https://doi.org/10.56093/JISAS.V78I3.6
Available online at http://isas.org.in/isa/jisas
JOURNAL OF THE INDIAN SOCIETY OF AGRICULTURAL STATISTICS 78(3) 2024 221–229

Hybrid ARFIMA-LRNN Model for Forecasting Commodity Prices

Debopam Rakshit[1] and Ranjit Kumar Paul[2]

[1]*ICAR-Indian Veterinary Research Institute, Izatnagar*
[2]*ICAR- Indian Agricultural Statistics Research Institute, New Delhi*

Received 01 November 2024; Revised 23 November 2024; Accepted 26 December 2024

SUMMARY

The unexpected fluctuation of prices of agricultural commodities may have impactful repercussions on the producers. The price volatility can be modeled by applying time series analysis. A time series consists of linear and non-linear components. The linear component can be modeled by the autoregressive integrated moving average (ARIMA) methodology. Again, long-term dependencies amongst the realizations of any time series can also be observed. This long-term dependency can be addressed by incorporating fractional differencing in the ARIMA model which is known as the autoregressive fractionally integrated moving average (ARFIMA) model. To address both the linear and non-linear components effectively, hybrid time series models can be used. In a hybrid model, more than one model is clubbed together such that one is used for capturing the linear component and another captures the nonlinear counterpart. In this article, a hybrid ARFIMA-LRNN (Layer recurrent neural network) model is employed for modeling the price series of arhar for the Mumbai market. The forecasting accuracy of the hybrid model outperformed the standalone models.

Keywords: ARFIMA; Hybrid model; Long memory; Neural network; RNN.

1. INTRODUCTION

Time series analysis plays a pivotal role in fostering the socio-economic development of any country. The dependencies of successive realizations are the backbone of the time series analysis. Time series data often exhibit both linear and nonlinear characteristics. The linearity can be captured using the autoregressive integrated moving average (ARIMA) model (Box *et al.*, 2015) and its component models. Again, the realizations of a time series may be affected by its own values at distanced lag. This is due to long-term dependency among the realizations. To address this situation, the fractionally integrated version of the ARIMA model, i.e., the ARFIMA model (Granger and Joyeux, 1980) is useful. In literature, there are numerous uses for the ARIMA and ARFIMA models (Paul, 2014). To capture both the linear and nonlinear dynamics, a hybrid model can be more effective. Hybrid time series models, one for linear components and another for nonlinear components are more statistically sound than standalone models.

The study of price fluctuation of agricultural commodities is an important area for researchers. The foundation of the Indian economy is agriculture. Agriculture provides livelihood to around half of the country's population. Modeling and forecasting of the price of various agricultural commodities may be immensely helpful for farmers, policymakers, government agencies, agribusiness industries, and also for a large number of middlemen in the supply chain. Different studies are available in the literature regarding the modeling and forecasting of agricultural commodity prices using various parametric models (Rakshit *et al.*, 2021; Rakshit and Paul, 2024a; Rakshit and Paul, 2024b).

Of recent, different machine learning (ML) and deep learning (DL) techniques like artificial neural networks (ANN), generalized neural network (GRNN), random forest (RF), support vector regression (SVR), recurrent neural networks (RNN) etc. are gaining popularity for time series forecasting. Paul *et al.* (2022) used various

Corresponding author: Ranjit Kumar Paul
E-mail address: ranjit.paul@icar.gov.in

ML techniques for modeling and forecasting brinjal prices and found that GRNN performed better than others. Garai *et al.* (2023) found that the wavelet-based RF model is the best performer among other ML models for modeling and forecasting onion price series of major Indian markets. Paul *et al.* (2023) studied various ML and DL models for forecasting cauliflower prices. Garai *et al.* (2024) used the SVR model to predict the price fluctuation of onion induced by the nationwide lockdown due to COVID-19 in India. Tamilselvi *et al.* (2024) found that the wavelet decomposed long short-term memory (LSTM) model performed significantly better than other ML models for the forecasting of the price series of spices.

Arhar is among the pulses subject to frequent and significant price variations, which can also affect the pricing of other pulses. Given the importance of pulses to the Indian diet, any notable rise in arhar prices might have a ripple impact on the country's overall food affordability and inflation rates. The top three arhar producer states are Maharashtra, Karnataka and Uttar Pradesh (Agricultural Statistics at a Glance, 2022). Previously, some research has been done for the forecasting of the arhar price series. Paul *et al.* (2015) used the ARFIMA model for modeling and forecasting the arhar price in Karnal, Haryana. Mitra *et al.* (2018) followed the two-stage forecasting algorithm for modeling and forecasting of arhar price in Bhopal, Madhya Pradesh using the ARFIMA model in association with a structural break. In this manuscript, the price series of arhar for the Mumbai market is studied using the hybrid ARFIMA-LRNN (Layer recurrent neural network) model (Pwasong and Sathasivam, 2018). This hybrid model's forecasting effectiveness is assessed, and it is confirmed that it is better than the standalone models.

2. LONG MEMORY PROCESS AND TEST FOR LONG MEMORY

The statistical dependency among the realizations of any time series is generally measured through the autocorrelation function (ACF). For any stationary time series $\{y_t\}$, the autocorrelation with a time lag k is defined as

$$\rho_k = \frac{cov(y_t, y_{t-k})}{var(y_t)} \tag{1}$$

For a short memory process, the decay of the autocorrelation is very rapid over lags (exponential rate) such that $\rho_k \approx |m|^k$, where $|m| < 1$ and k is very large. For a long-term persistence process, the decay of the autocorrelation is very slow (hyperbolic rate) such that $\rho_k \approx Ck^{2d-1}$, where C is a constant and d is the long memory parameter. Again, the expression can be written as $\sum_{k=0}^{\infty} |\rho_k| < \infty$ for short memory and for long memory, it is $\sum_{k=0}^{\infty} |\rho_k| = \infty$. (Brockwell & Davis, 1991)

2.1 ACF plot

ACF plots are commonly used to visualize the correlation between a series and lagged versions of itself. Plotting of ACF is the most popular visual method to identify the presence of long memory in the time series data. If the autocorrelation of the process decreases very slowly or hyperbolically (not exponentially), then it represents the possible presence of long-term persistence.

2.2 GPH estimate

The most widely used method for long memory parameter estimation is Geweke and Porter-Hudak's (1983) GPH statistic. This method is based on the spectral density at the origin

$$f_y(\lambda) \sim \lambda^{-2d}, \lambda \to 0 \tag{2}$$

For a stationary model y_t ,$t = 1, 2, \ldots, T$, the spectral density function can be expressed as

$$f_y(\lambda) = \left[4\sin^2\left(\frac{\lambda}{2}\right)^{-d}\right] f_\varepsilon(\lambda) \tag{3}$$

where $f_\varepsilon(\lambda)$ is the spectral density of ε_t, which is a continuous and finite function on the interval $[-\pi, \pi]$. The log-spectral density can be expressed as

$$log\left(f_y(\lambda)\right) = log\left(f_\varepsilon(0)\right) - dlog\left[4\sin^2\left(\frac{\lambda}{2}\right)\right] + log\frac{f_\varepsilon(\lambda)}{f_\varepsilon(0)} \tag{4}$$

Now, let $I_y(\lambda_j)$ be the sample periodogram, then the log-spectral density can be expressed as

$$log\left(I_y\left(\lambda_j\right)\right)=log\left(f_\varepsilon\left(0\right)\right)-dlog\left[4\sin^2\left(\frac{\lambda_j}{2}\right)\right]+log\frac{f_\varepsilon\left(\lambda_j\right)}{f_\varepsilon\left(0\right)}+log\frac{I_y\left(\lambda_j\right)}{f_y\left(\lambda_j\right)} \quad (5)$$

where λ_j are the Fourier frequencies such that $\lambda_j=2\pi j/T$, $j=1,2,\ldots,n$ and $n=\sqrt{T}$

The GPH estimator is based on two hypotheses. These are

H_{01}: For low frequencies, the term $log\frac{f_\varepsilon\left(\lambda_j\right)}{f_\varepsilon\left(0\right)}$ is negligible.

H_{02}: The random variable $log\frac{I_y\left(\lambda_j\right)}{f_y\left(\lambda_j\right)}$ is asymptotically distributed and IID.

Under the above-mentioned hypotheses, the linear regression (equation no. 5) can be written as

$$log\left(I_y\left(\lambda_j\right)\right)=\alpha-dlog\left[4\sin^2\left(\frac{\lambda_j}{2}\right)\right]+\varepsilon_j \quad (6)$$

where, α is the intercept term and defned as the $log\left(f_\varepsilon\left(0\right)\right)$ plus the mean of $log\frac{I_y\left(\lambda_j\right)}{f_y\left(\lambda_j\right)}$.

The parameter d can be estimated by the ordinary least squares method by considering the above equation as a regression of $log\left(I_y\left(\lambda_j\right)\right)$ on $log\left[4\sin^2\left(\frac{\lambda_j}{2}\right)\right]$.

3. ARFIMA MODEL

ARIMA methodology is used for addressing the linear component of a short memory process. Here, it is assumed that a variable's future realization is linearly dependent on both its previous lag-realizations and random errors. Generally, this is represented as ARIMA $\left(p,d,q\right)$ where p is the order of autoregression, d indicates the order of integration (differencing) and q represents the order of moving average. For a univariate time series $\left\{y_t\right\}$, the ARIMA process is represented as

$$\varphi\left(L\right)\left(1-L\right)^d y_t=\theta\left(L\right)\varepsilon_t \quad (7)$$

where, y_t is the actual observation and ε_t is the error term observed at time t such that $\varepsilon_t\sim IID\left(0,\sigma^2\right).\varphi\left(L\right)$ and $\theta\left(L\right)$ denote the polynomial of lag operator L of order p and q, respectively. In ARIMA methodology, the order of differencing d is considered as an integer. ARFIMA model is used for a long memory process in the same line as the ARIMA model except with a fractional value of d ($-0.5<d<0.5$ (Hosking, 1981)). Fractional differentiation is a general form of integer differentiation. The parameters can be estimated using the quasi-maximum likelihood estimation technique (Tsai, 2006).

4. LRNN MODEL

Elman (1990) proposed the basic form of the LRNN. LRNNs are similar to feed-forward neural networks. Every layer in an LRNN has a recurrent connection and a tap delay associated with it. The concept of LRNN is similar to time delay and distributed delay neural networks (Liu *et al.*, 2012).

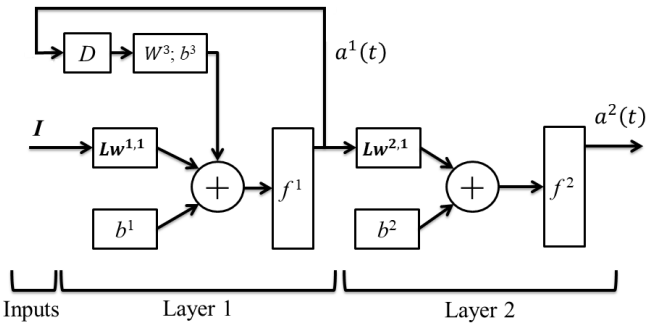

Fig. 1. A two-layered LRNN architecture

In this article, a two-layered LRNN is employed. The schematic diagram of the LRNN model is given in Figure 1. The feedback layer, i.e., layer 1 provides recurrence. The feed-forward layer is layer 2 and this layer propagates the information in the forward direction. The b^1 and b^2 indicate the bias, the f^1 and f^2 indicate the hidden layers, and the $a^1\left(t\right)$ and $a^2\left(t\right)$ are the outputs of layer 1 and layer 2, respectively. The weights for the connections between the input layer to the hidden layer and between the hidden layer to the output layer are denoted by $\boldsymbol{Lw}^{1,1}$ and $\boldsymbol{Lw}^{2,1}$, respectively. The time delay is indicated by *D*. The b^3 is the bias of the time delay later.

The Levenberg-Marquardt (LM) algorithm can be used to train the LRNN. This LM algorithm combines

the benefits of both the gradient descent and Gauss-Newton methods. It acts like gradient descent for large errors and switching to Gauss-Newton for fine-tuning near the optimal point. The Hessian matrix can be calculated using a back-propagation (BP) algorithm. The parameters are updated by the search scheme (Levenberg, 1944):

$$\left[\boldsymbol{J}'\boldsymbol{W}\boldsymbol{J}+\lambda\boldsymbol{I}\right]\boldsymbol{\delta}=\boldsymbol{J}'\boldsymbol{W}\left(\boldsymbol{y}-\hat{\boldsymbol{y}}\right) \tag{8}$$

where, $\boldsymbol{J}$ is the Jacobian matrix of the errors, $\boldsymbol{W}$ is the weight matrix, λ is a non-negative scalar known as damping parameter, $\boldsymbol{I}$ is an identity matrix, $\boldsymbol{\delta}$ is the perturbation of the parameters at each iteration step, $\boldsymbol{y}$ and $\hat{\boldsymbol{y}}$ are the vectors of actual and predicted values. Later, Marquardt (1963) modified it by replacing $\boldsymbol{I}$, with $diag\left(\boldsymbol{J}'\boldsymbol{W}\boldsymbol{J}\right)$ resulting in the Levenberg-Marquardt algorithm:

$$\left[\boldsymbol{J}'\boldsymbol{W}\boldsymbol{J}+\lambda\, diag\left(\boldsymbol{J}'\boldsymbol{W}\boldsymbol{J}\right)\right]\boldsymbol{\delta}=\boldsymbol{J}'\boldsymbol{W}\left(\boldsymbol{y}-\hat{\boldsymbol{y}}\right) \tag{9}$$

In two-layered LRNN, different activation functions are used in different situations. For hidden neurons, it uses sigmoid function. The sigmoid function is smooth and continuously differentiable, which is essential for gradient-based optimization algorithms like back propagation.

Sigmoid function:

$$f^{h}\left(x(t)\right)=\frac{1}{1+e^{-x(t)}} \tag{10}$$

For time delay neurones, a time delay function is employed as an activation function and for output neurones, it is served by an identity function.

Time delay function:

$$f^{c}\left(x(t)\right)=x(t-1) \tag{11}$$

Identity function:

$$f^{o}\left(x(t)\right)=x(t) \tag{12}$$

The error is calculated as the discrepancy between the LRNN's output and the referenced output. This error is used to optimize the weights. The sum of the input weights $\boldsymbol{Lw}^{1,1}$ and $\boldsymbol{Lw}^{2,1}$ with the bias b^1 and b^2 are calculated by:

$$\gamma_1(t)=\left[\boldsymbol{Lw}^{1,1},f(t-1)\right]+b^1 \tag{13}$$

$$\gamma_2(t)=\left[\boldsymbol{Lw}^{2,1},f(t-1)\right]+b^2 \tag{14}$$

The input to the output layer of layer 1 and layer 2 can be respectively,

$$f^{1}(t)=f^{0}\left(x(t)\right)\left(\gamma_1(t)\right) \tag{15}$$

$$f^{2}(t)=f^{0}\left(x(t)\right)\left(\gamma_2(t)\right) \tag{16}$$

The input of the outputs to the hidden layer neurons is given by:

$$\gamma_f(t)=\left[\boldsymbol{W}_1x(t)\right]+\left[\boldsymbol{Lw}^{1,1}f(t)\right]+\left[\boldsymbol{Lw}^{2,1}f(t)\right]+b_1$$

$$h(t)=f^{h}\left(x(t)\right)\left(\gamma_f(t)\right) \tag{17}$$

where, $\left(\gamma_f(t)\right)$ is the aggregated input to the hidden layer neurons, $\boldsymbol{W}_1$ represents the weight connecting the input layer and the hidden layer. $\boldsymbol{Lw}^{1,1}f(t)$ and $\boldsymbol{Lw}^{2,1}f(t)$ represent the output weights of the feedback layer. Computations of the input to the context neuron of the context layer are:

$$\gamma_c(t)=\left[\boldsymbol{W}_2h(t)\right]+\left[\boldsymbol{W}_3y(t)\right]+b_3$$

$$c(t)=f^{c}\left(x(t)\right)\left(\gamma_c(t)\right) \tag{18}$$

where, $\gamma_c(t)$ is the aggregated input to the context layer neurons, $\boldsymbol{W}_2$ is the weight connecting the hidden layer and context layer and b_3 is the bias applied to the context layer neurons. $\boldsymbol{W}_3$ represents the context weights of the feedback layer.

5. THE HYBRID ARFIMA-LRNN MODEL

Merging a neural network model with a time series model can be used as a hybrid model to model the data more precisely. It can be represented as follows:

$$Z_t=Y_t+X_t \tag{19}$$

where, Y_t and X_t are the linear and nonlinear components, respectively. The first step involves fitting of the ARFIMA model to the dataset. Then, the residuals are obtained and this residual series is used as the input to the LRNN model because it is assumed that only the nonlinear component exists in the residual series. For training the model, the least mean absolute error (MAE) principle is utilized. The forecast obtained from the LRNN is added with the forecast obtained from the ARFIMA model to get the final forecast.

Let $\{e_t\}$ is the residual series of the ARFIMA model, then $e_t=Z_t-\hat{Y}_t$, where, $\hat{Y}_t$ is the predicted value. With n input nodes, the LRNN model for the residuals will be

$$e_t = f\left(e_{t-1}, e_{t-2}, \ldots, e_{t-n}\right) + \varepsilon_t \quad (20)$$

where, f is a non-linear function considered by the LRNN method and $\varepsilon_t \sim IID\, N\left(0, \sigma^2\right)$ is the random error. If the predicted value from the LRNN is $\hat{X}_t$, then the resultant prediction would be

$$\hat{Z}_t = \hat{Y}_t + \hat{X}_t \quad (21)$$

6. EMPIRICAL ILLUSTRATION

To conduct the study, the daily wholesale prices of arhar for the Mumbai market between the time period of 1st August 2012 to 31st December 2020, are collected from the Ministry of Consumer Affairs, Food and Public Distribution, Government of India. The Table 1 represents the descriptive statistics of the price series and its plot is depicted in Figure 2. The analysis has been done using the 'rugarch' package of R programming language and the 'layrecnet' command of MATLAB programming language.

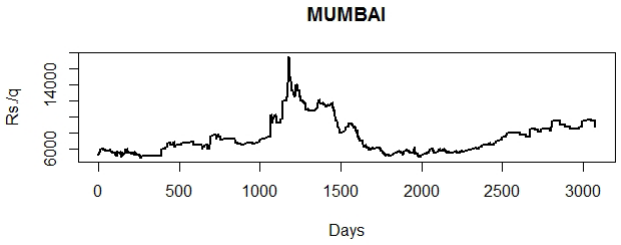

Fig. 2. Time plot of arhar price series

Table 1. Descriptive statistics

Statistics	Price Series
Mean (Rs./quintal)	7446.63
Median (Rs./quintal)	6750.00
Minimum (Rs./quintal)	4990.00
Maximum (Rs./quintal)	17500.00
S.D. (Rs./quintal)	2065.85
C. V. (%)	27.74
Skewness	1.26
Kurtosis	1.35

A prerequisite assumption for applying the ARMA methodology is the absence of unit root in the dataset, i.e., the data should be stationary. Two statistical tests namely, the Augmented Dickey-Fuller (ADF) test (Dickey and Fuller, 1979), and Phillips-Perron (PP) test (Phillips and Perron, 1988) are carried out for this purpose. The results of these tests are given in Table 2. For both tests, the null hypothesis is that a unit root is present in the underlying series. These tests confirm the presence of unit root in the price series. As the price series is not stationary, the difference series is obtained. The absence of a unit root in the difference series is confirmed by both tests. Hence, the further steps are carried out using the difference series.

Table 2. Test for stationarity

Test	Price Series		Difference Series	
	Test statistic	p-value	Test statistic	p-value
ADF	-1.95	0.60	-13.80	0.01
PP	-1.79	0.67	-52.58	0.01

The Shapiro-Wilk test (Shapiro and Wilk, 1965) has been carried out to check the normality of the price series and the differenced series. The null hypothesis for this test is that the series is normally distributed. It is seen that both series do not follow normality (Table 3). Figure 3 depicts the kernel density plot of the difference series. The non-normality of the difference series is also supported by the kernel density plot. In the absence of normal distribution for both the series, it is considered that they are following the generalized error distribution (GED) as a heavy-tailed alternative one.

Table 3. Test for normality (Shapiro-Wilk test)

	Price Series	Difference Series
Test statistic	0.879	0.299
p-value	<0.001	<0.001

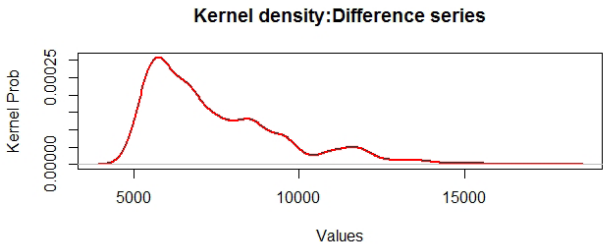

Fig. 3. Kernel density plot of the difference series

The statistical dependency among the realizations of the price series and the difference series are investigated with the help of the ACF and partial ACF plots (Figure 4). From the ACF, it is found that statistical dependency is present among the successive observations. It is also seen that the autocorrelations over the lag are decreasing at a hyperbolic rate which confirms the presence of long memory. This inference

is confirmed by the GPH test (Table 4). For this test, the null hypothesis is defined as $d = 0$ against the alternative $d \neq 0$. The existence of long memory is confirmed for the difference series as well as for the original price series by this test.

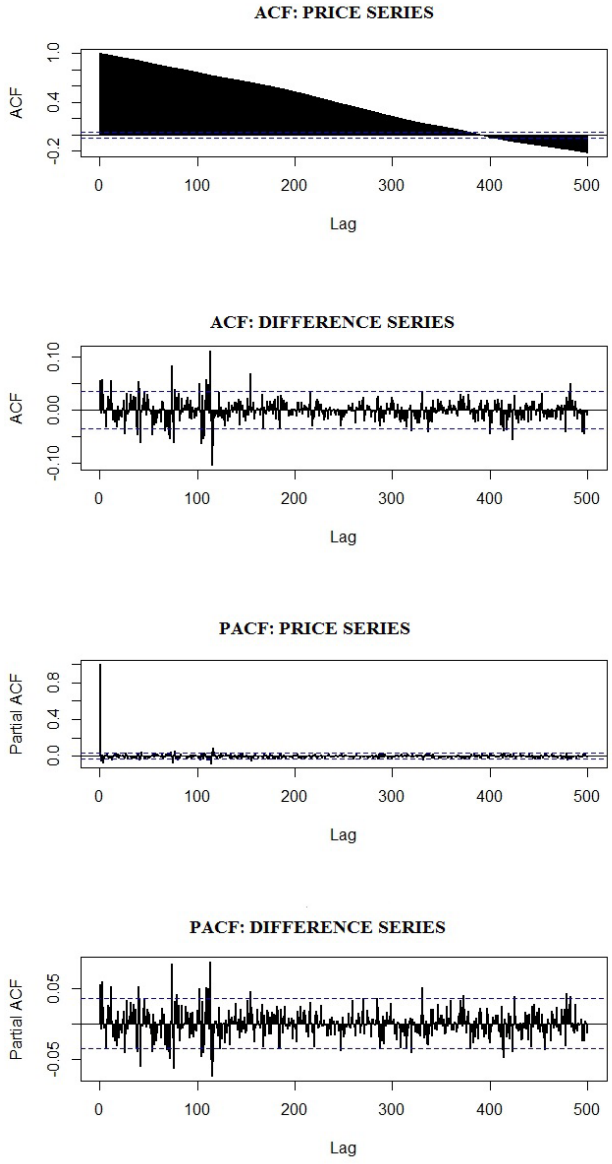

Fig. 4. ACF and PACF plots

Table 4. Test for long memory (GPH test)

	Price Series	Difference Series
d	1.02	0.201
S.D.	0.04	0.097

In the next step, the whole difference series is grouped into two parts, a) model initialization set, which is used to train the model and b) holdout set, which is used to test the forecasting performance. The data set is comprised of 3075 data points, hence, the last three hundred data points are used as the holdout set and the remaining previous realizations as the model initialization set. ARMA models of different order are fitted to the model initialization set. The Akaike information criterion (AIC), Bayesian information criterion (BIC), and Hannan-Quinn information criterion (HQC) are used to select the best-fitted ARMA order. The best-fitted ARFIMA model is obtained as ARFIMA (2, d, 2) and the parameters of the fitted model are given in Table 5. It has been seen that all the parameters of the ARFIMA model are highly significant.

Table 5. The estimated parameters of the fitted ARFIMA (2, d, 2) model

Parameters	Estimate
μ	-0.637***
φ_1	-0.047***
φ_2	0.266***
θ_1	0.037***
θ_2	-0.274***
d	0.011***
ν	0.171***

***p<0.01

The fitted values are obtained to examine the degree of the model fitting. The in-sample forecast is done using this model for the holdout set. This is considered the performance of the standalone ARFIMA model. As ARFIMA is a linear model, it is assumed that the non-linear component is present in the residual series. The nonlinearity of the residual series is verified using the BDS test (Broock *et al.*, 1996). According to the test's null hypothesis, the underlying series is linear. The test comes out to be significant and it is considered that the residual series is nonlinear.

The residual series is considered as the input to the LRNN for the hybrid ARFIMA-LRNN model. The fitted values and the in-sample forecasted values are obtained from the neural network. These fitted and forecasted values are clubbed with their counterparts obtained from the standalone ARFIMA model to get the corresponding output for the hybrid model. Another standalone LRNN method is applied to the difference

series itself. The fitted and in-sample forecasted values are also obtained for the standalone LRNN method. To train the neural network for both scenarios the number of hidden nodes for layer 1 is considered as 50 and for layer 2 it is 1. Different activation function is used for different types of neurons. For hidden, time delay, and output neurons, the activation functions are sigmoid, time delay, and identity functions, respectively.

The fitting performance is evaluated based on minimum values of Root Mean Squared Error (RMSE), Root Mean Squared Log Error (RMSLE), and Mean Absolute Error (MAE) in the model building set (Table 6). For a series of actual observation $\{y_t\}$ and the corresponding predicted series $\{\hat{y}_t\}$, these three evaluation criteria are calculated as

$$RMSE = \left[\frac{1}{h}\sum_{t=1}^{h}\left(y_t - \hat{y}_t\right)^2\right]^{1/2} \quad (22)$$

$$RMSLE = \left[\frac{1}{h}\sum_{t=1}^{h}\left[ln\left(\hat{y}_t + 1\right) - ln\left(y_t + 1\right)\right]^2\right]^{1/2} \quad (23)$$

$$MAE = \frac{1}{h}\sum_{t=1}^{h}\left|y_t - \hat{y}_t\right| \quad (24)$$

Table 6. Observed vs. fitted performance of fitted models

Model	RMSE	RMSLE	MAE
ARFIMA (2, *d*, 2)	117.91	0.014179	33.40
LRNN	118.01	0.014187	35.70
ARFIMA(2, *d*, 2)-LRNN	**116.81**	**0.014128**	**28.41**

It is seen that the hybrid ARFIMA (2, *d*, 2)-LRNN model is the best performer by all the criteria, followed by the standalone ARFIMA (2, *d*, 2) model and standalone LRNN method. The in-sample forecasting performance of standalone ARFIMA (2, *d*, 2), standalone LRNN, and hybrid ARFIMA (2, *d*, 2)-LRNN models are evaluated using the above-mentioned error functions for six moving windows of 50, 100, 150, 200, 250, and 300 days (Table 7).

From Table 7, it can also be observed that the hybrid model is the best-performed model in the model validation set, irrespective of the horizon length of forecasting, followed by the standalone ARFIMA and LRNN methods. Again, the effect of the long memory property of the data series plays a noticeable role in the in-sample forecasting performance. It has been seen that the forecasting performance is improved when the horizon is increased up to 200 days, after that performance decreases. The short-term forecasting horizons, such as 50 days and 100 days, are worse performers than the long-term forecasting horizon. The extent of the fitting is also visualized for the best-fitted hybrid ARFIMA-LRNN method in Figure 5. The weekly comparison of actual and forecasted values for the first 12 weeks of the holdout set is given in Table 8. For the weekly forecast also, the same order of performance can be seen.

Table 7. The forecasting performance of the fitted models in different moving horizons

Days	Model	RMSE	RMSLE	MAE
50	ARFIMA (2, *d*, 2)	135.46	0.015126	34.46
	LRNN	136.01	0.015212	49.56
	ARFIMA(2, *d*, 2)-LRNN	**135.11**	**0.015049**	**30.47**
100	ARFIMA (2, *d*, 2)	103.22	0.011443	31.46
	LRNN	103.49	0.011474	34.84
	ARFIMA(2, *d*, 2)-LRNN	**103.08**	**0.011425**	**24.95**
150	ARFIMA (2, *d*, 2)	89.44	0.009955	20.81
	LRNN	89.63	0.009987	27.15
	ARFIMA(2, *d*, 2)-LRNN	**89.38**	**0.009945**	**17.82**
200	ARFIMA (2, *d*, 2)	79.07	0.008813	17.99
	LRNN	79.87	0.008887	21.94
	ARFIMA(2, *d*, 2)-LRNN	**78.97**	**0.008801**	**14.59**
250	ARFIMA (2, *d*, 2)	84.41	0.009346	18.71
	LRNN	84.57	0.009387	21.48
	ARFIMA(2, *d*, 2)-LRNN	**84.24**	**0.009325**	**15.03**
300	ARFIMA (2, *d*, 2)	88.23	0.009725	18.51
	LRNN	88.25	0.009727	21.91
	ARFIMA(2, *d*, 2)-LRNN	**88.18**	**0.009720**	**16.67**

Table 8. Weekly actual and forecasted values for different models

Weeks	Actual series (Rs./quintal)	ARFIMA(2, *d*, 2)-LRNN (Rs./quintal)	ARFIMA(2, *d*, 2) (Rs./quintal)	LRNN (Rs./ quintal)
1	8500.00	8500.55	8505.62	8528.84
2	8500.00	8500.54	8505.35	8526.05
3	8500.00	8499.53	8505.08	8523.43
4	8514.29	8507.94	8496.26	8492.41
5	9371.43	9275.65	9261.74	9256.83
6	9600.00	9599.50	9604.38	9616.55
7	9600.00	9600.49	9604.17	9614.57
8	9600.00	9600.48	9603.97	9612.74
9	9600.00	9600.46	9603.79	9611.06
10	9600.00	9600.42	9603.61	9609.52
11	9600.00	9600.39	9603.45	9608.12
12	9485.71	9556.62	9560.44	9563.99

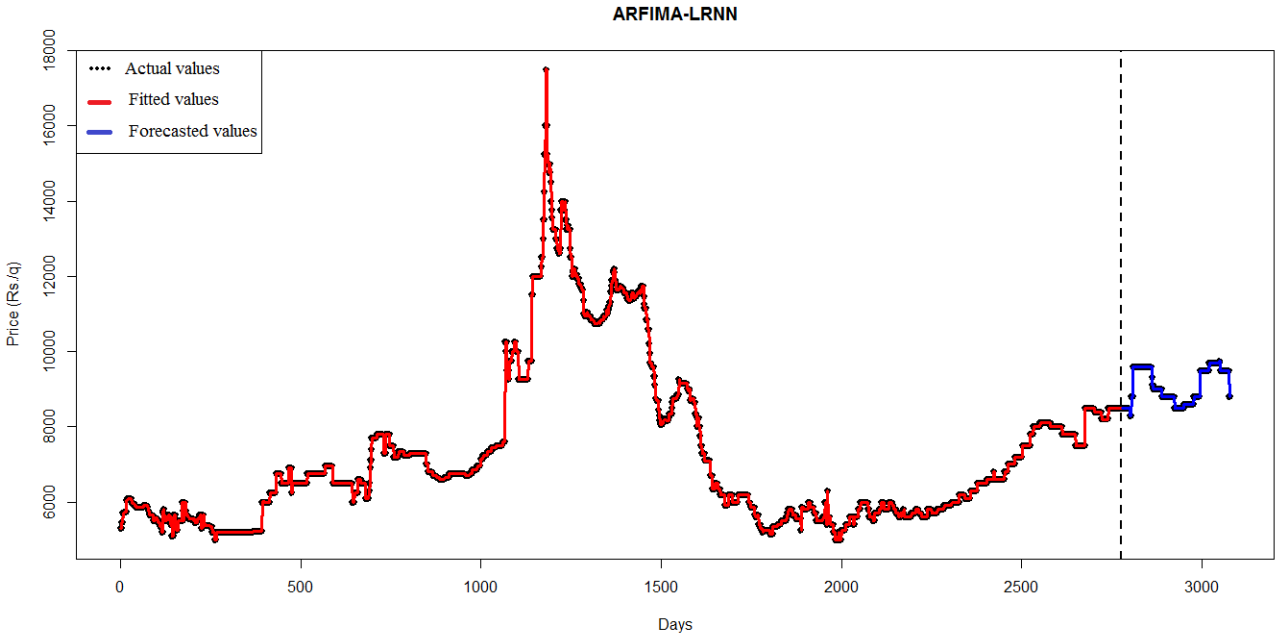

Fig. 5. Visualization of the extent of fitting of the hybrid ARFIMA-LRNN method

7. CONCLUSION

In this article, the predictive efficacy of the hybrid ARFIMA (2, *d*, 2)-LRNN model is evaluated. The performance of this hybrid model is also compared with the performance of the standalone ARFIMA (2, *d*, 2) and standalone LRNN method. The wholesale price data of arhar of the Mumbai market is used for empirical illustration. It has been seen that the hybrid ARFIMA (2, *d*, 2)-LRNN model outperformed as compared to these standalone methods in the model building set. Moving window forecast is done for 50, 100, 150, 200, 250, 300 days horizons. The hybrid model again outperformed the standalone models for all the forecasting horizons. The long memory property played a significant role in forecasting. Forecasting performance improved when the forecasting horizon was increased up to 200 days and then declined. As a further improvement, the hybrid model using LRNN with a greater number of layers can be evaluated.

ACKNOWLEDGEMENTS

The authors acknowledge the suggestions provided by the anonymous reviewers which led significant improvement of the article.

REFERENCES

Box, G.E., Jenkins, G.M., Reinsel, G.C., and Ljung, G.M. (2015). *Time series analysis: forecasting and control*. John Wiley & Sons, New Jersey, USA.

Brockwell, P.J., and R.A. Davis. (1991). Time Series: Theory and Methods, 2nd ed., Springer, New York.

Broock, W.A., Dechert,W.D., and Scheinkman, J.A. (1996). A test for independence based on the correlation dimension. *Econometric Reviews*, **15**(3), 197-235.

Dickey, D., and Fuller, W. (1979). Distribution of the Estimators for Autoregressive Time Series with a Unit Root.*Journal of American Statistical Association*, **74**, 427-431.

Elman, J.L. (1990). Finding structure in time. *Cognitive Science*, **14**, 179-211.

Garai, S., Paul, R.K., Rakshit, D., Yeasin, M., Emam, W., Tashkandy, Y., and Chesneau, C. (2023). Wavelets in combination with stochastic and machine learning models to predict agricultural prices. *Mathematics*, **11**(13), 2896.

Garai, S., Paul, R.K., and Paul, A.K. (2024). Spillover Effects of Covid-19 Induced Lockdown on Onion Prices in India. *Journal of Scientific Research and Reports*, **30**(3), 21-31.

Geweke, J., and Porter-Hudak, S. (1983). The estimation and application of long memory time series models. *Journal of Time Series Analysis*, **4**(4), 221-238.

Granger, C. W. J., and Joyeux, R. (1980). An introduction to long-memory time series models and fractional differencing. *Journal of Time Series Analysis*, **1**(1),15-29.

Hosking, J.R.M. (1981). Fractional differencing. *Biometrika*, **68**,165-176.

Levenberg, K. (1944). A Method for the Solution of Certain Non-Linear Problems in Least Squares. *Quarterly of Applied Mathematics*, **2**(2), 164-168.

Liu, Q., Guo, Z., and Wang, J. (2012). A one-layer recurrent neural network for constrained pseudoconvex optimization and its application for dynamic portfolio optimization. *Neural Networks*, **26**, 99-109.

Marquardt, D. (1963). An Algorithm for Least-Squares Estimation of Nonlinear Parameters. *SIAM Journal on Applied Mathematics*, **11**(2), 431-441.

Mitra, D., Paul, R.K., Paul, A.K., and Bhar, L.M. (2018). Forecasting Time Series Allowing for Long Memory and Structural Break. *Journal of the Indian Society of Agricultural Statistics*, **72**(1), 49-60.

Paul, R.K. (2014). Forecasting wholesale price of pigeon pea using long memory time-series models. *Agricultural Economics Research Review*, **27**(2), 167-176.

Paul, R.K., Gurung, B., and Paul, A.K. (2015). Modelling and forecasting of retail price of arhar dal in Karnal, Haryana. *The Indian Journal of Agricultural Sciences*, **85**(1), 69-72.

Paul, R.K., Yeasin, M., Kumar, P., Kumar, P., Balasubramanian, M., Roy, H.S., ... and Gupta, A. (2022). Machine learning techniques for forecasting agricultural prices: A case of brinjal in Odisha, India. *Plos one*, **17**(7), e0270553.

Paul, R.K., Yeasin, M., Kumar, P., Paul, A.K., and Roy, H.S. (2023). Deep learning technique for forecasting the price of cauliflower. *Current Science*, **124**(9), 1065-1073.

Phillips, P.C.B. and Perron, P. (1988). Testing for a unit root in time series regression, *Biometrika*. **75**(2), 335-346.

Pwasong, A. and Sathasivam, S. (2018). Forecasting Comparisons using a Hybrid ARFIMA and LRNN Models. *Communications in Statistics–Simulation and Computation*, **47**(8), 2286-2303.

Rakshit, D., Paul, R.K., and Panwar, S. (2021). Asymmetric price volatility of onion in India. *Indian Journal of Agricultural Economics*, **76**(2), 245-260.

Rakshit, D., and Paul, R.K. (2024a). Development of out-of-sample forecast formulae for the FIGARCH model. *Model Assisted Statistics and Applications*, **19**(2), 133-143.

Rakshit, D., and Paul, R.K. (2024b). Modeling Time Series with Asymmetric Volatility and Long Memory. *Indian Journal of Agricultural Economics*, **79**(2), 231-244.

Shapiro, S. S., and Wilk, M.B. (1965). An analysis of variance test for normality (complete samples). *Biometrika*, **52**(3/4), 591-611.

Tamilselvi, C., Yeasin, M., Paul, R. K., and Paul, A.K. (2024). Can Denoising Enhance Prediction Accuracy of Learning Models? A Case of Wavelet Decomposition Approach. *Forecasting*, **6**(1), 81-99.

Tsai, H. (2006). Quasi-Maximum Likelihood Estimation of Long-Memory Limiting Aggregate Processes. *Statistica Sinica*, 16(1), 213-226.