Contents lists available at ScienceDirect

Computer Methods and Programs in Biomedicine

journal homepage: www.elsevier.com/locate/cmpb

Granger causality test with nonlinear neural-network-based methods: Python package and simulation study.

Maciej Rosoł*, Marcel Młyńczak, Gerard Cybulski

Faculty of Mechatronics, Institute of Metrology and Biomedical Engineering, Warsaw University of Technology, Warsaw, Poland

ARTICLE INFO

Article history:
Received 7 September 2021
Revised 25 January 2022
Accepted 26 January 2022

Keywords:
Granger causality
Time series
Neural networks
Python
Prediction models

ABSTRACT

Background and objective: Causality defined by Granger in 1969 is a widely used concept, particularly in neuroscience and economics. As there is an increasing interest in nonlinear causality research, a Python package with a neural-network-based causality analysis approach was created. It allows performing causality tests using neural networks based on Long Short-Term Memory (LSTM), Gated Recurrent Unit (GRU), or Multilayer Perceptron (MLP). The aim of this paper is to present the nonlinear method for causality analysis and the created Python package.

Methods: The created functions with the autoregressive (AR) and Generalized Radial Basis Functions (GRBF) neural network models were tested on simulated signals in two cases: with nonlinear dependency and with absence of causality from Y to X signal. The train-test split (70/30) was used. Errors obtained on the test set were compared using the Wilcoxon signed-rank test to determine the presence of the causality. For the chosen model, the proposed method of study the change of causality over time was presented.

Results: In the case when X was a polynomial of Y, nonlinear methods were able to detect the causality, while the AR model did not manage to indicate it. The best results (in terms of the prediction accuracy) were obtained for the MLP for the lag of 150 (MSE equal to 0.011, compared to 0.041 and 0.036 for AR and GRBF, respectively). When there was no causality between the signals, none of the proposed and AR models did indicate false causality, while it was detected by GRBF models in one case. Only the proposed models gave the expected results in each of the tested scenarios.

Conclusions: The proposed method appeared to be superior to the compared methods. They were able to detect non-linear causality, make accurate forecasting and not indicate false causality. The created package enables easy usage of neural networks to study the causal relationship between signals. The neural-networks-based approach is a suitable method that allows the detection of a nonlinear causal relationship, which cannot be detected by the classical Granger method. Unlike other similar tools, the package allows for the study of changes in causality over time.

1. Introduction

1.1. Granger causality

The causality concept, which is discussed in this paper was presented by Sir Clive Granger in 1969 [1]. Nowadays it is widely used in economics [2–5] and neuroscience [6–14]. Recently, there has also been a growing interest in Granger causality in the field of physiology, where it can be used for searching the cause of phenomena and even as a physiological marker [15–20]. In this approach, a causal relationship (where the time series Y is the cause of the time series X, denoted by Y→X) occurs if the variance of the prediction error of the Y based on the past of all available information (U) is statistically significantly smaller than the prediction error variance of the X based on the past of all available information except for the time series Y (Eq. (1)). In practice, some defined by researcher number of lag values of X and Y time series are treated as all available information, while past lags of X are treated as all available information except for Y.

$$\sigma^2(X|U) < \sigma^2(X|U-Y) \tag{1}$$

To test hypotheses about causality 2 linear autoregressive models (AR) and covariance stationary time series X and Y are assumed. The first model predicts the current value of time series

[*] Corresponding author.
E-mail address: maciej.rosol.dokt@pw.edu.pl (M. Rosoł).

https://doi.org/10.1016/j.cmpb.2022.106669

X based only on p lagged values of X and Y (Eq. (2)), the second one predicts the current value of Y, based on the same values but with different coefficients (Eq. (3)). In the equations shown below A are the regression coefficients and E are the regression errors.

$$X(t) = \sum_{j=1}^{p} A_{11,j} X(t-j) + \sum_{j=1}^{p} A_{12,j} Y(t-j) + E_1(t) \tag{2}$$

$$Y(t) = \sum_{j=1}^{p} A_{21,j} X(t-j) + \sum_{j=1}^{p} A_{22,j} Y(t-j) + E_2(t) \tag{3}$$

If in the first equation the variance of error E_1 is statistically significantly smaller for the model taking into account the variable Y (coefficients A_{12} different from zero) than the variance of the same model without taking into account the variable Y (coefficients A_{12} equal to zero), it means that the variable Y is the G-cause of the variable X [21]. Similarly, if the variance of the error E_2 in the second equation is smaller to a statistically significant degree for the model including the variable X (coefficients A_{21} other than zero) from the variance of the prediction error for a model in which the variable X was not included (coefficients A_{21} equal to zero), it means that the variable X is the G-cause of the variable Y.

To test if variable Y is G-causing variable X, the F-test can be applied [22]. First, the residual sums of squares (RSS) are calculated for the model making prediction only on past values of X (Eq. (4)) and for the model that uses for prediction past values of X and Y time series (Eq. (5)) [23].

$$RSS_1 = \sum_{t=1}^{T} E_X(t)^2 \tag{4}$$

$$RSS_2 = \sum_{t=1}^{T} E_{X,Y}(t)^2 \tag{5}$$

Based on this, the test statistic for the F-test (S) can be computed according to Eq. (6), where l is equal to the number of considered lags, and T is equal to the number of predicted values [24].

$$S_1 = \frac{(RSS_1 - RSS_2)/l}{RSS_2/(T-2l-1)} \sim F_{l,T-2l-1} \tag{6}$$

The value of the S_1 test statistic is consistent with the Fisher distribution with

l and $(T\text{-}2l\text{-}1)$ degrees of freedom. To test whether Y causes X (Y→X), the F-test is performed under the null hypothesis that Y does not cause X and the alternative hypothesis that Y is causing X.

Also, the Chi-squared test may be applied with test statistics computed according to Eq. (7) [23].

$$S_1 = \frac{T(RRS_1 - RSS_2)}{RSS_2} \sim \chi^2(p) \tag{7}$$

Causality can not only be tested for occurrence but also quantified according to Eq. (8), where $\sigma^2 E_X$ means variance of the error obtained from the model based only on the past values of X and $\sigma^2 E_{X,Y}$ is the variance of the error obtained from the model based on the past of both signals [6,7,25].

$$F_{Y\to X} = ln\frac{\sigma^2 E_X}{\sigma^2 E_{X,Y}} \tag{8}$$

If three time series (X–Z) are available, the mutual Granger causality analysis for two variables for each pair is not able to show some relationships between the data. For example, if Y is the signal causing Z, and Z is the signal causing X, but Y is not the signal causing X, then in the case of a two-variable analysis, this will be indistinguishable from the situation where Y is the signal causing both Z and X. This situation is presented in Fig. 1. With a causality analysis for two time series for both presented cases, the obtained result will suggest the presence of the relationship presented in part A of Fig. 1.

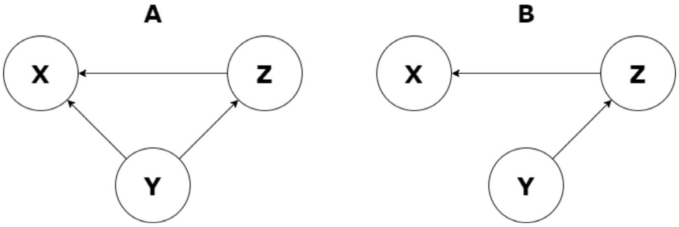

Fig. 1. Two cases of dependencies are indistinguishable for the two-way Granger causality analysis and in both cases, the result of the analysis would suggest the presence of causality presented in Diagram A.

In order to distinguish such situations, a conditional Granger causality test can be used. It is able to indicate whether the past of the Y signal helps to reduce the variance of the prediction error of X predicted from the past of X and Z time series. In the case of conditional causality analysis, the linear autoregressive equations presented in Eq. (2) are extended by the sum of the products of the respective coefficients (A_{13}) and the third variable Z, so that this variable is used for forecasting the present value of X, as presented in Eq. (9).

$$X(t) = \sum_{j=1}^{p} A_{11,j} X(t-j) + \sum_{j=1}^{p} A_{12,j} Y(t-j) + \sum_{j=1}^{p} A_{13,j} Z(t-j) + E_1(t) \tag{9}$$

Similar to the causality analysis for two time series, two models are created in the conditional causality analysis. The first one does not take into account the past of variable Y in the model (coefficients A_{12} equal to 0), the second one takes into account the past of all variables (coefficients A_{12} different from 0). If error E_1 is statistically significantly smaller for the second model it means that Y is causing X conditioned on Z, which is written as Y→X|Z.

1.2. Nonlinear methods for Granger causality analysis

The main limitation of the approach presented by Granger is the usage of the linear model. Due to the fact that many real data turn out to be non-stationary processes, this introduces a large limitation for the above-mentioned methods, which assume the stationarity of the tested time series. It is possible to overcome the issue of non-stationarity of the time series by the usage of vector error correction models however, the modeling is still using linear autoregression [26,27]. The limitation coming from the usage of the AR model is that more complex causality dependencies may not be captured by this method [21,28]. Thus, many researchers use other models for prediction instead of linear autoregression. One of the approaches of nonlinear causality testing is Kernel Granger Causality (KGC) [29,30]. It is based on the transformation of the data using a specified kernel function. The inner product of the data and kernel function is used to perform the linear regression. The nonlinearity of this method is controlled by the choice of the kernel function. KGC can be applicable for multivariate problems, it also allows for quantification of the causality and does not suffer from overfitting problem which is a common issue for many methods [31]. Another approach that is growing in popularity is to use neural networks as a forecasting method [14,32–37]. Proposed in 2017, Causal Relationship Estimation by Artificial Neural Network (CREANN) method uses weights of Multilayer Perceptron to assess the causality of the individual lags [36]. This method is robust with respect to signal-to-noise ratio and model order however, it does not make statistical inference about causality. Another recently proposed method for causality assessment is large-scale nonlinear Granger causality (lsNGC) [14]. In this approach, the Generalized Radial Basis Functions neural network is used as a prediction model. In the experiments conducted by Wismüller et al. [14], this method outperformed other methods compared, including KGC. The lsNGC was implemented in Python and made available to researchers to use [38]. Other existing Python solutions using neural networks to study causality use lasso regularization [39,40]. By using this technique, some input weights are zeroed and those Y lag values whose weights are not equal to zero are considered to be the values that cause the time series X. The results obtained using this approach do not allow for statistical inference using statistical tests and depend on the value of the lambda parameter corresponding to the lasso penalty.

The created Python package presented in this paper was designed to test for causality with precise forecasting thanks to the usage of neural networks. By the usage of dropout and out-of-sample testing, the overfitting problem, which might lead to false causality detection, can be omitted. Moreover, the package allows the quantification of the change in causality over time (which is not enabled by any other available package). Created functions allow for the forecasting using Multilayer Perceptron (MLP) and recurrent neural networks, in particular Long-Short Term Memory (LSTM) and Gated Recurrent Unit (GRU). It also supports the usage of ARIMA model, which is not in the scope of this paper. The approach proposed by the authors does not make the obtained results dependent on an additional parameter or chosen kernel and allows for statistical inference. The package is designed to help scientists use more complex models in terms of Granger causality in an easy user-friendly way without very specific programming knowledge, as well as study causality changes over time, which is not provided by any other framework. It was designed to study the relationship between biological signals, however, it can be widely used in any field of science. The paper aims at presenting the method and corresponding Python package, along with a simulation study showing its usage and relevance.

2. Methods

2.1. Used models

2.1.1. Multilayer perceptron

MLP is a type of artificial neural network, which is built from many single perceptrons organized in layers. MLP are the most popular artificial neural networks used for forecasting, but also in the field of physiology [41,42]. Each perceptron in the first layer takes as an input the features vector and calculates its output as presented in Eq. (23), where *w* are weights and *f* is the activation function.

$$y = f(w_0 + w_1 * x_{t-1} + w_2 * x_{t-2} + \ldots + w_p * x_{t-p}) \tag{23}$$

Neurons in successive layers take the output values of neurons from previous layers *y* as the input vector. In the created package, the activation function in hidden layers is the Rectified Linear Unit (ReLU) function, while in the output layer it is a linear function. The architecture of a simple MLP model based only on past values of X and predicting the current value of X for the Granger causality test is presented in Fig. 2.

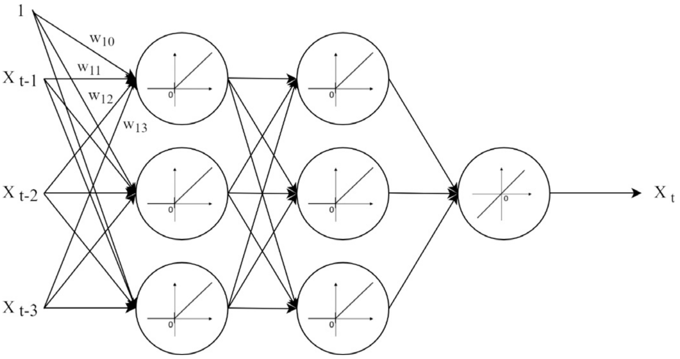

Fig. 2. MLP model which is forecasting the current value of X based on 3 past values of X with activation functions and example weights shown.

2.1.2. Long short-term memory

The LSTM is a type of gated recurrent neural network. It is often used for sequence data analysis like text and speech recognition, time series forecasting, or physiological data analysis [43–45]. The LSTM network includes a system of gate units, thanks to which it is able to control the flow of information, and thus "remember" or "forget" information from previous moments in time. What is more, this type of network is not affected by the gradient vanishing or explosion problem, which is common for recurrent neural networks [46]. The big advantage of using LSTM in causality testing is that, unlike linear regression models, those types of networks do not assume the stationarity of the predicted time series. The LSTM cell takes as input as the input vector *x(t)*, the value of the long-term state from the previous time point *s(t-1)* and the value of the short-term state from the previous time point *h(t-1)*. Instead, it returns the value of the short-term *h(t)* and long-term *s(t)* state at the current moment in time. The first gate is the forget gate, which controls what part of the information from the past is “remembered” and which one is “forgotten”. For this purpose, a sigmoid function is used, and the value passed further from the forget gate is calculated based on Eq. (24).

$$f^{(t)} = \sigma\left(U^f x^{(t)} + W^f h^{(t-1)} + b^f\right) \tag{24}$$

Another important gate is the input gate. It determines the degree of status update at a given moment in time. For this, it uses the sigmoid function to select the values that should be used to update the state at a given moment from the input vector and the output vector for the previous moment of time. The formula describing the operation of the input gate is presented in Eq. (25), where U^i is the gate input weights, W^i is the recursive weights, and b^i is the bias of the input gate. For the state to be updated, the candidate values are also calculated using the hyperbolic tangent for the input vector and the output vector of the previous time moment, taking into account the appropriate weights U^s and W^s and the bias b^s as shown in Eq. (26).

$$i^{(t)} = \sigma\left(U^i x^{(t)} + W^i h^{(t-1)} + b^i\right) \tag{25}$$

$$\tilde{s}^{(t)} = \tanh\left(U^s x^{(t)} + W^s h^{(t-1)} + b^s\right) \tag{26}$$

Updating the LSTM cell state takes place by summing the product of the result obtained on the forget gate $f^{(t)}$ and the state value at the previous time point $s^{(t-1)}$ with the product of the result obtained on the input gate with the candidate values. The formula for updating the state is shown in Eq. (27).

$$s^{(t)} = f^{(t)} s^{(t-1)} + i^{(t)} \tilde{s}^{(t)} \tag{27}$$

The last gate in the LSTM cell is the output gate, the operation of which is described in Eq. (28), where U^o is the gate input weights, W^o is the recursive weights, and b^o is the bias of the output gate. Like the other gates, it uses a sigmoidal function to control which state values at a given point in time will be included in the calculation of the final output of the network.

$$o^{(t)} = \sigma\left(U^o x^{(t)} + W^o h^{(t-1)} + b^o\right) \tag{28}$$

The computation of the output of the LSTM cell is done by multiplying the hyperbolic tangent of the cell's state and the result obtained from the output gate as shown in Eq. (29).

$$h^{(t)} = o^{(t)}\tanh\left(s^{(t)}\right) \tag{29}$$

Thanks to the use of the three above-described gates, the LSTM neural network is able to control the flow of information between consecutive time moments by "remembering" or "forgetting" them. What is more, LSTM networks have greater ease in recognizing long-term dependencies than simple recursive network architectures [47].

2.1.3. Gated recurrent unit

GRU neural network was presented by Cho et al. in 2014 [48]. GRU, like LSTM, is a kind of gated recursive neural network. The GRU network is characterized by the use of two gates - a reset gate and an update one. Such a recursive neural network architecture also prevents gradient vanishing and exploding effects [47]. In the GRU cell, the update gate is a kind of counterpart of the forget gate and input gate in the LSTM cell. The update gate controls which information is “forgotten” and which new information from a given point in time will be taken into account in further calculations. The operation of the update gate is based on calculating the value of the sigmoid function for the input data and the cell result from the previous time moment, taking into account the appropriate weights U^u and W^u and the bias b^u, as shown in Eq. (30) [49].

$$u^{(t)} = \sigma\left(U^u x^{(t)} + W^u h^{(t-1)} + b^u\right) \tag{30}$$

The reset gate, on the other hand, only serves to control the amount of past information that will be used to compute the candidate state [50]. Like the other gates, the reset gate uses a sigmoidal function with appropriate weights U^r and W^r and bias b^r as presented in Eq. (31).

$$r^{(t)} = \sigma\left(U^r x^{(t)} + W^r h^{(t-1)} + b^r\right) \tag{31}$$

After calculating the value of the reset gate, the candidate state value is calculated, which will be used in the next step to calculate the value of the GRU cell state at a given time moment. The candidate state value is calculated using the hyperbolic tangent function, according to Eq. (32), where U^h and W^h are the weights, while b^h corresponds to the bias.

$$\tilde{h}^{(t)} = \tanh\left(U^h x^{(t)} + W^h\left(r^{(t)} h^{(t-1)}\right) + b^h\right) \tag{32}$$

The value of the state h^t at a given point in time is computed using the value of the update gate, the value of the candidate state and the value of the state at the previous point in time, according to Eq. (33).

$$h^{(t)} = \left(1 - u^{(t)}\right)h^{(t-1)} + u^{(t)}\tilde{h}^{(t)} \tag{33}$$

Thanks to the use of a reset gate and an update gate, the GRU network is able to "forget" and "remember" information at successive time points. GRU networks, as well as LSTM networks, are not affected by the gradient vanishing or explosion problem and do not assume the stationarity of the predicted time series. The recursive neural network of the GRU type has similar advantages to the LSTM type network, however, thanks to the use of two gates instead of three, the GRU network has fewer parameters, which simplifies the learning process and reduces computational complexity.

2.2. Statistical significance assessment

The F-test is based on residual sums of squares, which are minimalized by linear autoregression models using the least-squares method. This test cannot be performed in the case of using neural networks for prediction, because those models are fitted using other methods as the “Adam” algorithm and thus value obtained from Eq. (6) for neural networks may not follow the assumed F-distribution [51].

Therefore, we decided to test if the error obtained by the model, which uses both time series for prediction is significantly smaller than the error obtained by the model using only one time series as an input by using Wilcoxon signed-rank test [52].

The null hypothesis is that the median absolute error for a model based on past values of X is equal or smaller than for a model based on past values of X and Y, while the alternative hypothesis is that the model based on past values of both X and Y time series has a smaller median absolute error. In order to examine this hypothesis differences between absolute errors of both models are calculated. Then ranks are assigned to obtained numbers based on their absolute values. The sign of the obtained difference is allocated to the rank and the sum of positive and negative ranks are calculated. In Python implementation, the sum of positive ranks is taken as a statistic value (T) based on which p-value is calculated [53]. If the number of differences (n) is less than or equal to 25 then the p-value is derived from the tables. If n is greater than 25, then a normal approximation is applied. In this case, the test statistic (z) is calculated according to Eq. (34) and follows the normal distribution [54].

$$z = \frac{T - \frac{n(n+1)}{4}}{\sqrt{\frac{n(n+1)(2n+1)}{24}}} \tag{34}$$

2.3. Description of the created package

Taking all considerations together, we created the Python package, which aims to help researchers to study causality using nonlinear prediction methods [55,56]. The use of neural networks for prediction has been implemented using the Keras library [57]. The package consists of 8 functions, which can be split into 2 different types. The first type of functions are simply functions for testing the causality relationship between two time series (with possible additional time series in terms of conditional causality) using the above-mentioned methods and the ARIMA model, which is not in the scope of this paper. The first input of those functions is *NumPy ndarray* with 2 columns, where each column represents one time series, and the number of rows depends on the number of times steps. The second input may be an *int, list, tuple,* or *NumPy ndarray*. If it is an *int*, then the causality test is made for lags in the range from 1 to the given number, in other cases, the test is made only for numbers given in this variable. In functions using neural networks for prediction additional required inputs are describing the architecture of the network. As an output functions of this type return a *dictionary*, where the keys are the number of lags for which the test was performed. Each key stores a *list*, which contains test results, the model for prediction of X fitted only on X time series and the model for prediction of X fitted on X and Y time series, individual errors and RSS of both models, in case of functions using neural networks additional history of fitting the first model and history of fitting the second model are also stored under this key.

The second type of function is focused on measuring the change of causality over time. The first input for those functions is the same as in the first type, but the number of columns may be greater than 2. If so, then causality between each pair of time series (columns) is measured. As those functions measure the change of causality over time two windows were applied. The first *w1* is responsible for the number of time steps from which the causality is calculated, and the second window *w2* relates to the number of time steps by which the window *w1* is moved after the causality is calculated. So the first value of causality is calculated for time moments from 1st to *w1*th, the second value is calculated for

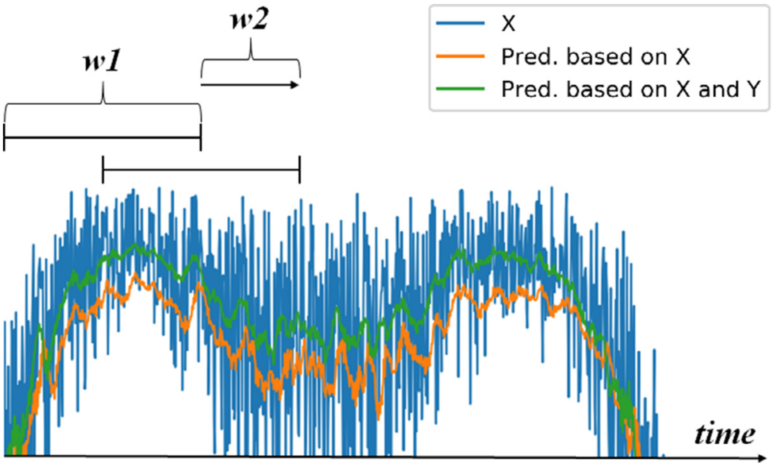

Fig. 3. Visualization of windows used for calculation of change of causality over time.

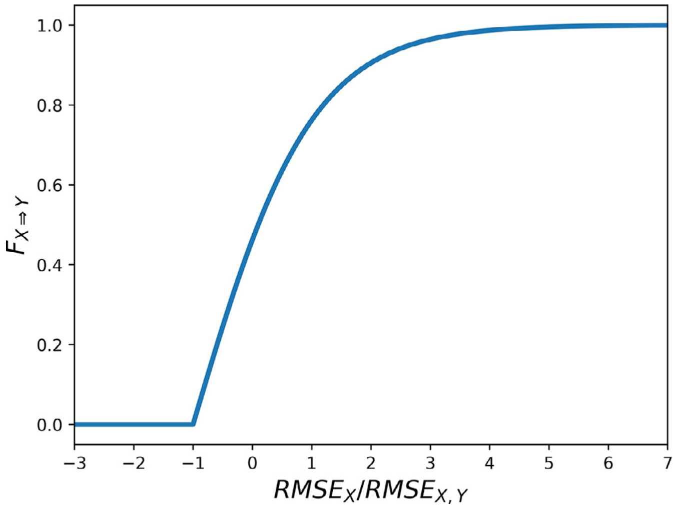

Fig. 4. Graph of proposed causality measure from the quotient of RMSE_X and $\text{RMSE}_{X,Y}$.

time moments from *w2*th to (*w1*+*w2*)th and so on as presented in Fig. 3.

As the measure presented in Eq. (8) is not applicable for the analysis of the change of causality over time, the new measure of this phenomenon was proposed. This measure of causality is based on the root mean square errors (RMSE) obtained from both models from the currently analyzed window *w1*. RMSE used for calculation of causality in the first window is presented in Eqs. (35) and (36).

$$RMSE_X = \sqrt{\sum_{t=1}^{w1} E_X(t)^2 / w1} \tag{35}$$

$$RMSE_{X,Y} = \sqrt{\sum_{t=1}^{w1} E_{X,Y}(t)^2 / w1} \tag{36}$$

The measure proposed by authors is shown in Eq. (37). If the obtained value is smaller than 0 then the result is changed to 0, because a negative number has no sense in terms of causality. The plot of values of the proposed measure in relation to the quotient of RMSE_X and $\text{RMSE}_{X,Y}$ is presented in Fig. 4.

$$F_{Y \to X} = \frac{2}{1 - e^{\left(-\frac{RMSE_X}{RMSE_{X,Y}} + 1\right)}} - 1 \tag{37}$$

3. Results

3.1. Simulated signals

In this paper, we would like to present the usage and possibilities of the created package in terms of detecting the causality by different models in two cases - with the presence of the causality between the signals and with no dependencies between the time series. The results of the causality detection and forecasting accuracy were compared with linear and nonlinear methods. What is more, we would like to present the unique feature of illustrating the change of causality over time using the chosen model. To present the performance of the package two signals with 10,000 samples each were generated, where Y is a periodic function and X is a polynomial function of Y delayed by 100-time steps as presented in Eqs. (38) and (39) and in Listing 1. Some random noise (E_1 and E_2) also was added to both signals in order to make them more real-like. Both X and Y time series were visualized in Fig. 5.

$$Y(t) = \cos(t) + \sin(0.15 * t) + E_1(t) \tag{38}$$

$$X(t) = 2 * Y(t-100)^3 - 5 * Y(t-100)^2 + 0.3 * Y(t-100) + 2 + E_2(t) \tag{39}$$

The first 70% of the signals were used as training data and the remaining 30% was used as a test dataset. The signals prepared in this way can be used for the causality analysis using the first kind of functions included in the developed module. It was decided to perform a causality test for two lags - smaller and greater than the actual delay. The lags were equal to 50 and 150.

Using the created Python module (version 1.0.3), three types of neural networks with the following architectures were applied for causality analysis:

- Two LSTM layers with 10 cells each (LSTM) followed by one output neuron with linear activation function;
- Two GRU layers with 10 cells each (GRU) followed by one output neuron with linear activation function;
- Two fully connected layers with 100 neurons each (MLP) and ReLU activation function followed by one output neuron with linear activation function.

For each network dropout regularization technique was used with a dropout rate equal to 0.01 (arguments *NN_config* and *NN_neurons* for MLP and *Dropout_rate* in case of GRU and LSTM). Neural networks were trained for 150 epochs, with a learning rate

```
y = np.cos(np.linspace(0,20,10100)) + np.sin(np.linspace(0,3,10100)) - 0.2*np.random.random(10100)
x = 2*y**3 - 5* y**2 + 0.3*y + 2 - 0.05*np.random.random(10100)
```

Listing 1. Generation of two signals.

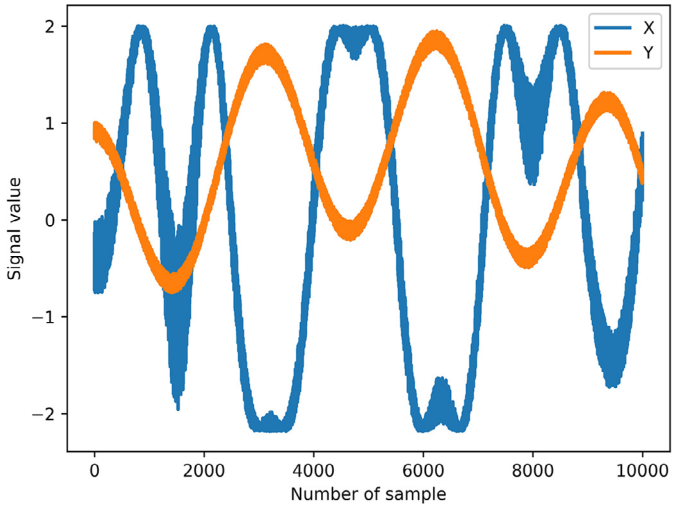

Fig. 5. Visualization of signals X and Y.

equal to 0.001 and 0.0001 for the first 50 epochs and the last 100 epochs, respectively. To obtain the most accurate models LSTM and GRU neural network models were created and trained 2 times (parameter *run*) and the MLP model 5 times. For final causality testing models (one based on X and one based on both X and Y) with the smallest RSS were chosen. The difference in the value of the *run* parameter was due to the long time needed for LSTM and GRU training. The usage of the created functions is presented in Listing 2.

Regarding the detection of causality and the prediction performance, the presented methods were compared with models used in two other methods for causality testing, both of which allow for out-of-sample forecasting and are as well implemented in Python. The first of these methods was a nonmodified Granger test, which uses linear autoregression (AR) for prediction. The second one was large-scale nonlinear Granger causality (lsNGC) [14,38], which is based on a Generalized Radial Basis Functions (GRBF) neural network. Its parameters c_f and c_g, which are the number of hidden layer neurons in the GRBF networks were set to 25. All models were fitted on the training set and used on the test set. The absolute values of the prediction errors obtained on a test set from a model based on the past of X and a model based on the past of both signals were compared using Wilcoxon signed-rank test, to determine the presence of causality. Performance of all methods was quantified using mean squared error (MSE), mean absolute error (MAE) and median absolute error (MedAE). Moreover, it was assessed if the usage of the proposed results in improvement of the prediction accuracy models in comparison to AR or GRBF network. For this purpose, Wilcoxon signed-rank test was used to compare the absolute values of errors obtained from corresponding models (e.g. MLP model based on past of X and AR model based also on a past of X). The effect size of including the past of signal Y in prediction for each method was assessed using Cohen's d. In order to better visualize the results plots of predicted values versus the true values of X and plots of prediction error versus the predicted X values were prepared for each model. The steps of the entire process of causality analysis were presented in Fig. 6.

In the next step of the analysis, the Y time series was replaced with the random noise to compare the results from the tests in the case where the causality relation should be detected with the case where there is no relation between the signals. This analysis was performed as well using the first 70% of the signal as the training set and the last 30% as the test set. The same metrics and tests were performed as described in the previous paragraph.

The second type of function in the package can be used to examine the change of causality over time. In order to simulate such a change of causality relation the first 50% of the test Y signal was changed to random noise (Fig. 7), so there would be no causal relation between X and Y at the first half of the test time series. To present this feature of the package the MLP architecture which obtained the highest Cohen's d was chosen. Mentioned in Section 2.2. Window values were set to 30 and 1 for *w1* and *w2*, respectively. The example usage of the function for assessment of the change of causality over time was presented in Listing 3.

The assumed significance level is equal to 0.05. All analyses were performed using Python version 3.7.10.

3.2. Presence of causality from Y to X

Each of the designed functions generates a plot of original testing signal X, values predicted by the model based only on past values of X and values predicted by the model based on past values of X and Y. For each function based on neural networks, it is possible to obtain a history of the fitting from the output of the function and create the learning curve. Sample plots of the original and predicted data obtained from function `nonlincausalityNN` for lag equal to 50 and 150 are presented in Fig. 8 while learning curves of the models returned by this function are presented in Fig. 9.

All error metrics calculated on a testing set for each model for lag equal to 50 and 150 are presented in Table 1. *P*-values obtained from the Wilcoxon signed-rank test used to assess the presence of causality from Y to X for each method and each lag are presented in Table 2.

In case of lag equal to 50, all models from a created package (NN) obtained similar results in terms of error metrics. The AR models had a similar accuracy of prediction, while GRBF models tend to obtain the biggest error metrics for model based only on the past values of X and the smallest one in the case of model based on both signals, while model based on both time series obtained slightly smaller error metrics than LSTM, GRU and AR models. For the lag of 150, NN and AR mostly obtained similar results, except for MLP model based on X and Y which outperformed all other models and got the smallest error in the case of all three metrics. In the case of the bigger lag GRBF model had a drop in performance for the model based only on X signal. In the case of the causality test, all nonlinear methods obtained a p-value smaller than the assumed significance level, thus in the case of using any of the presented models or GRBF the causality relationship between signals was detected, even for lag smaller, than the actual delay between the time series. The autoregressive model used in the state-of-the-art Granger method did not capture the causality for any given lag.

P-values obtained from the Wilcoxon signed-rank test used to assed the improvement in prediction due to usage of neural networks over autoregressive and GRBF models are presented in Table 3 and Table 4, respectively. Cohens'd used to assess the effect size of incorporating the past of Y signal into the models are presented in Table 5. Plots of predicted values from actual values are presented in Figs. 10, and 11. for lag 50 and 150, respectively and

```
lags = [50,150]

results_LSTM = nlc.nonlincausalityLSTM(data_train,maxlag=lags,xtest=data_test,LSTM_layers=2,
                                       LSTM_neurons=[10,10],run=2,epochs_num=[50,100],
                                       learning_rate=[0.001,0.0001],batch_size_num=128,
                                       Dropout_rate=0.01, plot=True,verbose=False)

results_GRU = nlc.nonlincausalityGRU(data_train,maxlag=lags,xtest=data_test,GRU_layers=2,
                                     GRU_neurons=[10,10],run=2,epochs_num=[50,100],
                                     learning_rate=[0.001,0.0001],batch_size_num=128,plot=True,
                                     Dropout_rate=0.01, verbose=False)

results_NN =
nlc.nonlincausalityNN(data_train,maxlag=lags,xtest=data_test,NN_config=['d','dr','d','dr'],
                      NN_neurons=[100,0.01,100,0.01],run=5,epochs_num=[50,100],
                      learning_rate=[0.001,0.0001], batch_size_num=128, plot=True,
                      verbose=False)
```

Listing 2. Using the functions for causality testing.

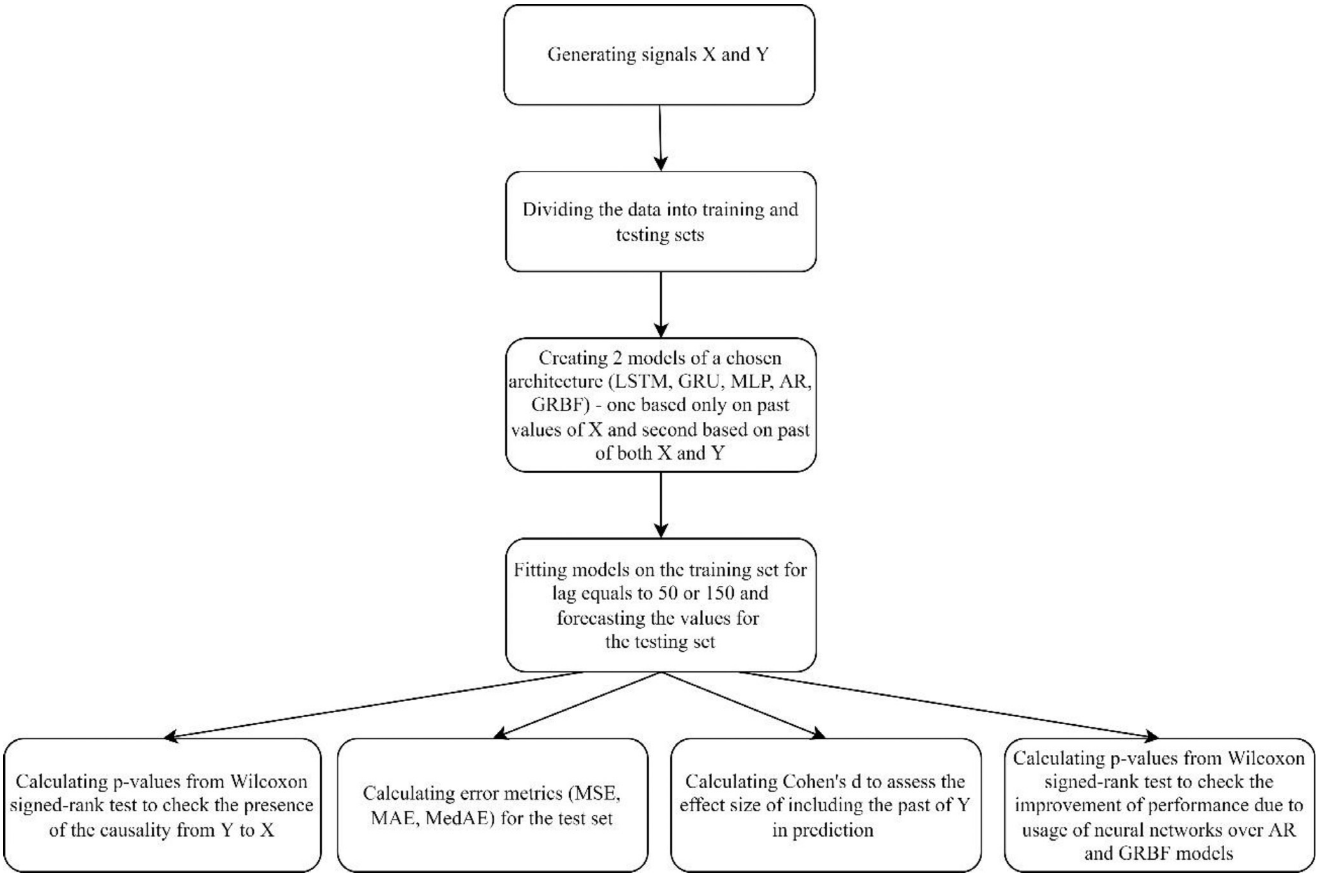
Generating signals X and Y
Dividing the data into training and testing sets
Creating 2 models of a chosen architecture (LSTM, GRU, MLP, AR, GRBF) - one based only on past values of X and second based on past of both X and Y
Fitting models on the training set for lag equals to 50 or 150 and forecasting the values for the testing set
Calculating p-values from Wilcoxon signed-rank test to check the presence of the causality from Y to X
Calculating error metrics (MSE, MAE, MedAE) for the test set
Calculating Cohen's d to assess the effect size of including the past of Y in prediction
Calculating p-values from Wilcoxon signed-rank test to check the improvement of performance due to usage of neural networks over AR and GRBF models

Fig. 6. The process scheme of the data analysis from generating the data up to the causality and prediction assessment. In the case of real-world application, the process would be the same, but X and Y would be the data obtained from the study.

plots of the prediction error from predicted values are presented in Figs. 12 and 13.

The usage of neural networks over AR did not statistically significantly improve the prediction of the X signal if taking only past values of X as an input. On the other hand, the usage of NN over GRBF with one signal as an input always results in significantly better prediction accuracy. If the model was based on both X and Y signals the usage of neural networks instead of the autoregressive model improved the prediction in 5 out of 6 cases (only the MLP model for 50 lags did not obtain significantly better results). In the case of GRBF, only MLP model for lag equal to 150 outperformed it. For both lags, the effect size of incorporating past of Y

```
res_meas_NN = nlc.nonlincausalitymeasureNN(data_train, lags, w1=30,w2=1, xtest=data_test_measure,
NN_config=['d','dr','d','dr'], NN_neurons=[100,0.01,100,0.01], run=5, epochs_num=[50,100],
learning_rate=[0.001,0.0001], batch_size_num = 128, verbose = False, plot = True,
plot_with_xtest=True)
```

Listing 3. Usage of the function for measuring the causality change over time with MLP models.

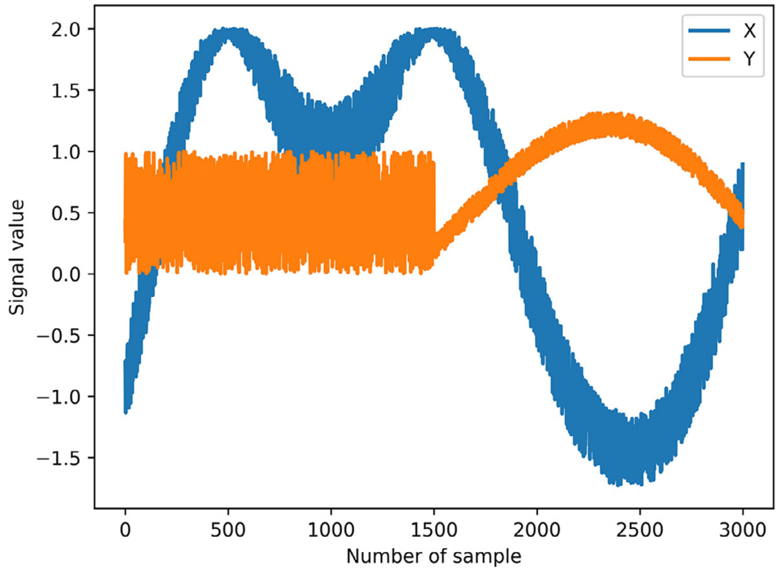

Fig. 7. Visualization of test signals X and Y with the first 50% of the test Y signal changed to random noise.

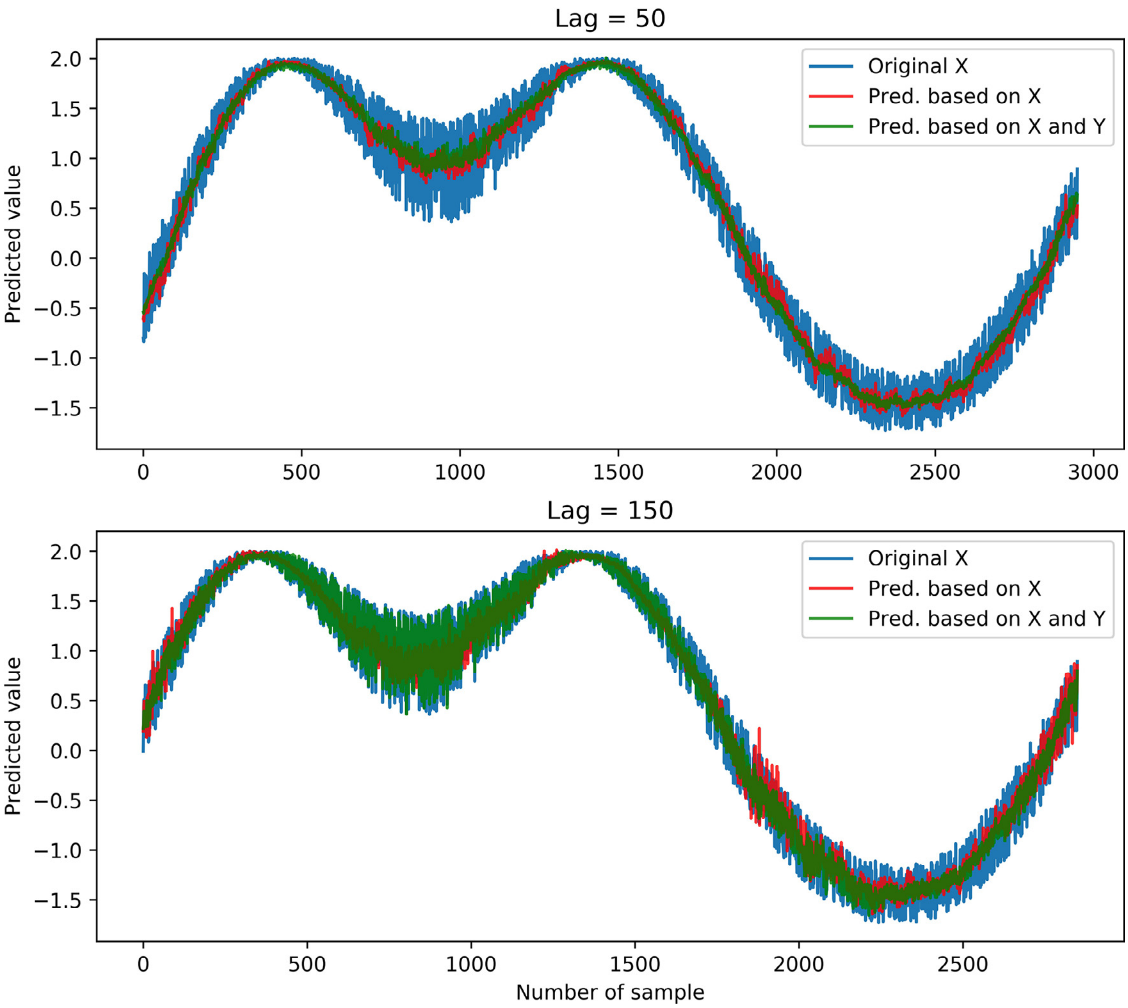

Fig. 8. The plot of the original value of X, predicted values from the model based on the past values of X and predicted values from the model based on past values of X and Y. If there is causality from Y to X, then the values predicted based on both signals should be closer to the actual values than the values predicted only with the past X values (prediction error of model based on both signals should be lower, then the error of model based only on X).

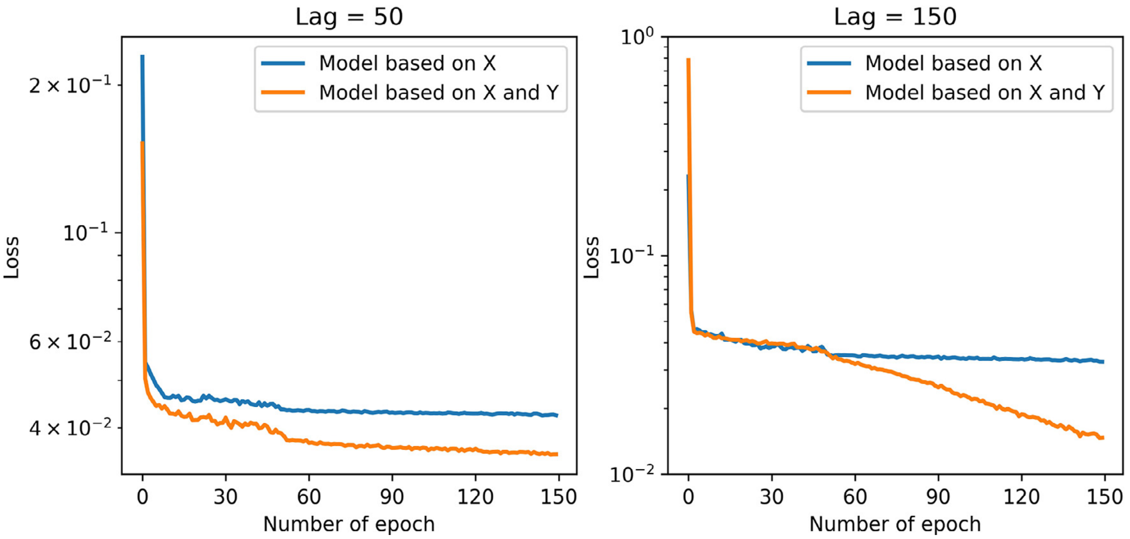

Fig. 9. Plot presenting the dependencies between loss and number of epochs for lag equal to 50 and 150.

Table 1
Error metrics (MSE, MAE and MedAE) obtained on a test set for each model created based on 50 and 150 past values in the case where Y→X. The smallest error metric for a given case is shown in bold.

Lag value	Metrics	Model trained on	LSTM	GRU	MLP	AR	GRBF
50	MSE	X	0.042	**0.041**	0.042	**0.041**	0.057
		X and Y	0.039	0.039	0.039	0.041	**0.038**
	MAE	X	0.162	**0.161**	0.162	**0.161**	0.188
		X and Y	0.158	0.158	0.161	0.161	**0.155**
	MedAE	X	0.139	0.138	**0.136**	**0.136**	0.153
		X and Y	0.137	**0.135**	0.137	0.140	**0.135**
150	MSE	X	**0.041**	**0.041**	0.045	**0.041**	0.254
		X and Y	0.039	0.038	**0.011**	0.041	0.036
	MAE	X	**0.159**	0.161	0.166	0.160	0.413
		X and Y	0.156	0.156	**0.078**	0.160	0.151
	MedAE	X	0.135	0.136	**0.132**	0.137	0.379
		X and Y	0.134	0.134	**0.056**	0.136	0.130

Table 2
P-values for each model and each tested lag obtained from the Wilcoxon signed-rank test in case where Y→X. Cases, where a causal relationship has been detected, are shown in bold.

Lag value	LSTM	GRU	MLP	AR	GRBF
50	**< 0.001**	**< 0.001**	**0.025**	0.703	**< 0.001**
150	**< 0.001**	**< 0.001**	**< 0.001**	0.261	**< 0.001**

Table 3
P-values form a comparison of the models performances, with the alternative hypothesis that error obtained from neural networks models is smaller than for autoregressive ones in a case where Y→X. Cases, where usage of the neural network resulted in significantly better performance, are shown in bold.

Lag value	Model trained on	LSTM	GRU	MLP
50	X	0.969	0.899	0.944
	X and Y	**< 0.001**	**< 0.001**	0.068
150	X	0.227	0.814	0.991
	X and Y	**0.010**	**0.008**	**< 0.001**

signal into prediction was the highest for MLP and the smallest for AR. The biggest Cohen's d equal to 0.659 was obtained for MLP for lag equal to 150. Prediction error was equally distributed around 0 for the whole range of predicted values.

Table 4
P-values form a comparison of the model performances, with the alternative hypothesis that absolute error obtained from neural networks models is smaller than for GRBF ones in a case where Y→X. Cases, where usage of the neural network resulted in significantly better performance, are shown in bold.

Lag value	Model trained on	LSTM	GRU	MLP
50	X	**< 0.001**	**< 0.001**	**< 0.001**
	X and Y	0.991	0.998	1.000
150	X	**< 0.001**	**< 0.001**	**< 0.001**
	X and Y	1.000	1.000	**< 0.001**

Table 5
Cohens'd for the case where Y→X. The highest Cohens'd for each lag are shown in bold.

Lag value	LSTM	GRU	MLP	AR	GRBF
50	0.047	0.046	**0.049**	0.005	0.013
150	0.037	0.044	**0.659**	0.001	0.054

3.3. *Absence of causality from Y to X*

The metrics calculated for the analysis, where there was no relationship between the time series (with the Y changed to random noise) are presented in Table 6. The results from the Wilcoxon signed-rank test used to test for the presence of causality are presented in Table 7.

For the lag of 50 neural networks obtained the same or higher error measure values than autoregressive models. For lag equal to

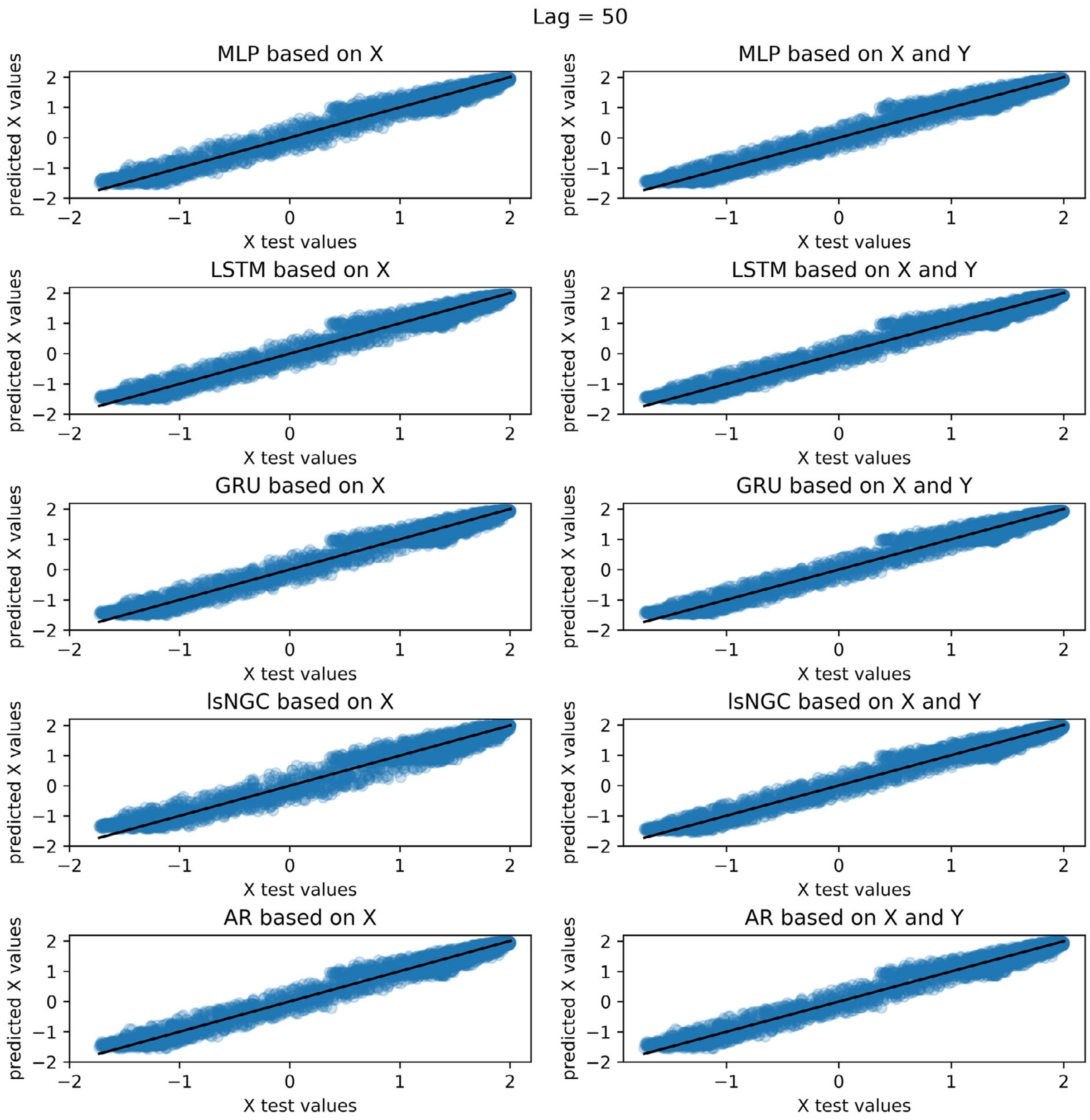

Fig. 10. The plot of predicted values from true X values for lag equals 50. The solid black line represents y equals to x.

Table 6
Error metrics (MSE, MAE and MedAE) obtained on a test set for each model created based on 50 and 150 past values in the case where there was no causality relation between signals. The smallest error metric for a given case is shown in bold.

Lag value	Measure	Model trained on	LSTM	GRU	MLP	AR	GRBF
50	MSE	X	0.042	**0.041**	0.042	**0.041**	0.055
		X and Y	0.042	**0.041**	0.042	**0.041**	0.055
	MAE	X	0.162	**0.161**	0.163	**0.161**	0.184
		X and Y	0.162	**0.161**	0.164	**0.161**	0.184
	MedAE	X	0.138	0.137	**0.136**	**0.136**	0.152
		X and Y	0.138	0.137	0.138	**0.135**	0.150
150	MSE	X	**0.041**	**0.041**	0.045	**0.041**	0.278
		X and Y	**0.041**	**0.041**	0.049	**0.041**	0.276
	MAE	X	0.161	**0.159**	0.168	0.160	0.443
		X and Y	**0.160**	**0.160**	0.173	0.162	0.442
	MedAE	X	0.137	**0.134**	0.135	0.137	0.417
		X and Y	0.137	**0.136**	0.140	0.137	0.411

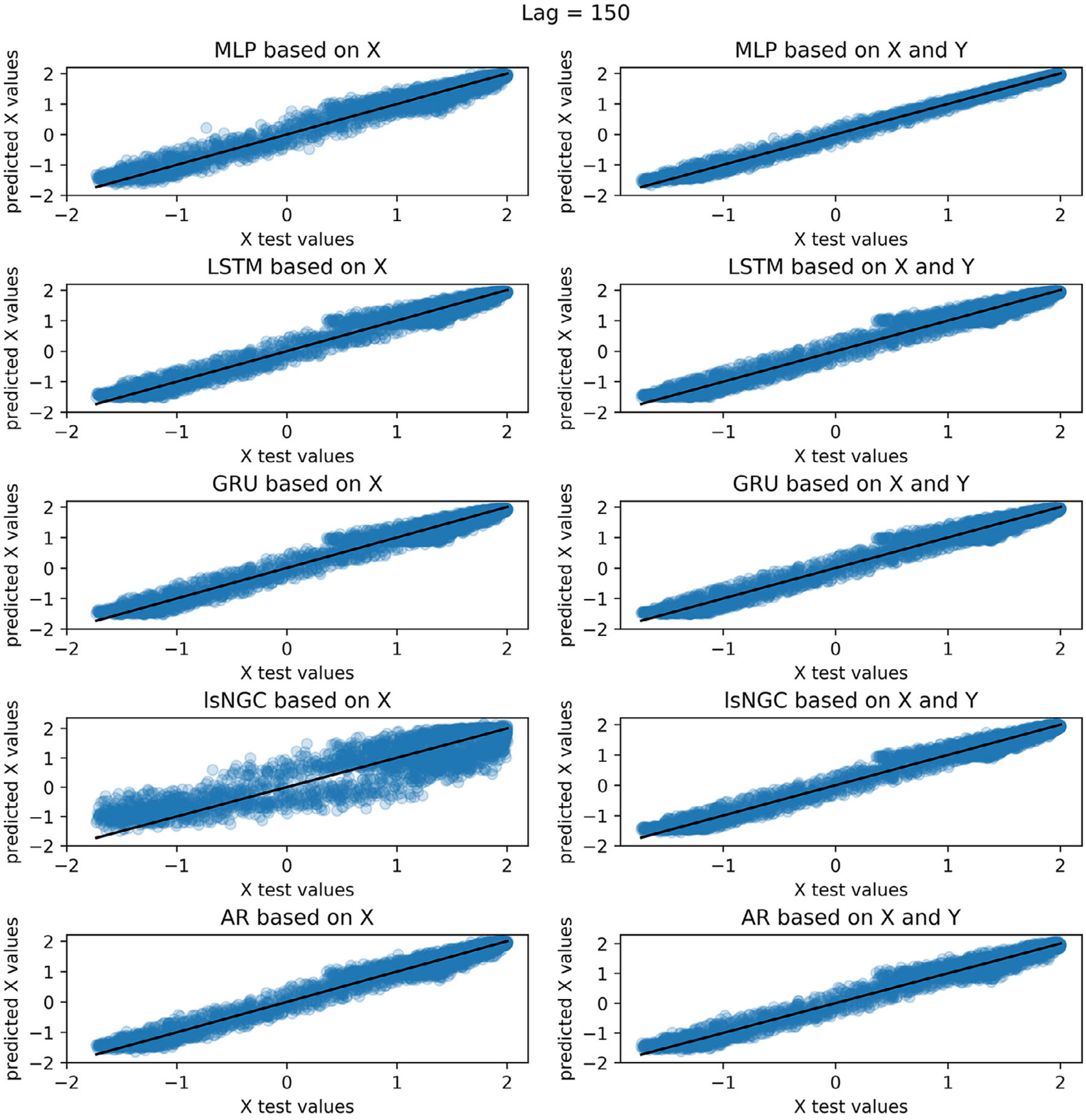

Fig. 11. The plot of predicted values from true X values for lag equals 150. The solid black line represents y equals to x.

Table 7
P-values for each model and each tested lag obtained from the Wilcoxon signed-rank test in the case where there was no causality relation between signals. Cases, where a causal relationship has been detected, are shown in bold.

Lag value	LSTM	GRU	MLP	AR	GRBF
50	0.558	0.267	0.930	1.000	0.127
150	0.073	0.592	0.999	1.000	**0.024**

Table 8
P-values from a comparison of the model performances, with the alternative hypothesis that error obtained from neural networks models is smaller than for autoregressive ones in the case where there was no causality relation between signals. Cases, where usage of the neural network resulted in significantly better performance, are shown in bold.

Lag value	Model trained on	LSTM	GRU	MLP
50	X	0.982	0.941	0.999
	X and Y	0.857	0.486	1.000
150	X	0.607	0.219	1.000
	X and Y	**0.041**	**0.038**	1.000

150 the measures were mostly higher for MLP, mostly lower for GRU and mostly equal to those obtained by AR for LSTM. The GRBF compared to other models obtained slightly higher error metrics for lag equal to 50 and several times larger for lag equal to 150. Causality was not detected using any of the models except the GRBF model for a delay of 150, where false causality was indicated.

P-values obtained from the Wilcoxon signed-rank test used to assed the improvement in prediction due to usage of neural networks over autoregressive and lsNGC models are presented in Table 8 and Table 9, respectively. Cohens'd used to assess the effect size of incorporating the past of Y signal (random noise in this case) into the model are presented in Table 10.

The only models which performed statistically significantly better than AR were GRU and LSTM models based on both X and Y for lag equal to 150. All NN models performed better than GRBF models in all cases. The highest Cohens'd was obtained again for MLP models, but all values were much smaller compared to those in the case where Y→X.

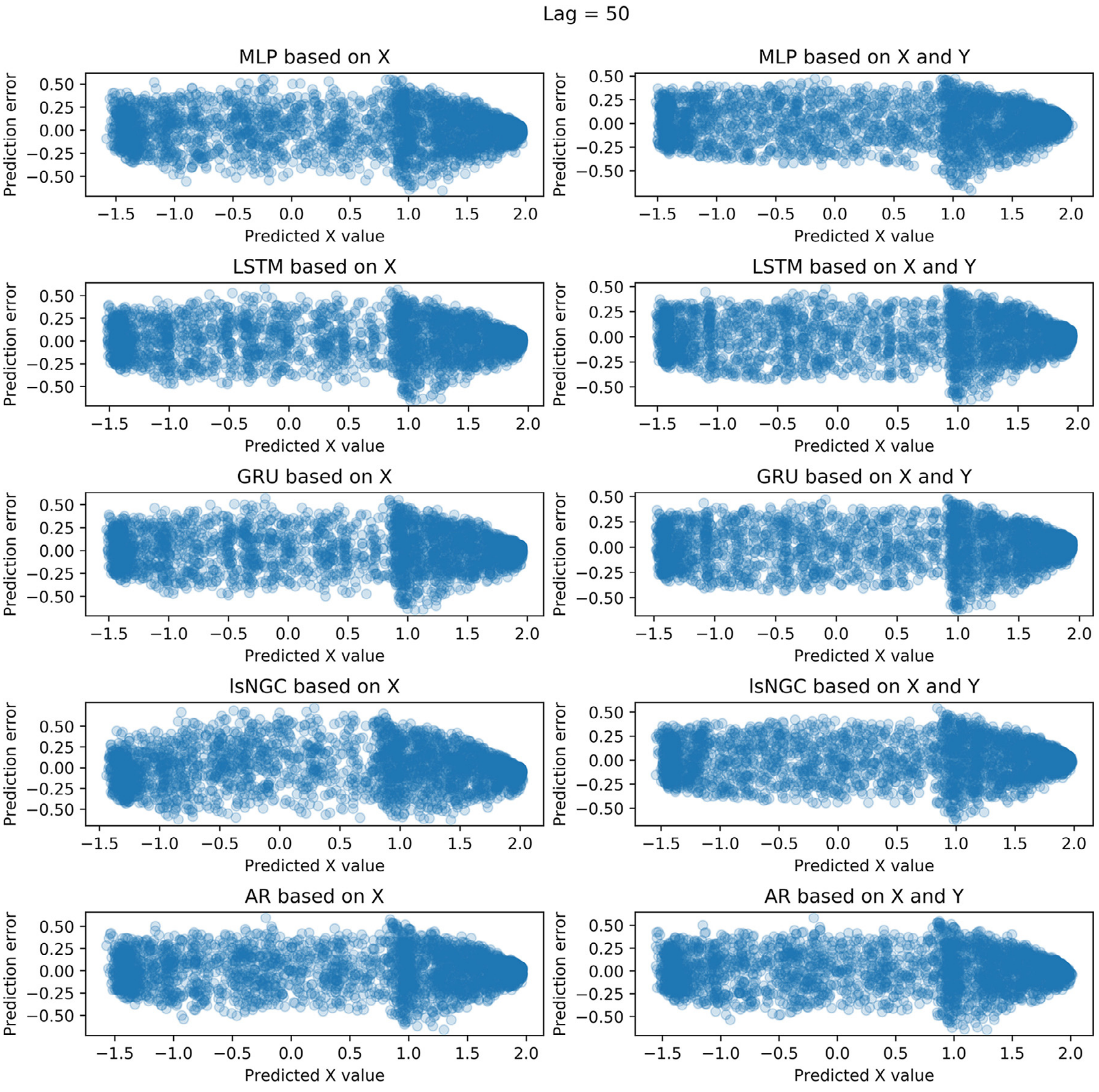

Fig. 12. The plot of prediction errors from predicted X values for lag equals 50.

Table 9
P-values form a comparison of the model performances, with the alternative hypothesis that error obtained from neural networks models is smaller than for GRBF ones in the case where there was no causality relation between signals. Cases, where usage of the neural network resulted in significantly better performance, are shown in bold.

Lag value	Model trained on	LSTM	GRU	MLP
50	X	**< 0.001**	**< 0.001**	**< 0.001**
	X and Y	**< 0.001**	**< 0.001**	**< 0.001**
150	X	**< 0.001**	**< 0.001**	**< 0.001**
	X and Y	**< 0.001**	**< 0.001**	**< 0.001**

Table 10
Cohens'd for the case where there was no causality relation between signals. The highest Cohens'd for each lag are shown in bold.

Lag value	LSTM	GRU	MLP	AR	GRBF
50	< 0.001	< 0.001	**0.004**	< 0.001	< 0.001
100	< 0.001	< 0.001	**0.005**	< 0.001	< 0.001

3.4. Change of the causality over time

The analysis of the change of causality over time was performed using the same MLP architecture as for causality testing which managed to obtain the biggest Cohen's d in the analysis. The measure of causality according to Eq. (37) and its change over time along with the original time series for better visualization of the dependency for 50 lags was presented in Fig. 14 and for 150 lags in Fig. 15.

For lag equal to 150 when the causality relation from signal Y to signal X appears (after the random noise) there is a visible increase in the measure of causality for Y→X. After the random noise the measure for Y→X tends to be much higher, than for X→Y (as there is no such causality). In case of lag equal to 50 in part of the plot where there is the random noise, the measures are similar for both X→Y and Y→X, while after this part the measure for Y→X seems to be slightly bigger than for X→Y (but smaller than for the lag of 150). In the middle of the signals for both lags, there is an increase in causality from X to Y, which means that in that part the model was able to benefit from using the X signal in forecasting the Y in terms of decreasing the prediction error. The maximum of the causality measure for the lag of 50 is equal to 0.324 and 0.139

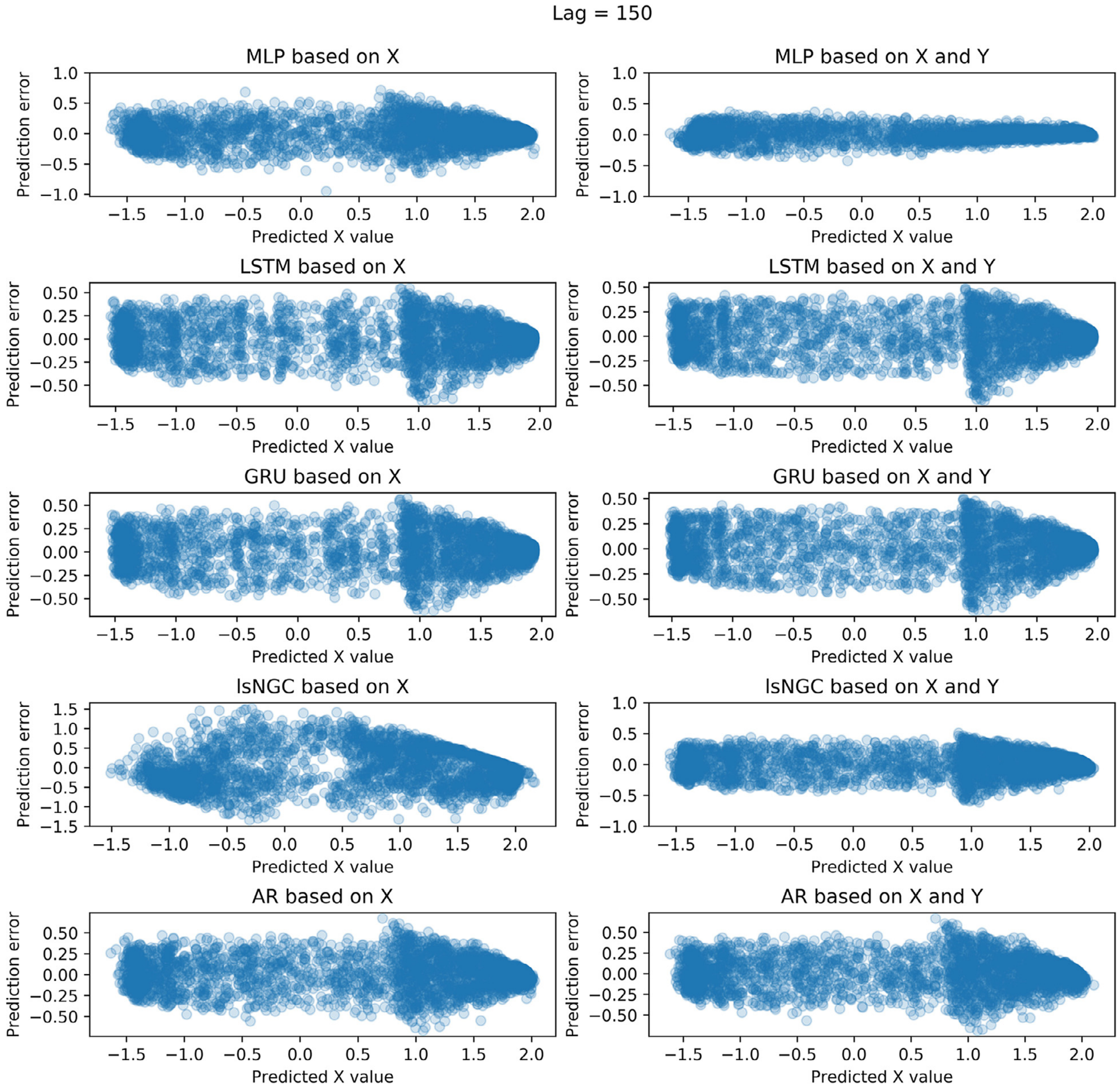

Fig. 13. The plot of prediction errors from predicted X values for lag equals 150.

for X→Y and Y→X, respectively and for lag equal to 150 those values were equal to 0.318 and 0.620. The mean value of the causality for lag equal to 50 was equal to 0.009 and 0.014 for X→Y and Y→X, respectively, while for lag equal to 150 those values were equal to 0.007 for X→Y and 0.128 for Y→X. However, it should be emphasized that the mean and maximum values of the causality measure are showing the gain from incorporating the other variable into prediction and those values cannot be interpreted as a presence of causality between signals on their own. As showed in the previous sections the assessment of the presence of causality is performed using Wilcoxon signed-rank test.

4. Discussion

The methods presented in this paper are focused on overcoming the main disadvantages of the non-modified Granger method based on linear autoregressive models and proposing an alternative for existing nonlinear methods. The first weakness of the linear Granger causality test is that of using AR models the state-of-the-art Granger approach may not capture the more complex, nonlinear causality dependencies like polynomial, exponential, logarithmic, or others that are visible in many biomedical, physiological, economic and social measurements/signals. What is more, the AR models are assuming the stationarity of the signals, which is problematic as most real-life signals are not stationary (however this problem might be also addressed by usage of vector error correction model [26,27], but this approach is still assuming the linearity of the dependency and is not implemented in the existing Python packages). Thanks to the usage of neural networks models it is possible not only to detect the causal relations, which are nonlinear but also to test the dependencies between nonstationary time series and obtain very accurate forecasting results. In order to test the proposed approach and the created Python package, two time series were simulated, where the signal X was a polynomial of the Y signal and was delayed in relation to it by 100-time steps. The data analysis was performed in 2 different ways. In the first approach, both models (based on the past of X and the past of X and Y) were trained on the first 70% of the data and testing for causal-

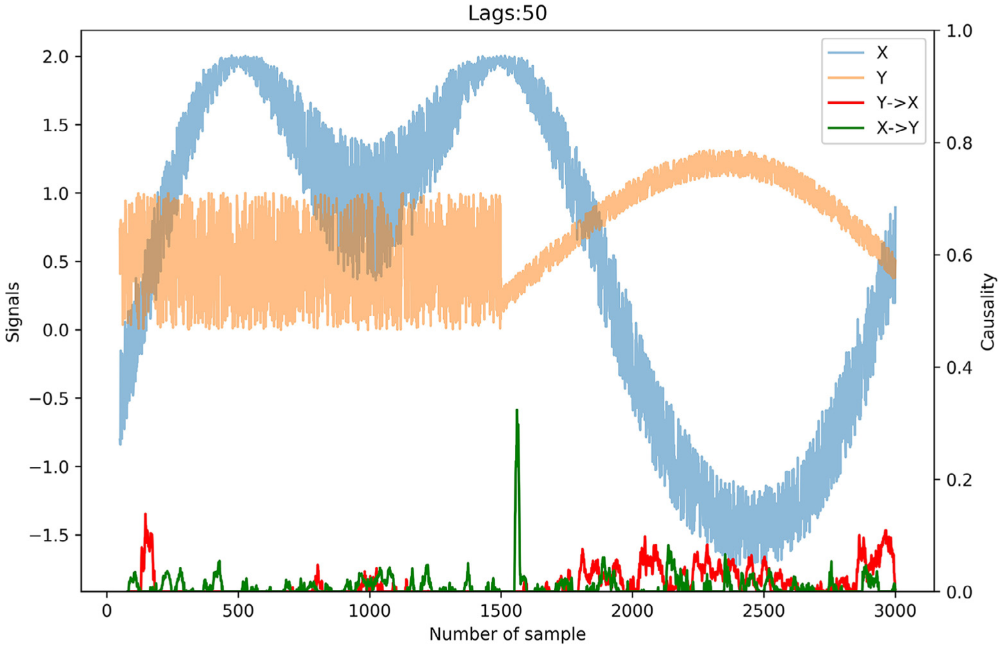

Fig. 14. The measure of causality (Eq. (37)) from signal Y to signal X (Y→X) and from signal X to Y (X→Y), along with signal X and Y for analysis for 50 lags. The axis on the left represents the signal values, and the axis on the right represents the causality value.

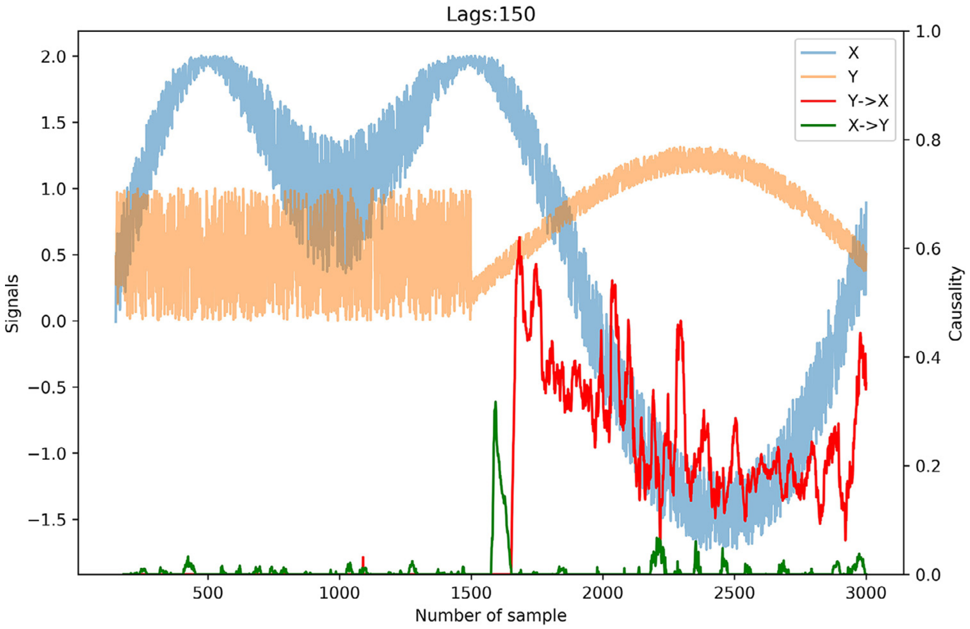

Fig. 15. The measure of causality (Eq. (37)) from signal Y to signal X (Y→X) and from signal X to Y (X→Y), along with signal X and Y for analysis for 150 lags. The axis on the left represents the signal values, and the axis on the right represents the causality value.

ity on the remaining 30% to check the robustness of the methods. The second analyses were similar to the first one, but the Y signal was changed to a random noise so there was no causality relation between X and Y. This change was made to test if the proposed methods do not indicate the false causality relations. The proposed models were compared with the autoregressive model used in the traditional Granger method and with the Generalized Radial Basis Functions neural network used in lsNGC approach [14].

The created functions based on neural networks were able to detect the causality relation between time series for lag both smaller and greater, than the actual delay between the two signals. The GRBF models also indicated the presence of true causality, unlike the AR ones, which were not able to detect this dependency. The biggest difference between the results for lag bigger and smaller than the actual delay between X and Y signals were the p-value, Cohens'd and error metrics for MLP models. The p-value was many orders of magnitude smaller for the lag equal to 150, while Cohen's d was much higher for this lag value. The MLP model based on both X and Y obtained much smaller error metrics in case of the lag equal to 150 compared to the results for lag equal to 50. In the case of lag equal to 150, there was a large drop in the precision of GRBF model based on the past of X compared to the same model but for a smaller lag value. In the case of other models, the performance was rather similar irrespective of the value of the lag. The plot of predicted value from the actual value was much more centered around the line y equals to x for

MLP model using both signals for lag equal to 150, which also indicates that the most accurate prediction was for this model. In the case of GRBF model using only X signal for lag equal to 150, the points are the least concentrated around the mentioned line, which also confirms its lowest accuracy. When it comes to the plot of the prediction error from the predicted values, the MLP model based on X and Y for lag equals 150 is distinguished, where the error at higher predicted values is much smaller than in the case of other models, while for GRBF based on X there is visible more dispersion of the points, which indicates the lowest prediction performance. The best results in terms of Cohen's d and error metrics were obtained for MLP for lag equal to 150 for models based on both X and Y signals. The usage of NN instead of AR mostly resulted in more accurate results in the case of models based on both signals, while compared to GRBF usage of NN was always more efficient for models based only on the past of X.

When Y time series was changed to a random noise to test if proposed methods do not indicate the false dependency when there is no causal relation between examined time series, all neural networks obtained from the created package and AR modes did not detect any causality between the signals for any lag value. In the case of GRBF the false causality was detected for lag equal to 150. Error metrics and Cohen's d obtained by NN and AR models were similar to each other, while for GRBF the prediction accuracy was the lowest especially in case of lag equal to 150.

Compared to the autoregressive models used in traditional Granger causality analysis the proposed approach was characterized by better or similar prediction performance and discovery of nonlinear causality relations undetected by the AR models. In the case of comparison with the nonlinear approach using GRBF the proposed solution was always superior in prediction based on the past of only one variable and did not indicate any false causality, which appeared to be the issue in one case for GRBF models. The advantage of the proposed method over other approaches using non-linear data transforms such as KGC is the lack of assumptions about the data transformation (kernel used). The prepared Python package is superior to other existing solutions that use neural networks with lasso regularization to study causality [39,40] as it allows for statistical inference and the results are independent of the lambda parameter used in the lasso regularization. As the neural networks applied in this research are already used in the analysis of physiological data [41,42,45], the created package can also find wide application in the study of causal relationships in physiology, but also other scientific fields.

To our knowledge, a novelty of the created package is the proposed method of studying the change in causality over time, which may allow for a better understanding of the causal relationships between the signals. For both lags, the causality values seem to be bigger for the part of the signal without random noise. As expected, the plots of the change of causality over time show much more causal dependency for the lag equal to 150, than for lag equal to 50. For the higher lag, there is very clearly visible the moment when the causality relationship appears. The value of the causality measure varies over time, probably due to the added noise to the signal and to the out-of-sample testing. This feature of the package can be very useful especially in the case of signals which dependence varies over time.

The limitations of the prepared package are the computational complexity and time-consuming involved in training neural networks. However, the benefits of using neural networks seem to outweigh these issues. The limitation of the study is the use of simulated data, so further studies on real-world data are planned. In the near future, we plan to focus on using the created package to investigate the causal relationships between biomedical signals and their dependence on various vital parameters as a continuation of research conducted by Młyńczak and Krysztofiak [16,17]. We plan, among others, to use the package in the analysis of the causality between the respiratory signal from impedance pneumography (tidal volume equivalent) and the cardiological signal from the ECG (mainly RR-intervals and tachogram as their interpolation) and to investigate the causality phenomenon in various groups of patients (network physiology paradigm [58]).

5. Conclusion

The usage of neural networks in causality testing allows capturing the nonlinear causal dependencies, which are not detected by the AR model used in the state-of-the-art Granger method. In the case when there is no causal relation neural-network-based methods do not indicate false causality, which might be an issue while using GRBF model for prediction. Usage of neural networks allowed to provide better prediction results especially in the case of multilayer perceptron taking as an input past values of both time series for a lag value greater than the actual delay. This model obtained the highest effect size of incorporating the past of Y signal into the prediction model. The measure of the change of causality over time seems to be a valid feature that allows to better understand the detected dependencies between the signals. The created package can be widely used (is available in PyPI [56]) in the analysis of signals in different scientific fields like neuroscience, physiology, or economy. Thanks to the proposed method, it is possible to study nonlinear dependencies, study causality changes over time, and unlike similar nonlinear approaches, it is easily usable thanks to the package created in Python.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This research was not financially supported by any institution or organization. No ethical approval was required. The authors declare that they have no conflict of interests.

References

[1] C.W.J. Granger, Investigating causal relations by econometric models and cross-spectral methods, Econometrica 37 (1969) 424, doi:10.2307/1912791.

[2] R.P. Maradana, R.P. Pradhan, S. Dash, D.B. Zaki, K. Gaurav, M. Jayakumar, A.K. Sarangi, Innovation and economic growth in European economic area countries: the Granger causality approach, IIMB Manag. Rev. (2019), doi:10.1016/j.iimb.2019.03.002.

[3] M.O. Appiah, Investigating the multivariate Granger causality between energy consumption, economic growth and CO_2 emissions in Ghana, Energy Policy (2018), doi:10.1016/j.enpol.2017.10.017.

[4] V. Troster, M. Shahbaz, G.S. Uddin, Renewable energy, oil prices, and economic activity: a Granger-causality in quantiles analysis, Energy Econ. (2018), doi:10.1016/j.eneco.2018.01.029.

[5] R.P. Pradhan, M.B. Arvin, S. Bahmani, Are innovation and financial development causative factors in economic growth? Evidence from a panel Granger causality test, Technol. Forecast. Soc. Change (2018), doi:10.1016/j.techfore.2018.01.024.

[6] S. Hu, Y. Cao, J. Zhang, W. Kong, K. Yang, Y. Zhang, X. Li, More discussions for granger causality and new causality measures, Cogn. Neurodyn. 6 (2012) 33–42, doi:10.1007/s11571-011-9175-8.

[7] M. Ding, Y. Chen, S.L. Bressler, Granger causality: basic theory and application to neuroscience, in: Handbook of Time Series Analysis: Recent Theoretical Developments and Applications, Wiley-VCH Verlag GmbH & Co. KGaA, 2006, pp. 437–460.

[8] S. Cekic, D. Grandjean, O. Renaud, Time, frequency, and time-varying Granger-causality measures in neuroscience, Stat. Med. (2018), doi:10.1002/sim.7621.

[9] F. Azarmi, S.N. Miri Ashtiani, A. Shalbaf, H. Behnam, M.R. Daliri, Granger causality analysis in combination with directed network measures for classification of MS patients and healthy controls using task-related fMRI, Comput. Biol. Med. (2019), doi:10.1016/j.compbiomed.2019.103495.

[10] Z. Abbasvandi, A.M. Nasrabadi, A self-organized recurrent neural network for estimating the effective connectivity and its application to EEG data, Comput. Biol. Med. (2019), doi:10.1016/j.compbiomed.2019.05.012.

[11] M.G. Tana, R. Sclocco, A.M. Bianchi, GMAC: a Matlab toolbox for spectral Granger causality analysis of fMRI data, Comput. Biol. Med. (2012), doi:10.1016/j.compbiomed.2012.07.003.

[12] R.M. Demirer, M.S. Özerdem, C. Bayrak, E. Mendi, Determination of ECoG information flow activity based on Granger causality and Hilbert transformation, Comput. Methods Programs Biomed. (2013), doi:10.1016/j.cmpb.2013.08.011.

[13] Y. Gao, X. Wang, T. Potter, J. Zhang, Y. Zhang, Single-trial EEG emotion recognition using Granger causality/transfer entropy analysis, J. Neurosci. Methods (2020), doi:10.1016/j.jneumeth.2020.108904.

[14] A. Wismüller, A.M. Dsouza, M.A. Vosoughi, A. Abidin, Large-scale nonlinear Granger causality for inferring directed dependence from short multivariate time-series data, Sci. Rep. (2021), doi:10.1038/s41598-021-87316-6.

[15] L. Faes, G. Nollo, A. Porta, Non-uniform multivariate embedding to assess the information transfer in cardiovascular and cardiorespiratory variability series, Comput. Biol. Med. (2012), doi:10.1016/j.compbiomed.2011.02.007.

[16] M. Młyńczak, H. Krysztofiak, Discovery of causal paths in cardiorespiratory parameters: a time-independent approach in elite athletes, Front. Physiol. (2018), doi:10.3389/fphys.2018.01455.

[17] M. Młyńczak, H. Krysztofiak, Cardiorespiratory temporal causal links and the differences by sport or lack thereof, Front. Physiol. (2019), doi:10.3389/fphys.2019.00045.

[18] A.D. Orjuela-Cañón, A. Cerquera, J.A. Freund, G. Juliá-Serdá, A.G. Ravelo-García, Sleep apnea: tracking effects of a first session of CPAP therapy by means of Granger causality, Comput. Methods Programs Biomed. (2020), doi:10.1016/j.cmpb.2019.105235.

[19] A.D. Jaimes-Albarracin, A.D. Orjuela-Canon, A.L. Jutinico, M.A. Bazurto, E. Duenas, Brain and heart physiological networks analysis employing neural networks granger causality, in: Proceedings of the 10th International IEEE/EMBS Conference on Neural Engineering (NER), 2021, doi:10.1109/NER49283.2021.9441375.

[20] C. Corbier, F. Chouchou, F. Roche, J.C. Barthélémy, V. Pichot, Causal analyses to study autonomic regulation during acute head-out water immersion, head-down tilt and supine position, Exp. Physiol. (2020), doi:10.1113/EP088640.

[21] A. Seth, Granger causality, Scholarpedia 2 (2007) 1667, doi:10.4249/scholarpedia.1667.

[22] C.A. Sims, Money, income, and causality, Am. Econ. Rev. 62 (1972) 540–552, doi:10.1126/science.151.3712.867-a.

[23] SAS, Granger causality test SAS, (2021). https://support.sas.com/rnd/app/ets/examples/granger/index.htm (accessed December 11, 2021).

[24] S.L. Bressler, A.K. Seth, Wiener-Granger causality: a well established methodology, Neuroimage 58 (2011) 323–329, doi:10.1016/j.neuroimage.2010.02.059.

[25] J. Geweke, Measurement of linear dependence and feedback between multiple time series, J. Am. Stat. Assoc. 77 (1982) 304–313, doi:10.1080/01621459.1982.10477803.

[26] A.E. Obayelu, A.S. Salau, Agricultural response to prices and exchange rate in Nigeria: application of co-integration and vector error correction model (VECM), J. Agric. Sci. (2010), doi:10.1080/09766898.2010.11884656.

[27] F.F.A.H. Asari, N.S. Baharuddin, N. Jusoh, Z. Mohamad, N. Shamsudin, K. Jusoff, A vector error correction model (VECM) approach in explaining the relationship between interest rate and inflation towards exchange rate volatility in Malaysia, World Appl. Sci. J. 12 (2011) 49–56.

[28] N. Ancona, D. Marinazzo, S. Stramaglia, Radial basis function approach to nonlinear Granger causality of time series, Phys. Rev. E Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top. (2004) 70, doi:10.1103/PhysRevE.70.056221.

[29] D. Marinazzo, M. Pellicoro, S. Stramaglia, Kernel method for nonlinear Granger causality, Phys. Rev. Lett. (2008), doi:10.1103/PhysRevLett.100.144103.

[30] D. Marinazzo, M. Pellicoro, S. Stramaglia, Kernel-Granger causality and the analysis of dynamical networks, Phys. Rev. E Stat. Nonlinear Soft Matter Phys. (2008), doi:10.1103/PhysRevE.77.056215.

[31] N. Nicolaou, T.G. Constandinou, A nonlinear causality estimator based on nonparametric multiplicative regression, Front. Neuroinform. (2016), doi:10.3389/fninf.2016.00019.

[32] A. Montalto, S. Stramaglia, L. Faes, G. Tessitore, R. Prevete, D. Marinazzo, Neural networks with non-uniform embedding and explicit validation phase to assess Granger causality, Neural Netw. 71 (2015) 159–171, doi:10.1016/j.neunet.2015.08.003.

[33] A. Attanasio, U. Triacca, Detecting human influence on climate using neural networks based Granger causality, Theor. Appl. Climatol. 103 (2011) 103–107, doi:10.1007/s00704-010-0285-8.

[34] T. Li, G. Li, T. Xue, J. Zhang, Analyzing brain connectivity in the mutual regulation of emotion-movement using bidirectional Granger causality, Front. Neurosci. 14 (2020) 369, doi:10.3389/fnins.2020.00369.

[35] Y. Huang, Z. Fu, C.L.E. Franzke, Detecting causality from time series in a machine learning framework, Chaos 30 (2020) 063116, doi:10.1063/5.0007670.

[36] N. Talebi, A.M. Nasrabadi, I. Mohammad-Rezazadeh, Estimation of effective connectivity using multi-layer perceptron artificial neural network, Cogn. Neurodyn. (2018), doi:10.1007/s11571-017-9453-1.

[37] N. Talebi, A.M. Nasrabadi, I. Mohammad-Rezazadeh, R. Coben, NCREANN: nonlinear causal relationship estimation by artificial neural network; applied for autism connectivity study, IEEE Trans. Med. Imaging (2019), doi:10.1109/TMI.2019.2916233.

[38] Large-Scale Nonlinear Granger Causality, 2021. https://github.com/Large-scale-causality-inference/Large-scale-nonlinear-causality (accessed December 11, 2021).

[39] R. Marcinkevičs, Đ. Miladinović, Granger-causal inference in time series for identifying molecular fingerprints during sleep. https://github.com/i6092467/NNGC-SLIMMBA (accessed December 11, 2021).

[40] A. Tank, I. Covert, N. Foti, A. Shojaie, E. Fox, Neural Granger Causality, IEEE Trans Pattern Anal Mach Intell (2021), doi:10.1109/TPAMI.2021.3065601.

[41] S. Kaushik, A. Choudhury, P.K. Sheron, N. Dasgupta, S. Natarajan, L.A. Pickett, V. Dutt, AI in healthcare: time-series forecasting using statistical, neural, and ensemble architectures, Front. Big Data (2020), doi:10.3389/fdata.2020.00004.

[42] K.M. Dalmeida, G.L. Masala, HRV features as viable physiological markers for stress detection using wearable devices, Sensors (2021), doi:10.3390/s21082873.

[43] Y. Yu, X. Si, C. Hu, J. Zhang, A review of recurrent neural networks: LSTM cells and network architectures, Neural Comput. (2019), doi:10.1162/neco_a_01199.

[44] J. Cao, Z. Li, J. Li, Financial time series forecasting model based on CEEMDAN and LSTM, Phys. A Stat. Mech. Appl. (2019), doi:10.1016/j.physa.2018.11.061.

[45] T. Umematsu, A. Sano, R.W. Picard, Daytime data and LSTM can forecast tomorrow's stress, health, and happiness, in: Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), 2019, doi:10.1109/EMBC.2019.8856862.

[46] S. Xiang, Y. Qin, C. Zhu, Y. Wang, H. Chen, Long short-term memory neural network with weight amplification and its application into gear remaining useful life prediction, Eng. Appl. Artif. Intell. 91 (2020) 103587, doi:10.1016/j.engappai.2020.103587.

[47] I. Goodfellow, Y. Bengio, A. Courville, Deep Learning, MIT Press, 2016.

[48] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, Y. Bengio, Learning phrase representations using RNN encoder-decoder for statistical machine translation, in: Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), 2014, doi:10.3115/v1/d14-1179.

[49] J. Chung, C. Gulcehre, K. Cho, Y. Bengio, Gated feedback recurrent neural networks, in: Proceedings of the 32nd International Conference on ICML, 2015.

[50] J.C. Heck, F.M. Salem, Simplified minimal gated unit variations for recurrent neural networks, in: Proceedings of the IEEE 60th International Midwest Symposium on Circuits and Systems (MWSCAS), 2017, doi:10.1109/MWSCAS.2017.8053242.

[51] R. Lund, Time series analysis and its applications: with R examples, 2007. 10.1198/jasa.2007.s209.

[52] F. Wilcoxon, Individual comparisons by ranking methods, Biom. Bull. 1 (1945) 80–83, doi:10.2307/3001968.

[53] The Wilcoxon Signed-Rank Test - Python implementation, (2021). https://github.com/scipy/scipy/blob/v1.6.0/scipy/stats/morestats.py#L2809-L3059 (accessed December 11, 2021).

[54] R.H. Riffenburgh, Statistics in Medicine, Elsevier Inc., 2006, doi:10.1016/B978-0-12-088770-5.X5036-9.

[55] M. Rosoł, Nonlincausality, Python package - github, (2021). https://github.com/mrosol/Nonlincausality (accessed December 11, 2021).

[56] M. Rosoł, Nonlincausality, Python package - PyPI, (2021). https://pypi.org/project/nonlincausality/ (accessed December 11, 2021).

[57] F. Chollet & others, Keras library, (2015). https://keras.io/ (accessed December 11, 2021).

[58] P.C. Ivanov, R.P. Bartsch, Network physiology: mapping interactions between networks of physiologic networks, Underst. Complex Syst. (2014), doi:10.1007/978-3-319-03518-5_10.