoch; LSTM, long short-term memory; MSE, mean squared error.

5 | CONCLUSIONS

Our research demonstrates that a neural network with a standard architecture, when trained with a substantial volume of data, significantly outperforms traditional statistical estimators in both accuracy and speed for estimating process parameters, provided it is trained on appropriate process types. The application of various machine learning methods may also be considered as, for example, in ref. [39] but our research primarily focused on the applicability within the realm of neural networks. Consequently, we decided not to include these methods in our current analysis. However, skewness in the data can adversely affect the accuracy of the deep learning estimator, particularly in scenarios involving rare but significant losses or risks, as highlighted by the error quantiles. Despite the overall high performance, the relative errors in the estimations can still be considerable under certain parameter configurations.

The training phase of the neural network may extend to several hours, but the estimation time remains significantly lower – by one or two orders of magnitude – compared to conventional methods. In terms of cross-process applicability, the network trained on fBm data also shows commendable performance when applied to fOU processes, but its effectiveness is reduced for fractional Lévy stable motions.

Our research is dedicated to exploring and comparing these two methodologies, with a particular focus on the fBm and fOU models. We concentrate on estimating the critical parameters within these frameworks and have observed that these estimations are not directly transferable to the alternative lfsm framework. This insight underscores the distinct nature of each modeling approach and highlights the necessity of tailoring the analytical methods to the specific characteristics and challenges of the data involved in battery degradation.

The methodology is applied to estimating the Hurst parameter in li-ion battery degradation data. The obtained confidence bounds conform with the findings of previously published research.

The proposed methodology, which includes the integration of lfsm, reveals certain limitations under specific conditions, particularly in terms of the reliability of the estimates when the underlying process structures are altered or when the driving noise is not fBm. These findings suggest that the methodology might not only fail to maintain the same level of estimation reliability as observed with fBm and fOU processes but also struggles with fitting even within fBm models under less-than-ideal conditions. Therefore, while the network can perform effectively under certain specific conditions, its ability to generalize across diverse scenarios, especially those involving non-fBm noise types or imperfect fits within fBm models, is limited. This points to a need for caution when applying this methodology to processes driven by different types of noise or also when the fit of fBm-driven models are weak.

Our results suggest that transforming the evaluation data can lead to impractical slowdowns in the estimation process, indicating that approaches like signature transform or certain transformer classes may not be promising alternatives. However, the current introduction of the iTransformer⁴⁰ offers a promising alternative. The iTransformer inverts the duties of the attention mechanism and the feed-forward network, focusing on embedding individual time series as variate tokens to capture multivariate correlations effectively. The iTransformer's ability to handle multivariate correlations makes it a highly viable and efficient backbone for time series forecasting, addressing the limitations of traditional Transformer-based forecasters. Therefore, we intend to exploit its novel capabilities in our further research.

However, the deep signature transforms²² emerge as a promising solution in scenarios where scale invariance is not a primary concern, offering more accurate results compared to the models we evaluated. The trade-off here is the lack of scale invariance and significantly longer training times for extended sequences, which is a critical consideration. To

illustrate this point, we present a comparative table showcasing the performance differences between deep signature transforms and the models previously discussed.

Table 6 presents the mean squared error (MSE) losses of various fBm Hurst estimators across different sequence lengths. It is evident that scale-invariant models, such as M_{LSTM}^{SI} , generally exhibit higher MSE losses as the sequence length increases, whereas non-scale invariant models, particularly M_{LSTM}^{SD} , demonstrate significantly lower losses, indicating better performance. Notably, the DeeperSigNet^{SD} model failed to produce results (Error) in several instances, and where it did provide results, the losses were substantially higher, especially for longer sequences.

ACKNOWLEDGMENTS

We would like to express our sincere gratitude to the two anonymous reviewers for their insightful comments and valuable suggestions. Their constructive feedback has been instrumental in enhancing the quality of our work. We would like to thank the National Research, Development and Innovation Office for the support within the framework of the Thematic Excellence Program 2021 – National Research Sub programme: “Artificial intelligence, Large Networks, data security: mathematical foundation and applications” and the Artificial Intelligence National Laboratory Program (MILAB). We appreciate the support provided by neptune.ai. We are also incredibly thankful for the Ministry of Innovation and Technology of Hungary from the National Research, Development and Innovation Fund, that took part in financing this research under the Eötvös Loránd University TKP 2021-NKTA-62 funding scheme. The last author gratefully acknowledges partial support from the Fulbright Teaching and Research Award while starting this research.

DATA AVAILABILITY STATEMENT

The dataset utilized in this study was provided by the NASA Prognostics Center of Excellence (PCoE). It contains data from experiments conducted on Li-Ion batteries, which include charging and discharging cycles at various temperatures, with impedance recorded as a damage criterion. The dataset is publicly available and can be accessed through the NASA PCoE Data Repository at: <https://www.nasa.gov/intelligent-systems-division/discovery-and-systems-health/pcoe/pcoe-data-set-repository/>.

ORCID

Dániel Boros  <https://orcid.org/0009-0008-1207-2251>

László Márkus  <https://orcid.org/0000-0002-2883-5048>

REFERENCES

- Liu H, Wanqing S, Enrico Z. Fractional Lévy stable motion with LRD for RUL and reliability analysis of li-ion battery. *ISA Trans.* 2022;125:360-370.
- Samorodnitsky G. Long range dependence. <https://people.orie.cornell.edu/gennady/techreports/LRD-NOW.pdf>
- Zhang H, Chen M, Xi X, Zhou D. Remaining useful life prediction for degradation processes with long-range dependence. *IEEE Trans Rel.* 2017;66(4):1368-1379.
- Xi X, Chen M, Zhou D. Remaining useful life prediction for degradation processes with memory effects. *IEEE Trans Reliab.* 2017;66(3):751-760.
- Zhang H, Zhou D, Chen M, Shang J. FBM-based remaining useful life prediction for degradation processes with long-range dependence and multiple modes. *IEEE Trans Reliab.* 2019;68(3):1021-1033.
- Si W, Shao Y, Wei W. Accelerated degradation testing with long-term memory effects. *IEEE Trans Reliab.* 2020;69(4):1254-1266.
- Zhang H, Mo Z, Wang J, Miao Q. Nonlinear-drifted fractional Brownian motion with multiple hidden state variables for remaining useful life prediction of lithium-ion batteries. *IEEE Trans Reliab.* 2020;69(2):768-780.
- Zhang H, Zhou D, Chen M, Xi X. Predicting remaining useful life based on a generalized degradation with fractional Brownian motion. *Mech Syst Sig Process.* 2019;115:736-752.
- Song W, Chen X, Cattani C, Zio E. Multifractional Brownian motion and quantum-behaved partial swarm optimization for bearing degradation forecasting. *Complexity.* 2020;2020:8543131.
- Wang H, Song W, Zio E, Kudreyko A, Zhang Y. Remaining useful life prediction for Lithium-ion batteries using fractional Brownian motion and fruit-fly optimization algorithm. *Measurement.* 2020;161:107904.
- Ding H, Yang L, Cheng Z, Yang Z. A remaining useful life prediction method for bearing based on deep neural networks. *Measurement.* 2021;172:108878. doi: 10.1016/j.measurement.2020.108878
- Qin Y, Chen D, Xiang S, Zhu C. Gated dual attention unit neural networks for remaining useful life prediction of rolling bearings. *IEEE Trans Ind Inf.* 2021;17:6438-6447. doi: 10.1109/TII.2020.2999442
- Yan H, Qin Y, Xiang S, Wang Y, Chen H. Long-term gear life prediction based on ordered neurons LSTM neural networks. *Measurement.* 2020;165:108205. doi: 10.1016/j.measurement.2020.108205
- Haris M, Hasan MN, Qin S. Early and robust remaining useful life prediction of supercapacitors using BOHB optimized deep belief network. *Appl Energy.* 2021;286:116541.

15. Wang H, Peng M, Liu Y, Liu S, Xu R, Saeed H. Remaining useful life prediction techniques of electric valves for nuclear power plants with convolution kernel and LSTM. *Sci Technol Nucl Install*. 2020;1–13. doi: [10.1155/2020/8349349](https://doi.org/10.1155/2020/8349349)
16. Fan Y, Nowaczyk S, Rögnvaldsson T. Transfer learning for remaining useful life prediction based on consensus self-organizing models. *Reliab Eng Syst Saf*. 2020;203:107098. doi: [10.1016/j.res.2020.107098](https://doi.org/10.1016/j.res.2020.107098)
17. Lenzi A, Bessac J, Rudi J, Stein ML. Neural networks for parameter estimation in intractable models (Version 1). 2021 *arXiv*. doi: [10.48550/ARXIV.2107.14346](https://doi.org/10.48550/ARXIV.2107.14346)
18. Han D, Korabel N, Chen R. Deciphering anomalous heterogeneous intracellular transport with neural networks. *eLife*. 2020;9:e52224.
19. Ledesma-Orozco S, Ruiz-Pinales J, García-Hernández G, Cerda-Villafaña G, Hernández-Fusilier D. Hurst parameter estimation using artificial neural networks. *J Appl Res Technol* 2011;9(2):227-241.
20. Kirichenko L, Pavlenko K, Khatsko D. Wavelet-based estimation of Hurst exponent using neural network. In: *2022 IEEE 17th International Conference on Computer Sciences and Information Technologies (CSIT)*. IEEE; 2022:40-43.
21. Mukherjee S, Sadhukhan B, Das AK, Chaudhuri A. Hurst exponent estimation using neural network. *Int J Comput Sci Eng*. 2023;26:157-170.
22. Bonnier P, Kidger P, Perez Arribas I, Salvi C, Lyons T. Deep signature transforms. *Adv Neural Inf Process Syst*. 2019;33:3105-3115.
23. Mandelbrot BB, Van Ness JW. Fractional Brownian motions, fractional noises and applications. *SIAM Rev*. 1968;10(4):422-437.
24. Cheridito P, Kawaguchi H, Maejima M. Fractional Ornstein-Uhlenbeck processes. *Electron J Probab* 2003;8(3):1-14.
25. Nualart D. *The Malliavin Calculus and Related Topics*. 2nd ed. Probability and Its Applications. Springer; 2006.
26. Biagini F, Hu Y, Øksendal B, Zhang T. *Stochastic Calculus for Fractional Brownian Motion and Applications*. Springer Science and Business Media; 2008.
27. Coutin L. An introduction to (stochastic) calculus with respect to fractional Brownian motion. In: *Séminaire de Probabilités XL*. Springer; 2007:3-65.
28. Høg E, Frederiksen P. *The Fractional Ornstein-Uhlenbeck Process: Term Structure Theory and Application*. Aarhus School of Business; 2006.
29. Samorodnitsky G, Taqqu MS. *Stable Non-Gaussian Random Processes. Stochastic Models with Infinite Variance. Stochastic Modeling*. Chapman & Hall; 1994:343-349.
30. Huillet T. Fractional Lévy motions and related processes. *J Phys A: Math Gen*. 1999;32(42):7225. doi: [10.1088/0305-4470/32/42/301](https://doi.org/10.1088/0305-4470/32/42/301)
31. Hochreiter S, Schmidhuber J. Long short-term memory. *Neural Comput*. 1997;9(8):1735-1780.
32. Loshchilov I, Hutter F. Decoupled weight decay regularization (Version 3). 2017. *arXiv*. doi: [10.48550/ARXIV.1711.05101](https://doi.org/10.48550/ARXIV.1711.05101)
33. Kroese DP, Botev ZI. Spatial process generation (Version 1). 2013. *arXiv*. doi: [10.48550/ARXIV.1308.0399](https://doi.org/10.48550/ARXIV.1308.0399)
34. Davies RB, Harte DS. *Tests for Hurst Effect*. Biometrika; 1987;74:95-102.
35. Dieker T. *Simulation of Fractional Brownian Motion*. Department of Mathematical Sciences, The Netherlands and University of Twente; 2004.
36. Xi X, Chen M, Zhang H, Zhou D. An improved non-Markovian degradation model with long-term dependency and item-to-item uncertainty. *Mech Syst Sig Process*. 2018;105:467-480.
37. Zhang H, Chen M, Xi X, Zhou D. Remaining useful life prediction for degradation processes with long-range dependence. *IEEE Trans Reliab*. 2017;66(4):1368-1379.
38. Yunfei S, Wujun S. Degradation modeling with long-term memory considering measurement errors. *IEEE Trans Reliab*. 2023;72:177-189.
39. Raubitzek S, Corpaci L, Hofer R, Mallinger K. Scaling exponents of time series data: a machine learning approach. *Entropy*. 2023;25:1671. doi: [10.3390/e25121671](https://doi.org/10.3390/e25121671)
40. Liu Y, Hu T, Zhang H, et al. iTransformer: Inverted transformers are effective for time series forecasting (Version 4). 2023. *arXiv*. doi: [10.48550/ARXIV.2310.06625](https://doi.org/10.48550/ARXIV.2310.06625)

How to cite this article: Boros D, Csanády B, Ivkovic I, Nagy L, Lukács A, Márkus L. Deep learning the Hurst parameter of linear fractional processes and assessing its reliability. *Qual Reliab Eng Int*. 2024;1-19. <https://doi.org/10.1002/qre.3641>

AUTHOR BIOGRAPHIES



Dániel Boros, a financial mathematics MSc graduate, is currently pursuing his PhD in Applied Mathematics. His research centers on fractional processes such as the Ornstein–Uhlenbeck process, exploring their application in stochastic correlation and the Heston model. His studies involve parameter estimation analysis and the utilization of stochastic correlation processes in sophisticated financial models. Dániel Boros is also exploring the integration of neural networks to enhance the simulation and understanding of these processes. His work effectively merges advanced mathematics with AI to shed light on complex financial phenomena and improve predictive modeling. Additionally, he is a member of the Eötvös Loránd University AI

Research Group.



Bálint Csanády is a PhD student at the Institute of Mathematics, Eötvös Loránd University. His work in the AI Research Group focuses on research and industrial projects, mainly in natural language processing and 1D sequence modeling.



Iván Ivkovic, a financial mathematics expert, focuses on machine learning for dependent datastreams and statistical learning theory. During his MSc in Applied Mathematics, he studied isonormal processes and applied learning theory fundamentals to deep neural networks. He collaborates with researchers to develop estimators for fractal noise-driven stochastic processes and algorithms for data generation. Iván Ivkovic works at the Alfréd Rényi Institute of Mathematics in the Financial Mathematics Research Group and in the Eötvös Loránd University AI Research Group, supervising theses on price forecasting with parametric models in the mathematics expert in data analytics and machine learning program.



Lóránt Nagy specializes in financial mathematics, focusing on optimal trading and utility maximization. His interests include non-Markovian market dynamics, long memory price models, utility theory, and turnpike theorems. He has substantial industrial exposure to projects that require the simultaneous application of econometrics, mathematical finance theory, and machine learning techniques. Examples include forecasting energy market states and generating alpha. Recently, he has been working on applying deep learning techniques to high-frequency algorithmic trading.



András Lukács received his degree in mathematics from Eötvös Loránd University in 1992 and his PhD in 1998. He specializes in artificial intelligence, machine learning, and natural language processing. He gained his research and industrial project experience while working at the SZTAKI Institute for Computer Science and Control, where he co-founded and led the Data Mining and Web Search Research Group in the early 2000s. Since 2018, András Lukács has been coordinating the Artificial Intelligence and Data Science Research Group at Eötvös Loránd University's Institute of Mathematics. Over the past two decades, he has led or been a leading specialist in dozens of domestic and European-supported and corporate artificial intelligence research and development

projects. He has been teaching at Eötvös Loránd University for more than three decades, introducing the data mining subject and currently organizing and instructing courses on the latest topics of artificial intelligence.



László Márkus is a professor of statistics at the Mathematical Institute of Eötvös Loránd University, with over 30 years of teaching experience. He earned his PhD from Lomonosov Moscow State University and completed a post-doctoral fellowship at Meijo University in Japan. László Márkus became a member of the Bernoulli Society in 1990 and served on its European Committee from 2008 to 2014, chairing it from 2010 to 2012. He was elected as a member of the International Statistical Institute in 2011. In 2019, László Márkus was awarded a Fulbright Teaching and Research Award to the University of Connecticut. He actively participates in the Education Committee on Actuarial and Financial Mathematics at his institution. His teaching portfolio includes subjects

such as mathematical finance and time series analysis. László Márkus has led applied and fundamental research projects funded by significant financial entities and government agencies. He specializes in integrating machine learning techniques into mathematical finance and has authored lecture notes in this field.