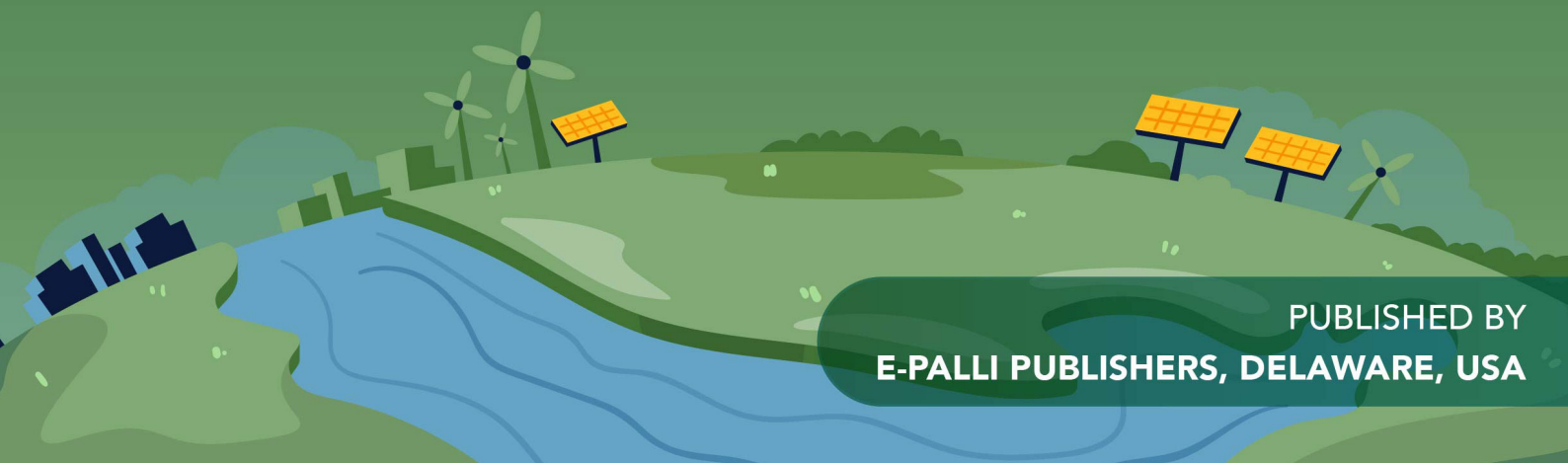
AMERICAN JOURNAL OF
ENVIRONMENT AND CLIMATE (AJEC)

ISSN: 2832-403X (ONLINE)

VOLUME 2 ISSUE 3 (2023)



PUBLISHED BY
E-PALLI PUBLISHERS, DELAWARE, USA

American Journal of Environment and Climate (AJEC)

Volume 2 Issue 3, Year 2023
ISSN: 2832-403X (Online)
DOI: https://doi.org/10.54536/ajec.v2i3.2093
https://journals.e-palli.com/home/index.php/ajec

Time Series Modeling of Global Average Absolute Sea Level Change

Yeong Nain Chi[1*]

Article Information

Received: September 27, 2023

Accepted: October 30, 2023

Published: November 06, 2023

Keywords

ARIMA, Bayesian Regularization Algorithm, Global Average Absolute Sea Level Change, HYBRID ARIMA-NARNN, Levenberg-Marquardt Algorithm, Mean Squared Error, Modeling, NARNN, Scaled Conjugate Gradient Algorithm, Time Series

ABSTRACT

This study aimed to demonstrate the effectiveness of time series models in modeling long-term records of global average absolute sea level changes from 1880 to 2014. Following the Box–Jenkins methodology, the ARIMA(0,1,2) model with drift was identified as the best-fit model for the time series due to its lowest AIC value. Using the LM algorithm, the results revealed that the NARNN model with 7 neurons in the hidden layer and 7 time delays exhibited the best performance among the nonlinear autoregressive neural network models, as indicated by its lower MSE. While ARIMA models excel in modeling linear problems within time series data, NARNN models are better suited for nonlinear patterns. However, a HYBRID model was explored, which combines the strengths of both ARIMA and NARNN models, offering the capability to address both linear and nonlinear aspects of time series data. The comparative analysis of this study demonstrated that the HYBRID model, with 6 neurons in the hidden layer and 7 time delays, outperformed the NARNN model with 7 neurons in the hidden layer and 7 time delays, as well as the ARIMA(0,1,2) model, with the lowest MSE in this study. These findings represent a significant step in time series forecasting by leveraging the strengths of both statistical and machine learning methods.

INTRODUCTION

Climate change is the long-term alteration in the Earth's average weather patterns. Sea level rise primarily results from two factors associated with climate change: the influx of water from the melting of land ice, including ice sheets and glaciers, into the world's oceans, and the expansion of seawater due to rising temperatures. The potential consequences of sea level rise encompass various challenges, including but not limited to: 1) Increased coastal flooding and erosion, 2) Damage to agricultural land and crops, 3) Impacts on coastal and urban settlements and infrastructure, 4) Harm to coastal flora and fauna ecosystems, 5) Escalation of environmental sanitation issues, and 6) Exacerbation of public health concerns. These effects underscore the urgency of addressing climate change and its associated sea level rise to mitigate their detrimental impacts.

In 2014, the Intergovernmental Panel on Climate Change (IPCC) estimated that global sea levels had risen by a range of 26 to 55 cm (approximately 10 to 22 inches) with a 67% confidence interval. According to the Fourth National Climate Assessment Report by the U.S. Global Change Research Program (USGCRP) in 2017, sea levels had risen approximately 7 to 8 inches (around 16 to 21 cm) since 1900, with about 3 inches (approximately 7 cm) of that increase occurring since 1993. The National Oceanic and Atmospheric Administration (NOAA), in its 2019 Global Climate Annual Report (https://www.ncdc.noaa.gov/sotc/global/201913), stated that global annual temperatures have been on the rise. They have increased at an average rate of approximately 0.07°C (0.13°F) per decade since 1880.

Numerous studies (Church *et al.*, 2008; Cazenave & Llovel, 2010; Cazenave & Cozannet, 2013; Horton *et al.*, 2018; Kulp & Strauss, 2019; Haasnoot, 2020) have consistently pointed out that sea levels are rising at an accelerating pace. This underscores the significance of comprehending historical sea level variations for analyzing present-day changes and anticipating future trends. In recent years, significant advancements have been in modeling sea level changes and understanding their underlying causes. These improvements are largely attributed to the availability of new in situ and remote sensing observations (Foster & Brown, 2014; Visser *et al.*, 2015; Bolin, 2015; Srivastava *et al.*, 2016). Despite the critical importance of sea level rise and its far-reaching consequences, it's noteworthy that there remains a paucity of studies in the technical literature dedicated to global sea level change prediction schemes.

Time series forecasting is a crucial data science technique applied across various disciplines. It involves using a model to predict future values based on past observations and has a solid theoretical foundation in statistics. In time series analysis, several widely-used models include: 1) AR (Autoregressive): This model considers the relationship between a current value and its past values, 2) MA (Moving Average): It focuses on the relationship between a current value and a stochastic white noise term, 3) ARMA (Autoregressive Moving Average): A combination of the AR and MA models, it incorporates both autoregressive and moving average components, and 4) ARIMA (Autoregressive Integrated Moving Average): This model combines differencing (to make the series stationary) with the ARMA model. These models

[1] University of Maryland Eastern Shore, United States
[*] Corresponding author's e-mail: ychi@umes.edu

are typically employed when dealing with time series data that exhibits constant variance. In contrast, the ARCH (Autoregressive Conditionally Heteroscedastic) model is designed for time series with changing variance over time. On the other hand, the GARCH (Generalized Autoregressive Conditional Heteroskedasticity) model extends ARCH by including both autoregressive and moving average components (Engle, 2001).

There is a burgeoning interest in harnessing neural networks for time series modeling and forecasting. Neural networks have emerged as a dominant trend in machine learning when tackling these tasks. They are nonlinear, nonparametric statistical methods (Zhang *et al*., 1998). One of the key advantages of neural networks is their ability to make predictions without imposing strong assumptions about the underlying data. They offer flexible, goal-driven modeling, which makes them a compelling alternative to traditional time series models for addressing complex forecasting challenges. Empirical evidence often supports the superiority of neural network models in terms of forecasting performance compared to conventional methods (Hill *et al*., 1996). However, it's worth noting that the construction of neural network models may not always follow a systematic procedure (Tealab, 2018). As a result, a significant area for future research lies in formulating new approaches to designing neural network models that can provide robust and reliable solutions to complex forecasting problems.

The primary objective of this study was to apply three distinct models-ARIMA, Nonlinear Autoregressive Neural Network (NARNN), and Hybrid ARIMA-NARNN (HYBRID)-to model and forecast the long-term records of global average absolute sea level change from 1880 to 2014. The NARNN and HYBRID models were employed in this study due to their notable advantages, including faster convergence and superior performance when compared to conventional neural network models (Adamowski *et al*., 2012). In this study, both the NARNN and HYBRID models underwent training utilizing three different training algorithms: Levenberg–Marquardt (LM), Bayesian Regularization (BR), and Scaled Conjugate Gradient (SCG). This innovative experiment can be regarded as a pilot study that explores the application of NARNN and HYBRID models for assessing global average absolute sea level change. The findings of this study have the potential to bridge a significant gap in time series forecasting by integrating the strengths of both statistical and machine learning methods. The remainder of the paper is organized into the following sections: Section 2: Introduces the time series data. Section 3: Describes the ARIMA, NARNN, HYBRID models, and the training algorithms used in this study. Section 4: Presents and analyzes the empirical results. Section 5: Concludes the study.

MATERIALS AND METHODS

Data

The data used in this study is publicly accessible through the US Environmental Protection Agency (EPA) website at https://www.epa.gov/climate-indicators/climate-change-indicators-sea-level, utilizing information sourced from the Australia's Commonwealth Scientific and Industrial Research Organization (CSIRO) in 2015, available at https://www.cmar.csiro.au/sealevel/sl_data_cmar.html. For the purpose of accurate analysis, this study combined two sets of adjusted sea level data: CSIRO data covering the period from 1880 to 1992 and NOAA data spanning from 1993 to 2014. The resulting dataset represents global average absolute sea level change from 1880 to 2014, as depicted in Figure 1. The mean global average absolute sea level change in this dataset was measured at 3.6408 mm, with a standard deviation of 2.4297 mm. Notable statistics within this dataset include a minimum sea level change of -0.4409 mm in 1882, a maximum of 8.6637 mm in 2014, and a median of 3.3740 mm in 1947.

The data contains "cumulative changes in sea level for the world's oceans since 1880, based on a combination of long-term tide gauge measurements and recent satellite measurements. It shows average absolute sea level change, which refers to the height of the ocean surface, regardless

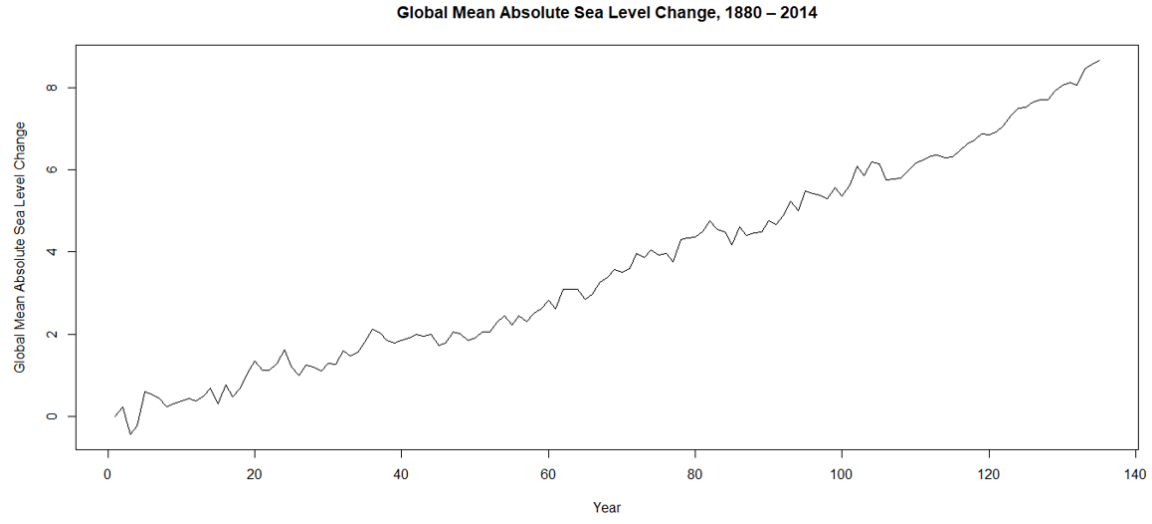


Figure 1: Time series plot of global average absolute sea level change, 1880 - 2014 (R Output)

of whether nearby land is rising or falling. Satellite data are based solely on measured sea level, while the long-term tide gauge data include a small correction factor because the size and shape of the oceans are changing slowly over time. (On average, the ocean floor has been gradually sinking since the last Ice Age peak, 20,000 years ago.)" (Quoted from https://datahub.io/core/sea-level-rise#readme).

METHODOLOGY

The ARIMA Model

Time series data involves the sequential collection of information over time. In a univariate time series, it comprises a series of individual (scalar) observations recorded at specific time points, denoted as "t." This sequence of random variables is represented as $\{y_t: t = 1, 2, \cdots, T\}$, where "yt" belongs to the real numbers ($y_t \in R$), and "t" in "T" denotes the time index when each data point was observed. This collection of random variables constitutes a stochastic process. Stochastic processes are frequently employed in modeling time series data to determine the underlying parameters of the time series. Once these parameters are established, the stochastic process serves as a model that can be utilized to forecast future values of the time series.

The ARIMA model is a statistical analysis tool used for modeling and forecasting time series data with the aim of predicting future trends. Each component of the ARIMA model-namely, the autoregressive (AR) component, the differencing (I) component, and the moving average (MA) component-serves a distinct purpose. These components collectively work to enhance the model's accuracy in predicting future data points within the time series (Montgomery *et al.*, 2008). Statistically, ARIMA(p,d,q) model can be expressed as:

$$y_t = \phi_1 y_{t-1} + \phi_2 y_{t-2} + \cdots + \phi_p y_{t-p} + e_t + \theta_1 e_{t-1} + \theta_2 e_{t-2} + \cdots + \theta_q e_{t-q}$$
$$= \sum_{i=1}^{p} \phi_i y_{t-i} - \sum_{j=1}^{q} \theta_j e_{t-j} + e_t \quad (1)$$

where p = the order of the autoregressive process (the number of lagged terms), d = the number of differences required to make the time series stationary, q = the order of the moving average process (the number of lagged terms), $\phi = (\phi_1, \phi_2, \cdots, \phi_p)$ is the vector of model coefficients for the autoregressive process, $\theta = (\theta_1, \theta_2, \cdots, \theta_q)$ is the vector of model coefficients for the moving average process, and et = the residual error (i.e., white noise).

In the realm of time series forecasting, the Box-Jenkins methodology (Box & Jenkins, 1970) represents a systematic approach used for the identification, estimation, verification, and prediction of ARIMA models. This methodology relies on analyzing the autocorrelation function (ACF) and partial autocorrelation function (PACF) to assess the stationary of the univariate time series and determine the appropriate lag lengths (Box *et al.*, 2016).

In time series analysis, a common assumption is that the time series is stationary, implying that the statistical characteristics, such as the mean and variance, remain constant over time. Consequently, when applying the Box-Jenkins methodology, it initially assumes the time series to be in a non-stationary state. To address this, differencing can be employed as a technique to transform the data and make it stationary. Practically, visual plots and summary statistics play a crucial role in identifying trends and autoregressive elements within the time series. This information helps determine the appropriate degree of differencing and the size of the lag required for model identification.

To determine suitable parameters for a time series model, one commonly utilized approach involves employing information criteria such as Akaike's Information Criterion (AIC) or the Bayesian Information Criterion (BIC). These criteria help in selecting the appropriate orders for an ARIMA model by minimizing their respective values. In the diagnostic checking phase, various tools can be employed to assess the model's performance. These include visual inspection of residual error plots and conducting statistical tests on the residuals. These diagnostic checks serve several purposes, such as evaluating how well the chosen model fits the time series data and identifying areas where model improvement may be necessary. This step is essential for ensuring the model's reliability and accuracy in forecasting.

The NARNN Model

The concept underlying the AR process is to explain the current value of the time series, denoted as yt, by a function of the previous p values, $(y_{t-1}, y_{t-2}, \cdots, y_{t-p})$. Consequently, the autoregressive process of order p, often represented as AR(p), is defined by the following equation:

$$y_t = \phi_1 y_{t-1} + \phi_2 y_{t-2} + \cdots + \phi_p y_{t-p} + e_t = \sum_{i=1} p\, \phi_i y_{t-i} + e_t \quad (2)$$

In this equation, ϕ represents a vector of model coefficients for the autoregressive process, denoted as $\phi = (\phi_1, \phi_2, \cdots, \phi_p)$, and et represents white noise with the property that et follows a normal distribution with mean 0 and variance σ^2, denoted as $e_t \sim N(0, \sigma^2)$ (Montgomery *et al.*, 2008)

The NARNN represents a natural extension of the traditional linear AR(p) process. An NARNN of order p can be expressed as follows:

$$y_t = \Phi(y_{t-1}, y_{t-2}, \cdots, y_{t-p}, w) + \varepsilon_t \quad (3)$$

In this equation, $\Phi(\cdot)$ is an unspecified function that is determined by the structure and connection weights of the neural network. The vector w encompasses all parameters, including the weights of the network. Lastly, ε_t represents the error term. Essentially, the NARNN conducts a nonlinear mapping from past observations $(y_{t-1}, y_{t-2}, \cdots, y_{t-p})$ to predict future values, yt. This concept aligns with the idea of a nonlinear autoregressive model, as introduced by Zhang in 2003.

In the context of time series data, lagged values of the time series can be employed as inputs to a neural network, resulting in what is commonly referred to as the NARNN model. Mathematically, the NARNN model (Benrhmach *et al.*, 2020) can be represented by the following equation:

$y_t = a_0 + \sum_{j=1}^{k} w_j \Phi(b_{0j} + \sum_{i=1}^{d} w_{ij} y_{t-i}) + \varepsilon_t$ (4)

where d is the number of input units, k is the number of hidden units, a_0 is the constant corresponding to the output unit, b_{0j} is the constant corresponding to the hidden unit j, w_j is the weight of the connection between the hidden unit j and the output unit, w_{ij} is the parameter corresponding to the weight of the connection between the input unit i and the hidden unit j, and $\Phi(\cdot)$ is a nonlinear activation function, so-called this the transfer (activation) function. The logistic function (i.e., sigmoid) is commonly used as the hidden layer transfer function, that is, $\Phi(y) = 1 / (1 + \exp(-y))$. The NARNN model essentially captures complex nonlinear relationships by using lagged values as inputs and applying the specified mathematical formulation for forecasting the current value of the time series.

The HYBRID Model

The ARIMA and NARNN models excel in modeling linear and nonlinear aspects of time series data, respectively. However, when a time series exhibits a combination of both linear and nonlinear characteristics, a HYBRID model, which merges the strengths of ARIMA and NARNN, can be a superior choice for accurate modeling. This HYBRID approach assumes the presence of an unknown function that represents the relationship between the linear and nonlinear components within the time series. This relationship can be depicted as follows:

$y_t = f(L_t, N_t)$ (5)

where linear component is represented by Lt, and nonlinear component is shown by Nt. Assuming that the linear and nonlinear components in the time series interact through simple additive relationships, as suggested by Zhang (2003), the time series can be conceptualized as a composite of both linear and nonlinear components, as articulated below:

$y_t = (L_t, N_t)$ (6)

Initially, the linear component of the time series is addressed by modeling it using the ARIMA model. Subsequently, the residuals obtained from the SARIMA (seasonal ARIMA) model capture the remaining nonlinear relationship within the data. This nonlinear component can be extracted by calculating the differences between the observed values and the predicted values, as demonstrated below:

$e_t = y_t - \hat{L}_t$ (7)

where et is the residual of the linear model at time t, and t is the predicted value for time t. To find the nonlinear relationship, residuals can be modelled by the NARNN model in this study as follows:

$\hat{N}_t = e_t = f(e_{t-1}, e_{t-2}, \cdots, e_{t-n}) + \varepsilon_t$ (8)

where f is the transformation function modelled by the NARNN model, and ε_t is the random error. The forecast from the ARIMA and NARNN models are combined to obtain the forecast of the time series ŷt which is denoted by

$\hat{y}_t = \hat{L}_t + \hat{N}_t$ (9)

Training Algorithms

The NARNN and HYBRID models commonly employ various learning algorithms, with the most frequent ones being the Levenberg-Marquardt, Bayesian Regularization, and Scaled Conjugate Gradient training algorithms. Training in these models involves the crucial process of finding the optimal network weights and bias points for the multilayer feedforward neural network. This optimization is achieved by defining a total error function, which quantifies the disparity between the network's output and the desired target values. The objective of the training process is to minimize this error function with respect to the network's weights and bias parameters. This iterative optimization process is what enables the neural network to learn and improve its ability to make accurate predictions or classifications.

The LM Algorithm

The LM algorithm, initially introduced by Levenberg in 1944 and later rediscovered by Marquardt in 1963, is a widely employed iterative method for solving nonlinear minimization problems. Such problems commonly emerge in applications like least squared curve fitting. The LM algorithm combines aspects of both gradient descent and the Gauss-Newton method. One of its key advantages is that it operates without the need to explicitly compute the exact Hessian matrix. Instead, it relies on the gradient vector and the Jacobian matrix, enabling faster training while maintaining stable convergence, as highlighted in Gavin's work in 2020.

The LM algorithm is a variant of Newton's method that is particularly effective for training neural networks, especially when the performance index is defined as the mean squared error. When the performance function, also known as the network error function, takes the form of a sum of squares, it becomes possible to approximate the Hessian matrix and compute the gradient as follows:

Hessian Matrix (H) is calculated as: $H = J^T J$ (10)

Gradient Vector (G) is calculated as: $G = J^T e$ (11)

In these equations, J represents the Jacobian matrix, which contains the first-order derivatives of the network errors with respect to the weights and biases, and e is a vector representing the network errors. Computation of the Jacobian matrix can be performed using a standard backpropagation technique, which is computationally less complex than directly calculating the Hessian matrix. The LM algorithm itself can be computed using the flowing equation:

$x_{k+1} = x_k - [J^T J + \mu I]^{-1} J^T e$ (12)

In this equation, x_k represents the current connection weights, x_{k+1} represents the next connection weights, I is the identity matrix, and μ is a scalar referred to as the combination coefficient. The LM algorithm exhibits an important feature: when μ is increased, it approaches the behavior of the steepest descent algorithm with a small learning rate, while decreasing μ towards zero makes the algorithm behave like the Gauss-Newton method, as

discussed by Hagan in 2014. This flexibility allows the LM algorithm to adapt its learning behavior during training.

The BR Algorithm

The BR algorithm, introduced by MacKay (1992), automatically determines the optimal performance function to achieve excellent generalization using a Bayesian inference approach. This algorithm is founded on the probabilistic interpretation of network parameters. Bayesian optimization of regularization parameters relies on calculating the Hessian matrix at the minimum point. Consequently, the BR algorithm incorporates a probability distribution of network weights, and the network architecture can be regarded as a probabilistic framework (Sariev & Germano, 2020).

Similar to the LM algorithm, the BR algorithm is employed for optimizing weights and biases while minimizing the sum of squared errors. It introduces network weights into the training objective function, denoted as F(W) as follows:

$$F(W) = \alpha E_W + \beta E_D \quad (13)$$

Here, E_D represents the sum of network errors, and E_W is the sum of squared network weights. The parameters α and β are the objective function parameters, and their values are determined using Bayes' theorem. Furthermore, the Gaussian distribution is utilized to model both the network weights and training sets. These parameters are updated iteratively until convergence is achieved (Yue *et al.*, 2011).

In order to discover the optimal weight space, the objective function must be minimized, which is equivalent to maximizing the posterior probability function, defined as follows:

$$P(x|D, \alpha, \beta, M) = P(D|x, \beta, M)\, P(x|\alpha, M) \,/\, P(D|\alpha, \beta, M) \quad (14)$$

In this equation, x represents the vector containing all the weights and biases in the network, D denotes the training data set, and α and β are parameters associated with the density functions $P(D|x, \beta, M)$ and $P(x|\alpha, M)$, respectively. M represents the selected model, which corresponds to the chosen architecture of the network (Hagan, 2014).

As a consequence of this process, the algorithm identifies optimal values for α and β within a given weight space. Subsequently, the algorithm transitions into the LM phase, involving Hessian matrix calculations to adjust the weight space and minimize the objective function. If convergence is not achieved, the algorithm estimates new values for α and β, and the entire procedure repeats iteratively until convergence is reached (Yue *et al.*, 2011).

The SCG Algorithm

The SCG algorithm, developed by Møller (1993), is grounded in the Conjugate Gradient Method. However, what sets this algorithm apart is its omission of a line search at each iteration. In contrast to many standard backward propagation algorithms, the SCG algorithm is fully automated, eliminating the need for critical user-specific parameters and avoiding time-consuming line searches. By incorporating the model trust region approach from the LM algorithm with the Conjugate Gradient method, the SCG algorithm can be expressed as (Møller, 1993):

$$s_k = [E`(w_k + \sigma_k p_k) - E`(w_k) \,/\, \sigma_k] + \lambda_k p_k \quad (15)$$

Here, s represents the Hessian matrix approximation, E is the total error function, and $E`$ is the gradient of E. Scaling factors λ_k and σ_k are introduced to approximate the Hessian matrix and are initialized by the user at the start of the algorithm, subject to the constraints $0 < \lambda_k < 10^{-6}$ and $0 < \sigma_k < 10^{-4}$. In the SCG algorithm, the calculation of factor "β_k" and the direction of the new search are defined as follows (Møller, 1993):

$$\beta_k = (|g_{k+1}|^2 - g_{k+1}^T g_k) \,/\, g_k^T g_k \quad (16)$$

$$p_{k+1} = -g_{k+1} + \beta_k p_k \quad (17)$$

One notable advantage of this algorithm is the ability to independently update the design parameters at each iteration, eliminating the need for user intervention. This stands as a significant advantage when compared to line search-based algorithms.

RESULTS AND DISCUSSION

The ARIMA Model

To find a solution, the function "auto.arima()" was employed from the "forecast" package in R version 4.3.1 for Windows. This function was used to identify both the structure of the time series (whether it's stationary or not) and its type (whether it's seasonal or not). Additionally, it automatically sets the model's parameters by considering the AIC, AICc, or BIC values generated to determine the best-fit ARIMA model. Based on this approach, the ARIMA(0,1,2) model was selected for further forecasting processes, and the parameters of the ARIMA(0,1,2) model are presented in Table 1.

Table 1: Parameters of the ARIMA(0,1,2) with drift model

Parameter	Estimate	Standard Error
Constant	0.0644	0.0080
Difference	1	
MA Lag 1	-0.3806	0.0840
MA Lag 2	-0.1598	0.0823
Sigma2 estimated as 0.0402, Log Likelihood = 26.57		
AIC = -45.13, AICc = -44.82, BIC = -33.54		
Training Set Error Measures:		
RMSE = 0.1975106, MSE = 0.0390140, MAE = 0.1525891		

Source: Own work

In this case, the Ljung-Box Q-test (Ljung and Box, 1978) was applied, yielding a test statistic of Q = 2.6565 with 7 degrees of freedom. The p-value associated with the test was 0.9149, taking into account the model degrees of freedom (3) and total lags used (10). These results indicate that the residuals are indeed random, signifying that the model provides a suitable fit for the time series. The combination of these findings, along with the Ljung-Box Q-test statistic, suggests that the ARIMA(0,1,2) model with drift effectively captures the dynamics of this time series.

The NARNN Model

In MATLAB, the NARNN model, applied for time series prediction, utilizes past values of a univariate time series and can be expressed as follows:

$$y(t) = \Phi(y(t-1), y(t-2), \cdots, y(t-d)) + e(t) \quad (18)$$

In this equation, y(t) represents the time series value at time t, d is the time delay, and e(t) stands for the error in approximating the time series at time t. This formula illustrates how the NARNN model predicts the future value of a time series, y(t), based on its past values, (y(t-1), y(t-2), ⋯, y(t-d)). The function Φ(·) represents an unknown nonlinear function, and training the neural network aims to approximate this function by optimizing the network weights and neuron biases. The objective is to minimize the sum of squared differences between observed values (y_i) and predicted values ($\hat{y}_i$), often quantified as the Mean Squared Error (MSE) given by:

$$MSE = (1/n) \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \quad (19)$$

As such, the prediction performance of the models is assessed using their MSE (Beale *et al.*, 2019).

In this study, the extracted features were trained using three different training algorithms: LM, BR, and SCG. These algorithms were implemented in the MATLAB (2022a) Neural Network Toolbox to model a time series dataset consisting of 135 timesteps, representing global average absolute sea level change from 1880 to 2014. For the purpose of this analysis, the time series dataset was divided into three segments: 70% for training, 15% for validation, and 15% for testing. Specifically, the 135 data samples were randomly split into 90 data samples for training, 19 for validation, and 19 for testing.

The training process automatically ceased when no further improvement in generalization was observed, as indicated by an increase in the mean square error of the validation samples (Beale *et al.*, 2019). The results showed that the training progress using the LM algorithm stopped after six iterations when the validation error reached 0.0204, with a gradient of 0.00151 and a Mu value of 0.001 at epoch 13. In terms of processing time, the LM algorithm completed training in 00:00:00. Furthermore, the error analysis revealed that the NARNN model, configured with 7 neurons in the hidden layer and a time delay of 7, yielded the best performance, with a mean squared error (MSE) of 0.0227 when trained using the LM algorithm.

The HYBRID Model

In MATLAB, the HYBRID model applied to time series prediction using its past residuals from the SARIMA model can be expressed as follows:

$$e(t) = \Phi(e(t-1), e(t-2), \cdots, e(t-d)) + \varepsilon(t) \quad (20)$$

where e(t) is the residual of the time series at time t, d is the time delay, and ε(t) is the error term. This equation describes how the HYBRID model is used to predict the future residual of a time series, e(t), using the past residuals of the time series, (e(t-1), e(t-2), ⋯, e(t-d)).

In a similar vein, devising the optimal architecture for the HYBRID model entails determining key factors such as time delays, the number of hidden neurons, and selecting an efficient training algorithm. In this analytical study, the time series dataset was divided into three segments: 70% for training, 15% for validation, and 15% for testing. This partitioning resulted in a random allocation of 135 data samples, with 90 allocated for training, 19 for validation, and 19 for testing.

Upon conducting error analysis, it was found that the HYBRID model, configured with 6 neurons in the hidden layer and a time delay of 7, delivered the most favorable performance, achieving a MSE of 0.0129 when trained using the LM algorithm (as detailed in Table 2). Furthermore, the error analysis revealed that the NARNN model, configured with 7 neurons in the hidden layer and a time delay of 7, yielded the best performance, with a

Table 2: HYBRID model selection using the LM, BR, SCG algorithms

Layer Size	Time Delay	LM	BR	SCG
		MSE	MSE	MSE
6	5	0.0288	0.0331	0.0303
	6	0.0228	0.0352	0.0360
	7	0.0129	0.0351	0.0338
	8	0.0367	0.0330	0.0321
	9	0.0274	0.0322	0.0338
7	5	0.0288	0.0363	0.0408
	6	0.0282	0.0298	0.0303
	7	0.0344	0.0358	0.0324
	8	0.0207	0.0325	0.0300
	9	0.0239	0.0355	0.0326
8	5	0.0186	0.0346	0.0278
	6	0.0184	0.0329	0.0333
	7	0.0233	0.0347	0.0284
	8	0.02589	0.0347	0.0296
	9	0.0323	0.0323	0.0358

9	5	0.0346	0.0337	0.0353
	6	0.0322	0.0325	0.0345
	7	0.0247	0.0355	0.0304
	8	0.0248	0.0333	0.0340
	9	0.0167	0.0346	0.0334

Source: Own work

MSE of 0.0227 when trained using the LM algorithm. Simultaneously, while utilizing the LM algorithm for training the HYBRID model, the training process halted when the validation error exhibited an increase over six consecutive iterations. This termination was observed with performance metrics of Performance = 0.0958, Gradient = 0.0056, and Mu = 0.0001, which occurred at epoch 18. Notably, in terms of processing time, the LM algorithm completed its training phase within a negligible duration of 00:00:00, as outlined in Table 3.

Table 3: HYBRID training output

Unit	Initial Value	Stopped Value	Target Value
Epoch	0	18	1000
Elapsed Time	---	00:00:00	---
Performance	0.293	0.0958	0
Gradient	128	0.0056	1e-07
Mu	0.001	0.0001	1e+10
Validation Checks	0	6	6

(MATLAB Output)

HYBRID Best Performance

The performance plot serves as an informative tool, depicting the relationship between the training, validation, and testing phases in forecasting global mean absolute sea level change, as measured by the MSE against the number of epochs. The evaluation of performance involved computing the MSE and epochs upon the completion of training. This graphical representation is invaluable for diagnostic purposes, as it allows for the observation of the progress of training by plotting the training, validation, and testing errors. The plot clearly indicates the point at which training ceased due to an increase in the validation error, as circled in the epoch.

As depicted in Figure 2, the HYBRID model achieved its best performance during the validation phase, yielding an MSE of 0.047631 at epoch 12. The results further demonstrate the model's robust performance, with the validation and testing errors exhibiting similar characteristics. This suggests that significant overfitting is unlikely to have occurred.

Figure 2: Performance plot of the HYBRID model (MATLAB Output)

HYBRID Neural Network Error Histogram

The error histogram serves as a valuable tool for identifying outliers within the dataset, which are data points exhibiting a considerably worse fit compared to the majority of the data. In Figure 3, the error histograms illustrate this phenomenon, with blue bars representing the training data, green bars representing the validation data, and red bars representing the testing data.

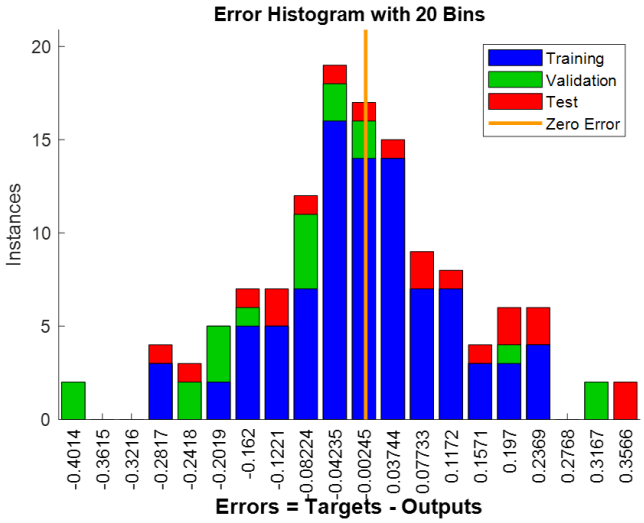


Figure 3: Error histogram of the HYBRID model (MATLAB Output)

The results from the histograms revealed the presence of a few testing and validation points that fell outside the expected range. If these outliers are indeed valid data points but exhibit dissimilarity to the rest of the dataset, it may indicate that the neural network is extrapolating to accommodate these points. In such cases, it becomes imperative to consider incorporating more data points similar to these outliers during the training analysis. Additionally, retraining the network may be necessary to improve its performance on these atypical data points.

HYBRID Time-Series Response

Figure 4 presents the dynamic network time-series response plots for the HYBRID model, illustrating that the model's outputs are evenly distributed on both sides of the response curve. Furthermore, the errors versus time remained consistently small throughout the training, validation, and testing phases. These findings provide strong evidence that the model successfully predicted the time series across the simulation period, demonstrating its efficiency and reliability.

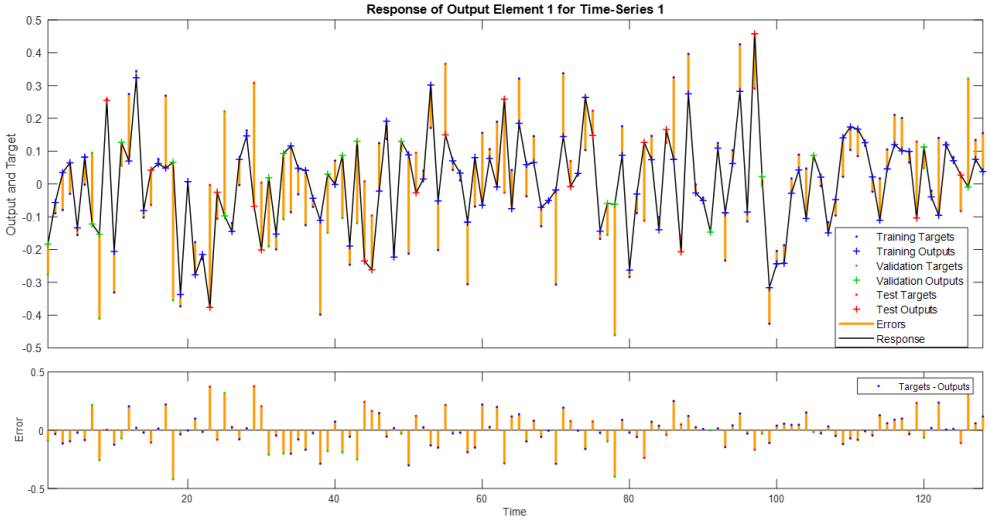


Figure 4: Network time-series response of the HYBRID model (MATLAB Output)

HYBRID Error Autocorrelation

The error autocorrelation function provides insight into how prediction errors are temporally related. In an ideal prediction model, only one nonzero value should appear in the autocorrelation function, precisely at zero lag (corresponding to the MSE). This scenario implies that the prediction errors are entirely uncorrelated, resembling white noise. However, if notable correlations exist among prediction errors, it may be possible to enhance the prediction model, perhaps by increasing the number of delays in the tapped delay lines.

In the case of the HYBRID model, most of the correlations, except for the one at zero lag, generally fell within the 95% confidence limits around zero, indicating that the model's performance was satisfactory (as depicted in Figure 5). However, isolated exceptions may suggest opportunities for improvement, either through retraining the network or by augmenting the number of neurons in the hidden layer.

For more precise results, retraining the network can be considered, as it would modify the initial weights and biases, potentially leading to an improved network

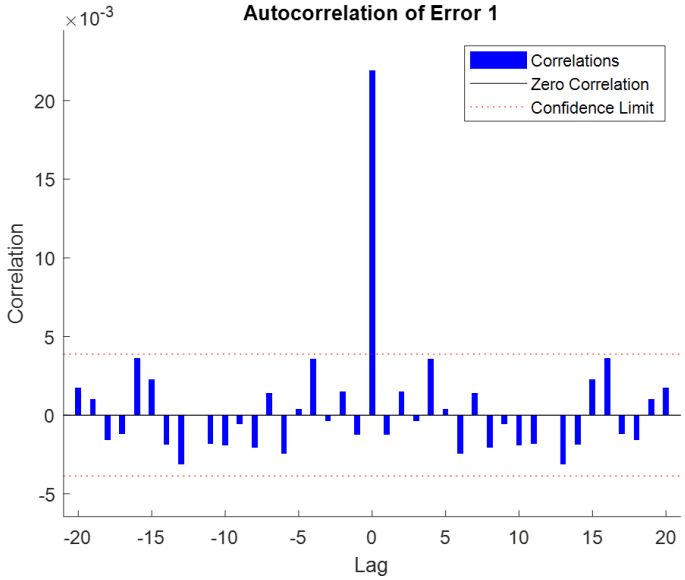


Figure 5: Error autocorrelation of the HYBRID model (MATLAB Output)

configuration. This approach can be particularly valuable when striving for even greater accuracy in predictions.

CONCLUSIONS

This study aimed to identify the most suitable model for analyzing the time series data of global average absolute sea level change spanning from 1880 to 2014. In this study, the suitable time series model was determined to be ARIMA(0,1,2) with a drift component, as it exhibited the lowest AIC values among the various models considered. Notably, this ARIMA(0,1,2) model with a drift component provided compelling evidence that future global average absolute sea level changes will follow an increasing trend over time. Additionally, this study evaluated the NARNN model, featuring 7 neurons in the hidden layer and 7 time delays, as the optimal neural network structure. This evaluation employed the LM algorithm, which is advantageous for its ability to expedite training without the need for exact Hessian matrix computations, ultimately ensuring stable convergence (Gavin, 2020).

Empirically, ARIMA and NARNN models have demonstrated their proficiency in modeling linear and nonlinear time series data, respectively. However, opting for a HYBRID model, which combines the strengths of both ARIMA and NARNN, offers a compelling advantage. This HYBRID approach seamlessly integrates linear and nonlinear modeling capabilities, making it superior choice for accurately modeling complex time series data. The comparative analysis unveiled that the HYBRID model, featuring 6 neurons in the hidden layer and 7 time delays, outperformed both the NARNN model, with 7 neurons in the hidden layer and 7 time delays (MSE = 0.0227), and the ARIMA(0,1,2) model with a drift component (MSE = 0.0390). The HYBRID model achieved the highest accuracy with a MSE of 0.0129, suggesting its superior predictive capability for global average absolute sea level change. According to the findings of this study, the HYBRID model not only enriches the information available but also holds significant importance for decision-making processes concerning the future impacts of global sea level rise.

Acknowledgment

This work is supported by the USDA National Institute of Food and Agriculture, Evans-Allen project [Accession # 7004162].

REFERENCES

Adamowski, J., Chan, H. F., Prasher, S. O., Ozga-Zielinski, B., & Sliusarieva, A. (2012). Comparison of multiple linear and nonlinear regression, autoregressive integrated moving average, artificial neural network, and wavelet artificial neural network methods for urban water demand forecasting in Montreal, Canada, *Water Resources Research, 48*, 1–14.

Australia's Commonwealth Scientific and Industrial Research Organisation. (2017). Global Average Absolute Sea Level Change, 1880–2014. https://www.cmar.csiro.au/sealevel/sl_data_cmar.html

Beale, M. H., Hagan, M. T., & Demuth, H. B. (2019). Deep Learning ToolboxTM: Getting Started Guide. The MathWorks, Inc.

Benrhmach, G., Namir, K., Namir, A., & Bouyaghroumni, J. (2020). Nonlinear autoregressive neural network and extended Kalman filters do prediction of financial time series. Journal of Applied Mathematics, 2020, Article ID 5057801, 1-6.

Box, G. E. P., Jenkins, G. M., Reinsel, G. C., & Ljung, G. M. (2016). Time Series Analysis: Forecasting and Control (5th ed.). John Wiley and Sons Inc.

DataHun. (2022). Global Average Absolute Sea Level Change, 1880–2014. https://datahub.io/core/sea-level-rise

Engle, R. (2001). GARCH 101: the use of ARCH/GARCH models in applied econometrics. *Journal of Economic Perspectives, 15*(4), 157-168.

Gardner, M.W., & Dorling, S. R. (1998). Artificial neural networks (the multilayer perceptron) - a review of applications in the atmospheric sciences. *Atmospheric Environment, 32*(14), 2627-2636.

Gavin, H. P. (2020). The Levenberg-Marquardt algorithm for nonlinear least squares curve-fitting problems. Department of Civil and Environmental Engineering Duke University. 19 pages. Retrieved from Duck University: http://people.duke.edu/~hpgavin/ce281/lm.pdf

Hagan, M. T., Demuth, H. B., Beale, M. H., & De Jesus, O. (2014). *Neural network design* (2nd ed.). pp. 12-19~12-22 and 13-12~13-14. Publisher: Martin Hagan.

Hill, T., O'Connor, M., & Remus, W. (1996). Neural network models for time series forecasts, *Management Science, 42*(7), 1082-1092.

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

Ljung, G. M., & Box, G. E. P. (1978). On a measure of lack of fit in time series models. *Biometrika, 65*(2), 297-303.

MacKay, D. J. C. (1992). Bayesian Interpolation. *Neural Computation, 4*(3), 415–447. https://doi.org/10.1162/neco.1992.4.3.415

Marquardt, D. W. (1963). An algorithm for least-squares estimation of nonlinear parameters. *Journal of the Society for Industrial and Applied Mathematics, 11*(2), 431-441.

Møller, M. F. (1993). A scaled conjugate gradient algorithm for fast supervised learning. *Neural Networks, 6*(4), 525-533. https://doi.org/10.1016/S0893-6080(05)80056-5

Montgomery, D. C., Jennings, C. L., & Kulahci, M. 2008. Introduction to Time Series Analysis and Forecasting. John Wiley & Sons. Inc.

Sariev, E., & Germano, G. (2020). Bayesian regularized artificial neural networks for the estimation of the probability of default. *Quantitative Finance, 20*(2), 311–328. https://doi.org/10.1080/14697688.2019.1633014

Tealab, A. (2018). Time series forecasting using artificial neural networks methodologies: *A systematic review, Future Computing and Informatics Journal, 3*(2), 334-340.

U.S. Environmental Protection Agency. (2022). Global Average Absolute Sea Level Change, 1880–2014. https://www.epa.gov/climate-indicators/climate-change-indicators-sea-level

Young, W. L. (1977). The Box-Jenkins approach to time series analysis and forecasting: principles and applications. RAIRO. *Recherche opérationnelle, 11*(2), 129-143. Retrieved from Numdam: http://www.numdam.org/article/RO_1977__11_2_129_0.pdf

Yue, Z., Songzheng, Z., & Tianshi, L. (2011). Regularization BP neural network model for predicting oil-gas drilling cost, *In Proceedings of the 2011 International Conference on Business Management and Electronic Information, 2*, 483–487, Guangzhou, China,

Zhang, G. P., Patuwo, B. E., & Hu, M. Y. (1998). Forecasting with artificial neural networks: The state of the art. *International Journal of Forecasting, 14*(1), 35-62.

Zhang, G. P. (2003). Time series forecasting using a hybrid ARIMA and neural network model. *Neurocomputing, 50*, 159-175.