Nonlinear Time Series Prediction, ARIMA vs. Neural Networks

Till Ehrich

March 14, 2021

Abstract. This thesis aims to illustrate the usage of Neural Networks in the prediction of nonlinear time series and their performance compared to traditional time series methods. Underlying theory is briefly layed out, with emphasis on detecting nonlinearity in time series with appropriate testing. Furthermore, ARIMA models, as well as Neural Networks are described and their forecasting accuracy is compared based on U.S. housing starts data. As a result, no clear dominance of one model type over the other can be established.

Contents

List of Tables

List of Figures

Abbreviations

ACF	Autocorrelation Function
ADF	Augmented Dickey-Fuller Test
AIC	Akaike Info Criterion
AR	Autoregressive
ARCH	Autoregressive Conditional Heteroscedasticity
ARIMA	Autoregressive Integrated Moving Average
DGP	Data Generating Process
GARCH	Generalized Autoregressive Conditional Heteroscedasticity
LM	Lagrange Multiplier
LSTM	Long Short Term Memory
MA	Moving Average
MLP	Multilayer Perceptron
MSE	Mean Squared Error
NN	Neural Network
PACF	Partial Autocorrelation Function
RNN	Recurrent Neural Network
w.r.t	with respect to

1 Introduction

Modern time series analysis is characterized by an abundance of both data and possible modeling algorithms used for inference assessment and prediction. The latter range from traditional statistical approaches like ARIMA, GARCH or Exponential Smoothing to modern Machine Learning methods like Penalized Regression, Regression Trees or Deep Learning. Brief overviews over available methods as well as empirical comparisons are given in Ahmed et al. (2010); Makridakis, Spiliotis, and Assimakopoulos (2018); Clark et al. (2020). Machine Learning methods have gained popularity in recent years (Makridakis, Spiliotis, and Assimakopoulos, 2018, p. 1) and appeal through their ability to map complex functional forms without imposing restrictions that are inherent to traditional statistical methods (Hill, O'Connor, and Remus, 1996, p. 1083), but also show their limitations in their Black-Box nature (Hastie, Tibshirani, and Friedman, 2009, p. 351, 352) and mixed success in empirical comparisons (Ahmed et al., 2010, p. 594),(Makridakis, Spiliotis, and Assimakopoulos, 2018, p. 1). Neural Networks as representatives of Deep Learning have gained particular popularity, with recent econometric applications in Bucci (2019); Šestanović and Arnerić (2021).
Aimed at providing an intuition about the forecasting accuracy of Neural Networks for nonlinear time series, as well as laying out the underlying theoretical groundwork, the rest of this thesis is structured as follows:
Chapter 2 gives a brief intuition about linear time series models and relevant assumptions. Chapter 3 elaborates on the subject of nonlinearity in time series and designated test procedures. Chapter 4 gives a brief introduction to Neural Networks. Chapter 5 analyses U.S. housing data with emphasis on detecting nonlinearity in the series. Chapter 6 compares the forecasting accuracy of ARIMA and NN models, chapter 7 sums up the results.

2 Linear Time Series Modeling

Linear models still lay the groundwork of modern time series analysis today. They are the first resort for any time series practitioner and the benchmark for all alternative modeling approaches. This chapter is based on Lütkepohl and Krätzig (2004), who give a detailed introduction into econometric time series analysis. Given the fact that an extensive overview is out of the scope of this thesis, only the most important concepts will be revised.

2.1 Stationarity

The basic assumption of stationarity involves two conditions of a time series process $\{y_t\}_{t\in T}$, constant mean of all members of a stationary process, and time invariance of the variance of the process:

$$E(y_t) = \mu_y \forall\ t \in T \tag{1}$$

$$E[(y_t - \mu_y)(y_{t-h} - \mu_y)] = \gamma_h \; \forall \; t \in T \text{ and all integers } h, \; (t-h) \in T \tag{2}$$

Failure to account for the absence of these properties results in biased model estimates, and model parameters can't be tested using traditional tests. Also the model will exhibit poor predictive performance. An adequate test for the presence of non-stationarity is the ADF Test, described in (Lütkepohl and Krätzig, 2004, chap. 2.7.1). Here it is important to point out that the test equation of the ADF test is not ideally equipped to handle nonlinear DGPs, as the test equation has a linear structure. However, it has been shown that the ADF test still has reasonable power against nonlinear alternatives (Demetrescu and Kruse, 2013, p. 40), (Corradi, Swanson, and White, 2000, p. 54) and might be used even when nonlinearity is present. Demetrescu and Kruse (2013); Corradi, Swanson, and White (2000) also propose and discuss alternative tests with nonlinear alternatives, which lay outside the scope of this thesis.

2.2 ARIMA Models

ARIMA models have been the workhorse of linear time series models for a long time. They will serve as the baseline for the comparison of forecasts between linear and nonlinear methods. The methodology briefly described here follows Lütkepohl and Krätzig (2004, chap. 2.3.3).
A general ARMA(p, q) model can be written as follows:

$$y_t = \alpha_1 y_{t-1} + ... + \alpha_p y_{t-p} + u_t + m_1 u_{t-1} + ... + m_q u_{t-q} \tag{3}$$

with $u_t \sim i.i.d \; WN(0, \sigma^2)$. This assumption is also called the $i.i.d$ assumption, which will be subject of the diagnostic tests discussed in chapter 3, as violations of this assumption lead to inconsistent model estimates and poor forecasting capabilities. This model is integrated by oder d, if differencing the data d times is required in order to achieve stationarity, which yields the general ARIMA(p, d, q) model. There is a multitude of procedures at hand to determine the optimal orders of the model, with one of the most popular being described in Hyndman and Khandakar (2008, chap. 3). It is most feasible in this use case as it is tried and tested, and can be implemented automatically.

3 Nonlinearity in Time Series

The first question at hand is: What is nonlinearity? And the answer is as simple as it is unsatisfying - everything that is not linear. More precisely it means this means all the dynamics, that a linear model is unable to capture. These end up in the residual of a model, and here a violation of the $i.i.d$ assumption mentioned in chapter 2.2 can be attributed to nonlinearity. Nonlinear dependence can occur in moments of any order in the process, though here, the focus lies on dependencies in first and second order moments, namely nonlinearity in mean and conditional heteroscedasticity, with special emphasis on modeling the nonlinear mean process. As mentioned in Lee, White, and Granger (1993), a process might still be linear

in mean if it exhibits conditional heteroscedasticity. Therefore possible conditional heteroscedasticity needs to be tested and accounted for, before testing for violations of the $i.i.d$ assumption to detect nonlinearity in mean.

3.1 Conditional Heteroscedasticity

A proven way to detect conditional heteroscedasticity in a residual series is the ARCH-LM test developed by Engle (1982). Following Lütkepohl and Krätzig (2004, chap. 2.6.2, p. 46), it is conducted by taking the squared residual series of a model (e.g. the ARIMA Model from chapter 2.2) and regressing those on lagged values of itself. This leads to the following test regression:

$$\hat{u}_t^2 = \beta_0 + \beta_1 \hat{u}_{t-1}^2 + ... + \beta_q \hat{u}_{t-q}^2 + \epsilon_t \tag{4}$$

where ϵ_t is an error term. Under the null hypothesis, all coefficients of lagged squared errors ranging from β_1 to β_q are zero, with the alternative hypothesis that one or more of these coefficients is different from zero. The resulting test statistic is the R^2 of regression 4, which has an asymptotic $\chi^2(q)$ distribution under the null hypothesis. For choosing the appropriate lag length for the test equation, the autocorrelations of the squared residuals can be consulted. It is important to mention, that the ARCH-LM Test is also prone to falsely rejecting the null of no ARCH effects, if nonlinearity is present in the mean of the series (Blake and Kapetanios, 2003, p. 2). Here it is assumed that after properly accounting for present conditional heteroskedasticity, a rejection of the null on the remaining series can be traced back to present nonlinearity in the mean.

3.2 Modeling Conditional Heteroscedasticity

When it comes to modeling conditional heteroscedasticity, one of the standard models is the GARCH model based on the works by Engle (1982), and Bollerslev (1986), which is still one of the cornerstones of econometric analysis today (Sánchez García and Cruz Rambaud, 2020, p. 1). The modeling is carried out on the residuals of a preceding mean model, such as the ARIMA model described in chapter 2.2. The derivation of the univariate GARCH model follows Lütkepohl and Krätzig (2004, chap. 5.2). A general GARCH(q, p) process u_t is defined as follows:

$$u_t = \xi_t \sigma_t, \xi_t \sim i.i.d.\ N(0, 1) \tag{5}$$

$$\sigma_t^2 = \gamma_0 + \gamma_1 u_{t-1}^2 + ... + \gamma_q u_{t-q}^2 + \beta_1 \sigma_{t-1}^2 + ... + \beta_p \sigma_{t-p}^2 \tag{6}$$

with $\gamma_0 > 0, \gamma_i, \beta_j \geq 0, \forall i = 1, ..., q, j = 1, ..., p$, in order for all conditional variances σ_t^2 to be positive. Standardized residuals can be obtained by dividing the residuals by the root of the fitted variance,

$$\tilde{u_t} = \frac{u_t}{\hat{\sigma_t}} \tag{7}$$

which are free of heteroscedasticity according to equation 5, if the conditional variances are fitted accurately.

3.3 Testing For Independence

To test for the violation of the *i.i.d* assumption mentioned in chapter 2.2, the BDS Test introduced by Brock et al. (Brock et al., 1996) can be used. There are many alternative tests to use in different scenarios. Bisaglia and Gerolimetto (2014) delivered an extensive comparison of different tests for nonlinearity, with the BDS test achieving excellent results and being recommended as a starting point when testing for nonlinearity (Bisaglia and Gerolimetto, 2014, p. 14). Another important property of this test is that is has an unspecified DGP under the alternative hypothesis, which means that it does not indicate which kind of nonlinearity is present in the time series (Bisaglia and Gerolimetto, 2014, p. 7, 14). In this application, this characteristic is negligible, as the model class presented in chapter 4 is flexible in adapting to any kind of functional form. The derivation of the test follows Bisaglia and Gerolimetto (2014, p. 8, 9).
Given a time series $u_t,\ t = 1, 2, ..., T$ the m-history of this time series at time t is given as $U_t^m = (u_t, u_{t-1}, ..., u_{t-m+1})$, the m lagged values of u_t starting at t. This allows the construction of $T_m = T - m + 1$ series of m-histories from the time series u_t. Now the point of interest is, whether the maximum deviation of a combination of these series, $sup||U_t^m - U_s^m||$, with $sup||x|| = max_{1\leq k\leq m}|x_k|$ for $t < s$ does not exceed a threshold ϵ. If this is the case, the indicator function $I_\epsilon(U_t^m, U_s^m)$ is 1, and 0 otherwise. By summing up the indicator function over all $T_m(T_m - 1)/2$ combinations of m-histories, for which $t < s$, and dividing by the number of combinations one receives the share of combinations of m-histories, for which the maximum deviation lies beneath the threshold ϵ. This is the Correlation Integral $C_{m,T}(\epsilon)$.

$$C_{m,T}(\epsilon) = \frac{2}{T_m(T_m - 1)} \sum_{t<s} I_\epsilon(U_t^m, U_s^m) \tag{8}$$

In other words, the correlation integral estimates the joint probability, that any two of the m-histories have a maximum deviation smaller than ϵ, $P(|U_t - U_s| < \epsilon, |U_{t-1} - U_{s-1}| < \epsilon, ..., |U_{t-m+1} - U_{s-m+1}| < \epsilon)$. If the U_t are *i.i.d*, this Probability should be equal to the following in the limiting case:

$$C_{1,T}(\epsilon)^m = P(|U_t - U_s| < \epsilon)^m \tag{9}$$

Which is the product of the disjoint probabilities of a combination of m-histories having a maximum deviation smaller than ϵ. This leads to the following BDS test statistic:

$$V_{m\epsilon} = \sqrt{T}\frac{C_{m,T}(\epsilon) - C_{1,T}(\epsilon)^m}{s_{m,T}} \tag{10}$$

with $s_{m,T}$ denoting the standard deviation, which can be estimated consistently, and the test statistic converging to $N(0, 1)$ according to (Brock et al., 1996, p. 205). Referring back to the test setup at the beginning of this chapter, this test will be carried out using the residuals of a linear model, where conditional heteroscedasticity has been accounted for if present. If the null hypothesis of *i.i.d* distribution is violated for these residuals, this indicates that the underlying time series is nonlinear. Caporale

et al. (2004, p. 283) suggest to carry out the test on the logarithm of the squared residuals when applied to residuals of a GARCH model, in order to make it nuisance-parameter free and improve performance. Caporale et al. (2004, p. 289) also suggest values to use for $\frac{\epsilon}{\sigma}$, where σ is the standard deviation of the GARCH residuals.

4 Neural Networks

Neural Networks have long been subject of discussion in time series analysis and forecasting, as they overcome the limitations of traditional, linear models (Hill, O'Connor, and Remus, 1996, p. 1083). They're inherently nonlinear and are able to approximate any kind of functional form (Funahashi, 1989)[1], which makes them ideal candidates for settings where assumptions necessary for linear time series models are violated. The derivation follows Hastie, Tibshirani, and Friedman (2009, chap. 11.3). There is a multitude of architectures available for NNs. Here, the focus lays on Multilayer Feedforward Neural Nets, also called MLPs, for regression cases.

4.1 Structure

The Structure displayed here is the one of a MLP with one hidden layer. Every one of the M Neurons in the hidden layer is composed of a linear combination of the inputs X_t, which consist of p lagged values of the target variable y_t, and the weights vector α_m, consisting of $\alpha_{m1}, ..., \alpha_{mp}$. A bias term α_{m0} is added, and lastly the linear combination is wrapped in an *activation function* σ:

$$Z_m = \sigma(\alpha_{m0} + \alpha_m^T X_t) \tag{11}$$

The target y_t is then again modelled as a linear combination of the M hidden neurons:

$$y_t = \sigma(\beta_0 + \beta^T Z),\ \beta = \beta_1, ..., \beta_m \tag{12}$$

See figure 1[2] for a visual representation of the NN. Note that while for classification tasks, the output layer is again transformed by an activation function, there is no transformation taking place in the output layer for regression applications. Additional hidden layers can be added, with every neuron of the next layer being linear combinations of all the neurons of the precvious layer, as described above. This allows for arbitrarily large numbers of layers and neurons. Typically the model specifications are optimized via grid search.

4.2 Activation Functions

One possible option for the activation function is the identity function $f(x) = x$, which lead to the Neural Network simply being a linear combination of all the inputs, as in a linear model. Hence, the activation functions are, what adds the nonlinearity

[1] cited in Hill, O'Connor, and Remus (1996, p. 1083)

[2] code for figure adapted from https://gist.github.com/anbrjohn/7116fa0b59248375cd0c0371d6107a59

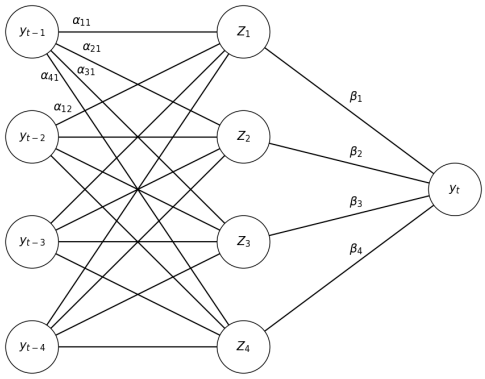

Figure 1: Structure of a Single Layer MLP with $p = 4$ and $M = 4$

to the model and enable the function approximation capabilities of the NN. Other possible options for the activation function are:

$$\text{The sigmoid function: } f(x) = \frac{1}{1 + exp(-x)} \tag{13}$$

$$\text{The tanh function: } f(x) = \frac{exp(x) - exp(-x)}{exp(x) + exp(-x)} \tag{14}$$

$$\text{The relu function: } f(x) = \begin{cases} 1 & \text{if } x > 0 \\ 0 & \text{if } x < 0 \end{cases} \tag{15}$$

Looking at the activation functions in figure 2, one can see that for the sigmoid and tanh functions, the slope is steep around zero and quickly approaches 0 the higher the absolute input values are. Therefore it is recommended to rescale the data either to the unit scale or by standardization.

4.3 Fitting Neural Networks

All the weights and biases of the NN, as mentioned in chapter 4.1, are usually fitted using backpropagation, a form of the gradient descent algorithm that has first been described for fitting NNs by Rumelhart, Hinton, and Williams (1986). Since then, a variety of different optimization strategies has evolved, an overview can be found in Goodfellow, Bengio, and Courville (2016, chap. 8). Here, the simplest case of this

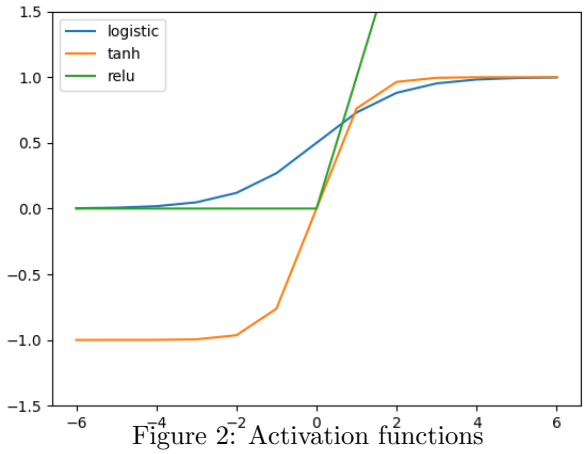

Figure 2: Activation functions

algorithm will be layed out for the model specification in chapter 4.1, following Hastie, Tibshirani, and Friedman (2009, chap. 11.4). The starting point is a cost function, which is usually the sum of squared errors for a regression model.

$$R(\theta) = \sum_{t=1}^{T}(y_t - f(X_t))^2 \tag{16}$$

Then the derivatives of the cost function w.r.t the parameters are computed.

$$\frac{\partial R(\theta)}{\partial \beta_m} = -2(y_t - f(X_t))\sigma(\alpha_{m0} + \alpha_m^T X_t) \tag{17}$$

$$\frac{\partial R(\theta)}{\partial \alpha_{mi}} = -2(y_t - f(X_t))\beta_m \sigma'(\alpha_{m0} + \alpha_m^T X_t) y_{t-i} \tag{18}$$

$$\beta_m^+ = \beta_m - \gamma \frac{\partial R(\theta)}{\partial \beta_m} \tag{19}$$

$$\alpha_{mi}^+ = \alpha_{mi} - \gamma \frac{\partial R(\theta)}{\partial \alpha_{mi}} \tag{20}$$

The algorithm starts by initializing random weights, calculating the cost function, and updating each weight according to equations 19 and 20. Next, the cost function is calculated again and the whole procedure starts over, which is done until convergence is achieved in the cost function. γ is called the learning rate, which is crucial for the performance of the algorithm. If it is set too low, the algorithm might not reach a minimum or get stuck in a local minimum. When a learning rate is picked that is too high, the algorithm can miss the global minimum and not converge.

In order to avoid overfitting, usually a regularization term $J(\theta)$, described in equation 21 is multiplied by $\lambda \geq 0$ and added to the cost function. It penalizes high

weights and shrinks them towards zero. λ is a tuning parameter. The higher λ, the stronger the penalization. As the model structure, it is usually optimized via grid search.

$$J(\theta) = \sum_{m=1}^{M} (\beta_i)^2 + \sum_{m=1}^{M} \sum_{i=1}^{p} (\alpha_{mi})^2 \tag{21}$$

5 Empirical Analysis

For the purpose of illustrating the theoretical concepts described in chapters 2 and 3, and to compare the forecasting accuracy of the models in chapter 6, monthly data on midwest housing starts in the US from the FRED-MD Database is used (McCracken and Ng, 2016), ranging from January 1959 to October 2020.

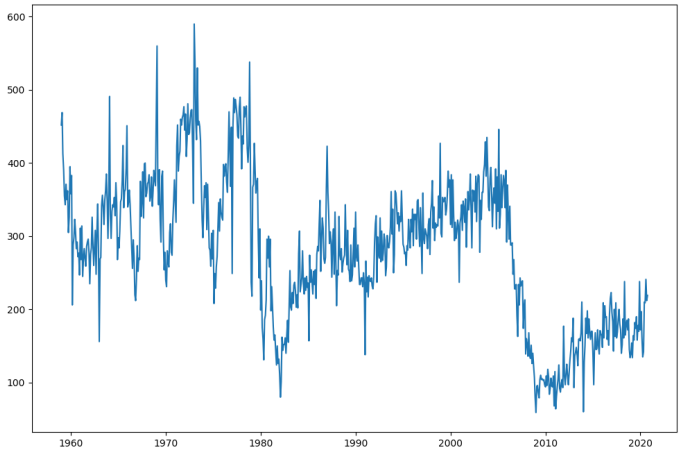

Figure 3: Midwest housing starts

The series has no clear up- or downward trending behaviour and seems to wander around non- linearly with a clear breaks in 2009 due to the global recession. Using the ADF Test with only a constant in the test regression, a p-value of 0.0994 is obtained indicating stationarity on the 10% confidence level. Next, an ARMA model to the data using the automated procedure mentioned in chapter 2.2, described in Hyndman and Khandakar (2008, chap. 3), optimizing by AIC and disregarding integration, as stationarity has been shown. After visual investigation of the ACF (figure 4) and PACF (figure 5) of the series, the maximum values of the ARMA parameters are set to the maximum number of significant lags in either the ACF or PACF, neglecting weakly significant correlations that occur after many lags of insignificant correlations.

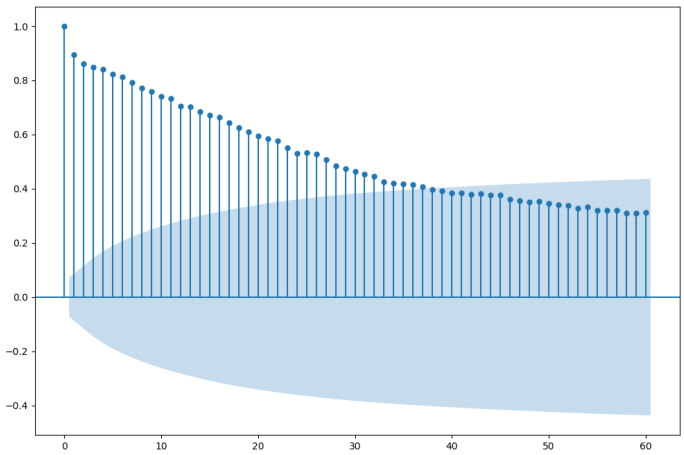

Figure 4: ACF for midwest housing starts

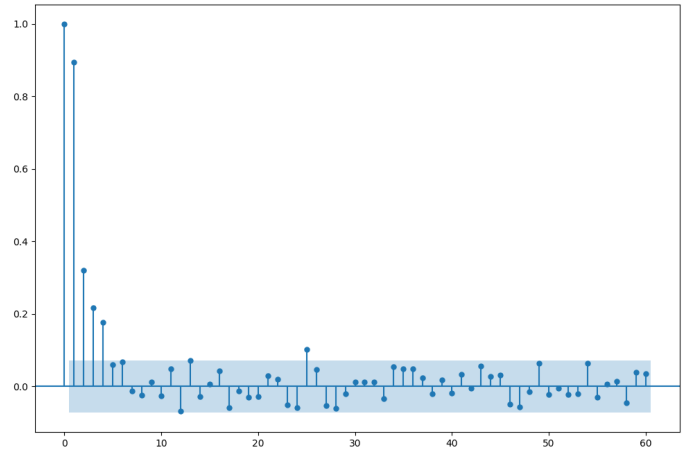

Figure 5: PACF for midwest housing starts

Carrying out the ARCH-LM Test using 3 lags according to figure 7 of the squared Residuals of the selected ARMA(6,0) [3], the null hypothesis of no conditional heteroscedasticity for the residual series of the ARMA model can clearly be rejected (p-value of 0.0).
If a fitted GARCH model captures the heteroscedasticity present in the residuals, the standardized residuals should exhibit no more heteroscedasticity (see equation 5). Hence, the optimal GARCH model is selected according to the highest p-value of an

[3]Details are given in tables 3, 4, 5

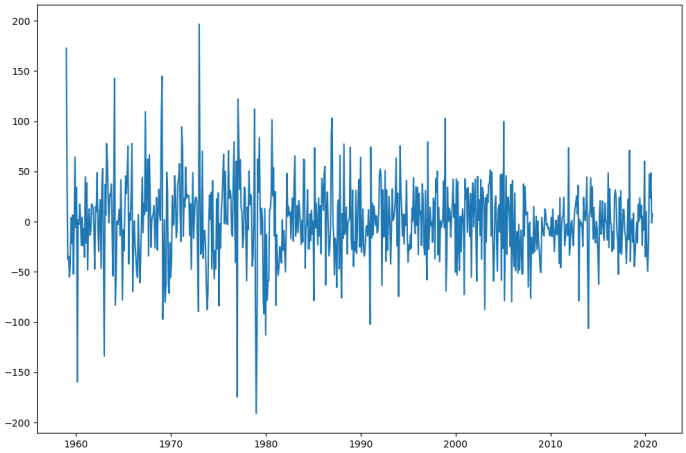

Figure 6: Residuals of ARMA model

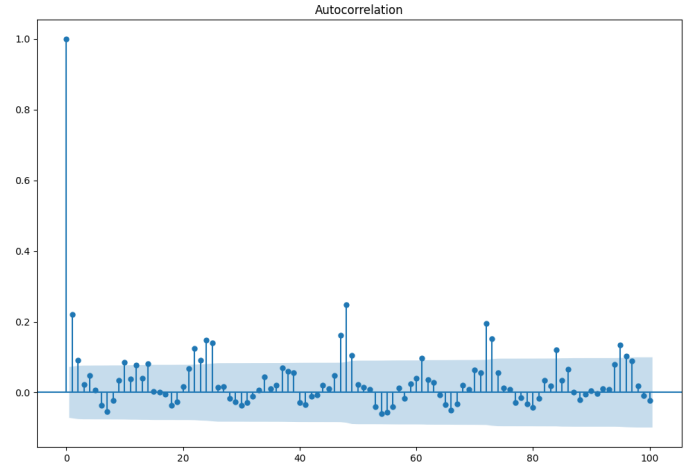

Figure 7: ACF of squared residuals of ARMA model

ARCH-LM test on its standardized residuals, selecting of an array of models with maximum p and q of 20. This leads to a GARCH(16, 8) model and a p-value of 0.2798 for the ARCH-LM test on the standardized residuals. This indicates that the present heteroscedasticty has been cleared from the residual series. Table 1 shows the results of the BDS Test, using $\frac{\epsilon}{\sigma} = 0.5$ as described in 3.3, for embedding dimensions 2 to 5. As the p-values for embedding dimensions 3 and 4 are below confidence levels of 5% and 10% respectively, the null hypothesis of independence can be rejected, and nonlinearity can be assumed for the series, as described in chapter 3.3. This means that the linear ARMA model is misspecified, which leads to inconsistent

Table 1: BDS Test Results

m	2	3	4	5
p-value	0.12176	0.048708	0.07283	0.17507

model estimates and poor forecasting capabilities, as described in chapter 2.2.

6 Forecast Comparison

In this chapter, the forecasting capabilities of ARIMA models, described in chapter 2.2, and NNs described in chapter 4 will be compared based on the time series analyzed in chapter 5, for which nonlinearity can be assumed. The dataset is split into a training and a validation sample. Next, the optimal models are determined based on the training set and h-step out of sample forecasts are computed for each values of the validation sample, as described in equations 22 to 25. y_t is the last value of the training set and real values for y are used, as long as they are part of the training sample. The forecast accuracy is then compared according to the MSE of each model described in equation 26.

$$y_t = f(y_{t-1}, ..., y_{t-p}) \tag{22}$$

$$\hat{y}_{t+1} = f(y_t, ..., y_{t-p+1}) \tag{23}$$

$$\hat{y}_{t+2} = f(\hat{y}_{t+1}, y_t, ..., y_{t-p+2}) \ ... \tag{24}$$

$$\hat{y}_{t+h} = f(\hat{y}_{t+h-1}, \hat{y}_{t+h-2}, ..., \hat{y}_{t+h-p}) \tag{25}$$

$$MSE = \frac{1}{H} \sum_{h=1}^{H} (y_{t+h} - \hat{y}_{t+h})^2 \tag{26}$$

The optimal specification of the ARIMA model is determined as in chapter 5. The optimal specification of the NN is determined via randomized grid search, allowing for nodesizes from 1 to 100 in increments of 5, maximum number of hidden layers ranging from 1 to 5 and possible values for alpha being generated on a log scale from 0.001 to 0.05. Furthermore, the relu activation function described in equation 15 was used for all hidden layers due to best performance among the three options described in chapter 4.2. For fitting the model, the Adam algorithm (Kingma and Ba, 2017) was used, which is a variant of the backpropagation algorithm described in 4.3. Out of the grid spanned by all parameter combinations, 2000 combinations are picked at random and trained based on of the first 90% of the training sample, using the remaining 10% as test set to evaluate the out of sample MSE. The best model is then picked, and retrained on the entire training set.
In order to get a more consistent measure for the forecasting accuracy, this whole

Table 2: Mean MSE for different forecast horizons

h	MLP	ARMA
1	979.43	819.13
5	984.88	997.30
10	1016.44	968.13

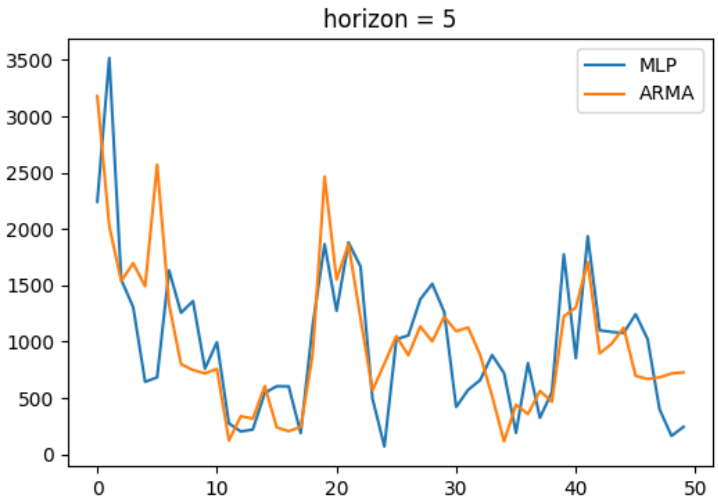

Figure 8: MSE for all subsamples and horizon = 5

procedure is repeated for 50 subsets of the original series, each time eliminating one more observation from the series, starting with the most recent one. Forecasts are computed for horizons 1, 5, and 10 in order to compare short- to long term forecasting capabilitiy. In the end, models are compared according to the mean of their MSE over all subsamples per forecast horizon.

Table 2 shows that for horizons 1 and 10, the ARMA model exhibited better forecasting capability, while for horizon 5, the MLP showed better performance. In general, the accuracies are close to another, therefore no real dominance of one method over the other can be established. Figure 8 also shows that predictive performance varies a lot depending on the subsample, with both models achiving better results than the other, but again no dominance can be inferred even for subsamples of the data. Figure 9 supports this claim. Furthermore, tables 6, 7 and 8 in the Appendix show that the structure of the NN changes drastically over subsamples. Alltogether this indicates, that neither of the model types are able to capture the underlying DGP accurately.

7 Conclusion

In summary it can be said that even though a nonlinear structure was found in the time series, NNs did not perform better at prediction tasks compared to traditional linear ARIMA models. In view of the interpretability and parsimony of traditional models, one should always stick to simpler models that are easier to interpret, if there is no sufficient advantage in using more complicated methods. There are many factors that could contribute to the poor results for NNs, for example the search space being too large to explore with reasonable computing power and the random search therefore missing global minima. Possible enhancements could be the usage of more sophisticated NN architectures, like RNNs with LSTM, that have already succesfully been applied to forecasting problems (Livieris, Pintelas, and Pintelas, 2020). This result also shows that NNs are not always the silver bullet they are often made out to be. Therefore in general, more traditional nonlinear methods like Smooth Transition Models or Kernel Regression should also be considered.

A Tables and Figures

Table 3: ARIMA Results 1

Dep. Variable:	HOUSTMW	No. Observations:	742
Model:	ARIMA(6, 0, 0)	Log Likelihood	-3789.677
Date:	Sun, 14 Mar 2021	AIC	7595.353
Time:	16:03:24	BIC	7632.228
Sample:	01-01-1959	HQIC	7609.569
NaN	- 10-01-2020		
Covariance Type:	opg		

Table 4: ARIMA Results 2

	coef	std err	z	P¿—z—	[0.025	0.975]
const	279.4040	37.648	7.422	0.000	205.616	353.192
ar.L1	0.4821	0.026	18.463	0.000	0.431	0.533
ar.L2	0.1447	0.033	4.343	0.000	0.079	0.210
ar.L3	0.1077	0.037	2.897	0.004	0.035	0.181
ar.L4	0.1388	0.038	3.666	0.000	0.065	0.213
ar.L5	0.0242	0.043	0.558	0.577	-0.061	0.109
ar.L6	0.0639	0.037	1.735	0.083	-0.008	0.136
sigma2	1590.0269	57.429	27.687	0.000	1477.469	1702.585

Table 5: ARIMA Results 3

Ljung-Box (L1) (Q):	0.00	Jarque-Bera (JB):	180.03
Prob(Q):	0.98	Prob(JB):	0.00
Heteroskedasticity (H):	0.40	Skew:	-0.06
Prob(H) (two-sided):	0.00	Kurtosis:	5.41

Table 6: Model specifications for horizon 1 for first 20 endpoints

Endpoint	MSE(MLP)	Sizes	Lambda	MSE(ARMA)	p	q
2020-10-01	108.0525	(65, 26, 71, 36, 8)	0.0022	56.0550	6	0
2020-09-01	767.7606	(46, 18, 34, 58)	0.0500	1.2838	6	0
2020-08-01	2131.1142	(57, 70, 96, 85)	0.0022	2388.6831	6	0
2020-07-01	1274.3979	(49, 50, 67, 98)	0.0005	572.6426	6	0
2020-06-01	697.9090	(23, 87, 40, 28, 17)	0.0005	2254.7438	6	0
2020-05-01	1494.3514	(29, 23, 54, 58, 98)	0.0001	370.1912	6	0
2020-04-01	2879.8432	(41, 47, 58, 66, 54)	0.0500	2462.1116	6	0
2020-03-01	1301.7760	(93, 41, 60, 62)	0.0022	1059.4028	6	0
2020-02-01	795.8054	(11, 53, 95, 42, 21)	0.0001	100.7934	6	0
2020-01-01	20.3010	(11, 72, 24, 46)	0.0022	1221.7323	6	0
2019-12-01	3746.0740	(16, 96, 66, 84, 45)	0.0106	3659.5995	6	0
2019-11-01	115.9164	(98, 31, 31, 33, 33)	0.0022	77.1133	6	0
2019-10-01	1.1357	(98, 84, 46)	0.0005	28.3060	6	0
2019-09-01	532.8066	(48, 82, 81, 12, 43)	0.0022	572.3778	6	0
2019-08-01	1680.2704	(42, 84, 88, 54, 99)	0.0001	265.7955	6	0
2019-07-01	16.4168	(51, 34, 85, 80)	0.0106	13.0928	6	0
2019-06-01	94.8374	(5, 11, 45)	0.0500	566.3249	6	0
2019-05-01	1146.4475	(87, 89, 55)	0.0005	0.1812	6	0
2019-04-01	186.3265	(13, 80, 25, 16, 79)	0.0001	317.2708	6	0
2019-03-01	439.4423	(9, 31, 88)	0.0001	444.8625	6	0

Table 7: Model specifications for horizon 5 for first 20 endpoints

Endpoint	MSE(MLP)	Sizes	Lambda	MSE(ARMA)	p	q
2020-10-01	2242.9722	(10, 79)	0.0500	3180.0076	6	0
2020-09-01	3518.1530	(46, 18, 34, 58)	0.0500	2028.0461	6	0
2020-08-01	1550.3796	(41, 98, 1, 54, 91)	0.0022	1538.4689	6	0
2020-07-01	1306.4317	(78, 77, 99, 52)	0.0022	1696.4606	6	0
2020-06-01	643.1234	(4, 99, 48, 58, 55)	0.0005	1490.8596	6	0
2020-05-01	683.6796	(59, 4, 73, 39)	0.0001	2571.0468	6	0
2020-04-01	1633.0679	(98, 37, 99, 46, 42)	0.0022	1329.2685	6	0
2020-03-01	1256.4935	(56, 89, 20, 77, 69)	0.0500	800.4911	6	0
2020-02-01	1360.4240	(44, 92, 46, 53, 94)	0.0001	747.9276	6	0
2020-01-01	760.3237	(71, 6, 85, 36, 29)	0.0106	718.7176	6	0
2019-12-01	994.7585	(78, 72, 2, 77, 91)	0.0001	758.1200	6	0
2019-11-01	274.5892	(60, 73, 99, 71, 41)	0.0001	123.1904	6	0
2019-10-01	202.5547	(91, 30, 16, 92, 63)	0.0500	338.3631	6	0
2019-09-01	219.8231	(48, 82, 81, 12, 43)	0.0022	316.4626	6	0
2019-08-01	545.7262	(3, 99, 3, 14, 94)	0.0500	606.1250	6	0
2019-07-01	604.4627	(25, 27, 98, 23, 82)	0.0106	238.6729	6	0
2019-06-01	603.4760	(4,)	0.0500	205.2611	6	0
2019-05-01	188.0122	(32, 1, 77, 64, 67)	0.0022	242.5507	6	0
2019-04-01	1139.1916	(31, 70, 75, 60)	0.0500	886.3485	6	0
2019-03-01	1865.1301	(54, 28, 37)	0.0022	2465.5638	6	0

Table 8: Model specifications for horizon 10 for first 20 endpoints

Endpoint	MSE(MLP)	Sizes	Lambda	MSE(ARMA)	p	q
2020-10-01	1409.9269	(18, 24, 81)	0.0500	1366.5520	6	0
2020-09-01	1966.6928	(9, 72, 13)	0.0106	1253.6409	6	0
2020-08-01	1337.5928	(11, 6, 43, 12)	0.0022	1185.1279	6	0
2020-07-01	877.9603	(41, 48, 66, 69)	0.0001	945.3006	6	0
2020-06-01	916.0912	(90, 17, 10, 8, 40)	0.0022	1110.5838	6	0
2020-05-01	1099.8955	(20, 92, 100, 77)	0.0022	968.6914	6	0
2020-04-01	1278.2388	(51, 90, 93)	0.0022	740.8532	6	0
2020-03-01	954.1422	(82, 64, 59)	0.0005	701.8305	6	0
2020-02-01	974.5060	(51, 26, 55, 98, 23)	0.0001	688.4155	6	0
2020-01-01	650.1563	(91, 86, 72, 61)	0.0005	914.2545	6	0
2019-12-01	783.1645	(51, 80, 30, 97)	0.0022	651.7925	6	0
2019-11-01	737.1873	(43, 53, 51, 78, 38)	0.0106	203.3346	6	0
2019-10-01	304.0333	(75, 56, 46, 60)	0.0022	207.8861	6	0
2019-09-01	554.0163	(12, 20, 69, 99, 85)	0.0005	566.7865	6	0
2019-08-01	885.5719	(84, 95, 31, 26, 85)	0.0022	1571.8325	6	0
2019-07-01	1215.4927	(81, 90)	0.0022	1349.1622	6	0
2019-06-01	669.1224	(4, 32)	0.0001	1875.4823	6	0
2019-05-01	2258.5108	(45, 74)	0.0106	1790.0809	6	0
2019-04-01	1724.5689	(77, 30, 70, 23, 32)	0.0022	1681.5795	6	0
2019-03-01	2721.9601	(14, 81, 5)	0.0005	2927.6227	4	1

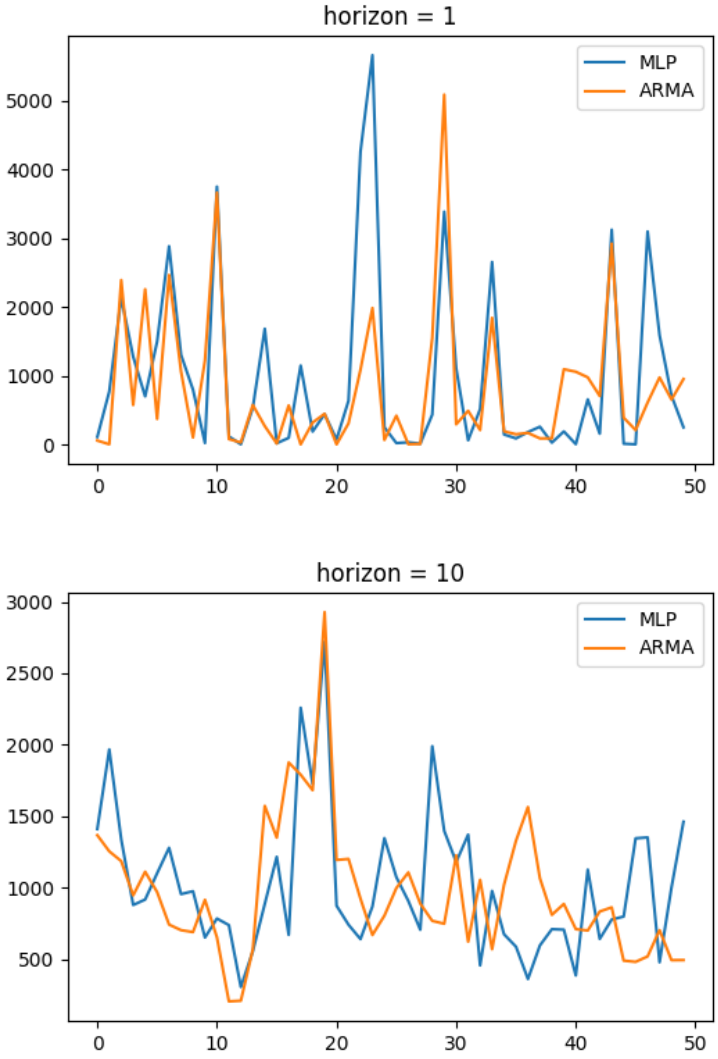

Figure 9: MSE for all subsamples and forecast horizons 1 and 10

References

Ahmed, N., A. Atiya, N. Gayar, and H. El-Shishiny (2010), "An Empirical Comparison of Machine Learning Models for Time Series Forecasting," *Econometric Reviews*, 29, 594–621, doi:10.1080/07474938.2010.481556.

Bisaglia, L., and M. Gerolimetto (2014), "Testing for (non)linearity in economic time series," *Quaderni Di Statistica*, 16, 5–32.

Blake, A. P., and G. Kapetanios (2003), "Testing for Arch in the Presence of Nonlinearity of Unknown Form in the Conditional Mean," doi: https://dx.doi.org/10.2139/ssrn.425402.

Bollerslev, T. (1986), "Generalized autoregressive conditional heteroskedasticity," *Journal of Econometrics*, 31(3), 307 – 327, doi:https://doi.org/10.1016/0304-4076(86)90063-1, URL `http://www.sciencedirect.com/science/article/pii/0304407686900631`.

Brock, W. A., J. A. Scheinkman, W. D. Dechert, and B. LeBaron (1996), "A test for independence based on the correlation dimension," *Econometric Reviews*, 15(3), 197–235, doi:10.1080/07474939608800353, URL `https://doi.org/10.1080/07474939608800353`.

Bucci, A. (2019), "Realized Volatility Forecasting with Neural Networks," *MPRA Paper No. 95443*, URL `https://mpra.ub.uni-muenchen.de/95443/`.

Caporale, G. M., C. Ntantamis, T. Pantelidis, and N. Pittis (2004), "The BDS Test as a Test for the Adequacy of a GARCH(1,1) Specification. A Monte Carlo Study," Economics Series 156, Institute for Advanced Studies, URL `https://ideas.repec.org/p/ihs/ihsesp/156.html`.

Clark, S., R. J. Hyndman, D. Pagendam, and L. M. Ryan (2020), "Modern strategies for time series regression," *International Statistical Review*.

Corradi, V., N. R. Swanson, and H. White (2000), "Testing for stationarity-ergodicity and for comovements between nonlinear discrete time Markov processes," *Journal of Econometrics*, 96(1), 39–73, doi:https://doi.org/10.1016/S0304-4076(99)00050-0, URL `https://www.sciencedirect.com/science/article/pii/S0304407699000500`.

Demetrescu, M., and R. Kruse (2013), "The power of unit root tests against nonlinear local alternatives," *Journal of Time Series Analysis*, 34(1), 40–61, doi:https://doi.org/10.1111/j.1467-9892.2012.00812.x, URL `https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1467-9892.2012.00812.x`.

Engle, R. F. (1982), "Autoregressive conditional heteroscedasticity with estimates of the variance of United Kingdom inflation," *Econometrica: Journal of the Econometric Society*, 987–1007.

Šestanović, T., and J. Arnerić (2021), "Neural network structure identification in inflation forecasting," *Journal of Forecasting*, 40(1), 62–79, doi:https://doi.org/10.1002/for.2698, URL https://onlinelibrary.wiley.com/doi/abs/10.1002/for.2698.

Funahashi, K.-I. (1989), "On the approximate realization of continuous mappings by neural networks," *Neural Networks*, 2(3), 183–192, doi:https://doi.org/10.1016/0893-6080(89)90003-8, URL https://www.sciencedirect.com/science/article/pii/0893608089900038.

Goodfellow, I., Y. Bengio, and A. Courville (2016), *Deep Learning*, MIT Press, http://www.deeplearningbook.org.

Hastie, T., R. Tibshirani, and J. Friedman (2009), *The elements of statistical learning: data mining, inference, and prediction*, Springer Science & Business Media.

Hill, T., M. O'Connor, and W. Remus (1996), "Neural Network Models for Time Series Forecasts," *Management Science*, 42, 1082–1092, doi:10.1287/mnsc.42.7.1082.

Hyndman, R. J., and Y. Khandakar (2008), "Automatic Time Series Forecasting: The forecast Package for R," *Journal of Statistical Software, Articles*, 27(3), 1–22, doi:10.18637/jss.v027.i03, URL https://www.jstatsoft.org/v027/i03.

Kingma, D. P., and J. Ba (2017), "Adam: A Method for Stochastic Optimization," .

Lee, T.-H., H. White, and C. W. Granger (1993), "Testing for neglected nonlinearity in time series models: A comparison of neural network methods and alternative tests," *Journal of Econometrics*, 56(3), 269–290, doi:https://doi.org/10.1016/0304-4076(93)90122-L, URL https://www.sciencedirect.com/science/article/pii/030440769390122L.

Livieris, I., E. Pintelas, and P. Pintelas (2020), "A CNN-LSTM model for gold price time series forecasting," *Neural Computing and Applications*, 32, doi:10.1007/s00521-020-04867-x.

Lütkepohl, H., and M. Krätzig (2004), *Applied Time Series Econometrics*, Themes in Modern Econometrics, Cambridge University Press, doi:10.1017/CBO9780511606885.

Makridakis, S., E. Spiliotis, and V. Assimakopoulos (2018), "Statistical and Machine Learning forecasting methods: Concerns and ways forward," *PLOS ONE*, 13(3), 1–26, doi:10.1371/journal.pone.0194889, URL https://doi.org/10.1371/journal.pone.0194889.

McCracken, M. W., and S. Ng (2016), "FRED-MD: A Monthly Database for Macroeconomic Research," *Journal of Business & Economic Statistics*, 34(4), 574–589, doi:10.1080/07350015.2015.1086655, URL https://doi.org/10.1080/07350015.2015.1086655.

Rumelhart, D. E., G. E. Hinton, and R. J. Williams (1986), *Learning Internal Representations by Error Propagation*, MIT Press, Cambridge, MA, USA, p. 318–362.

Sánchez García, J., and S. Cruz Rambaud (2020), "A GARCH approach to model short-term interest rates: Evidence from Spanish economy," *International Journal of Finance & Economics*, doi:https://doi.org/10.1002/ijfe.2234, URL `https://onlinelibrary.wiley.com/doi/abs/10.1002/ijfe.2234`.