Predicting the future of El-Niño Southern Oscillation using machine learning models

Koi R McArthur, William Sepahi, Zifei Liu, Paulina Bakos Lang
AMATH 445/900

Fall 2024

Abstract

The impact of El-Niño Southern Oscillation (ENSO) on global weather patterns combined with the impact of climate change on ENSO warrants the development of methods for predicting future ENSO trends. We look to resolve this issue by training three machine learning models: a neural ordinary differential equation (neural-ODE), a neural delay differential equation (neural-DDE), and a recurrent neural network (RNN), on ocean surface temperature, air temperature, precipitation and wind speed data. The RNN was effective at capturing short-term trends and demonstrated robust temporal continuity, providing reliable predictions for up to 24 months. However, in terms of the long term trends, further research must be conducted in order to accurately predict ENSO trends beyond this initial time period. While the neural-ODE and neural-DDE models were able to capture qualitative aspects of the ENSO dynamics for up to 15 years, the solutions diverged from the test data within one year of the simulation.

1 Introduction

El Niño Southern Oscillation (ENSO) is a nonperiodic, oscillatory pattern in sea surface temperature (SST) of the eastern equatorial Pacific Ocean. When SSTs are above average it is referred to as El Niño, and when SSTs are below average it is referred to as La Niña, these are known as the phases of ENSO. Oscillations take place on timescales of approximately two to seven years (Sarachik & Cane, 2010), though the length of the ENSO phases is difficult to predict given the complexity of the system. Yet, the prediction of future ENSO trends is of interest due to the global impact of the ENSO cycle (Sarachik & Cane, 2010).

Many systems in nature can be described by ordinary differential equations (ODEs) of the form

$$\frac{d\overrightarrow{x}}{dt} = \overrightarrow{f}(\overrightarrow{x}, t) \tag{1}$$

where t is time, $\overrightarrow{x}$ is the state vector, and $\overrightarrow{f}$ is some function which describes the dependence of the rate of change of $\overrightarrow{x}$ on $\overrightarrow{x}$ and t. Eq. (1) can be used to describe ODEs of order n, where $n > 1$ by adding the variables $\overrightarrow{x}_k, 2 < k \leq n$ to the state vector and setting their derivatives as $\overrightarrow{x}_k = \overrightarrow{x}_{k+1}$ for $k < n$. Using this procedure, all ODEs can be written in the form of Eq. (1). If $\overrightarrow{f}$ has no time dependence then Eq. (1) can be written as

$$\frac{d\overrightarrow{x}}{dt} = \overrightarrow{f}(\overrightarrow{x}) \tag{2}$$

and is called an autonomous system.

When $\overrightarrow{f}$ is known, it can be used to find future values of $\overrightarrow{x}$. However, in some cases a relation of the form (Eq. 1) may be assumed, without knowing the value of $\overrightarrow{f}$. The prediction of $\overrightarrow{f}(\overrightarrow{x})$ given time series data $\overrightarrow{x}(t_i)$, where i denotes different data points, is a problem that is not new to machine learning. Though, there are methods to write $\overrightarrow{f}$ as an explicit function of $\overrightarrow{x}$, such as sparse identification of dynamical systems (SINDy) (Brunton et al., 2016) and symbolic regression Koza (1992), these methods are not well suited for all systems of the form (Eq. 1). SINDy requires $\overrightarrow{f}$ to be sparse in the function space of $\overrightarrow{x}$ (Brunton et al., 2016), and symbolic regression is computationally expensive. In these cases, a neural-ODE (Chen et al., 2019) may be used to characterize Eq. (1). Neural-ODEs work by letting $\overrightarrow{f}$ be equal to a neural network, and using the adjoint method (Pontriagin, 1964) to train $\overrightarrow{f}$ to predict the next state vector from the current one using the formulation of an ODE.

The objective of this work is to predict future ENSO trends using machine learning algorithms, we consider recurrent neural networks RNNs, and neural ordinary differential equations.

2 Methodology

2.1 Preprocessing

The Earth's climate is a complicated system, and models of the Earth's climate range from simple toy models to large Earth System Models (ESMs) with coupled components acting across three spatial dimensions (Eyring et al., 2016). Many of these ESMs have produced two-dimensional historical reanalysis data for variables important to ENSO, such as precipitation and ocean temperature. For this project, we use a two-dimensional historical reanalysis of ocean temperature, atmosphere temperature, precipitation, and wind speed, at 1° spatial and 1 month temporal resolution from the Coupled Model Intercomparison Project Phase 6 (CMIP6) (Eyring et al., 2016) ensemble of models.

We consider the portion of the data confined to (-20.5 °,20.5 °) latitude and (120 °,280 °) longitude, which should capture most of the area relevant to ENSO. We consider the time period of 1930 to 2015. To remove annual trends from the data we compute a one year moving average of the data, and to adjust for climatology we subtract a previous 30 year rolling mean from the data as well. To reduce the dimensionality of the data we normalize the data at each grid cell and compute a proper orthogonal decomposition (POD) (Berkooz et al., 1993) for each variable being considered, only taking the first two projection coefficients and modes. We perform a feature importance analysis using SelectKBest with regression to determine the important features to be used in our models.

Additionally for the neural-ODE and neural-DDE models, to reduce noise in highly variable data such as wind speed, we apply a Gaussian filter with a standard deviation of the Gaussian kernel of 2. To increase the number of data points and further smoothen our data we take every 3 data points and use them to create a dataset with time steps every 0.01 a using a cubic spline interpolation. The high degree of data smoothing used in this project is justified by the fact that ENSO takes place on timescales of a couple years and prediction of noise on the one to couple month timescale is not the goal of this project.

2.2 Models

2.2.1 RNN

Recurrent Neural Networks (RNNs) were used to predict time series data for ENSO variables. By sequentially processing temporal data, the RNN captured short- and long-term dependencies. The RNN model was augmented with LSTM layers to improve memory retention and capture nonlinear relationships in the data.

Architecture:

Two **LSTM layers** each with 100 units:

- The first layer outputs sequences to pass temporal data to the next layer. The second layer outputs the final vector

Two **Dropout layers** included the to reduce over-fitting by randomly deactivating neurons during training. The dropout rate is 30%.

Two **Dense layers:**

- A hidden dense layer with 50 neurons and ReLU activation.
- A final dense layer with 1 neuron for predicting the next time step.

80% of data set allocated to training and 20% for testing. Our model trained using a batch size of 32 and 50 epochs, aiming to minimize the mean squared error (MSE) between predicted and actual values.

We used MinMaxScaler to normalize the datasets ranged between 0 to 1. Seasonal patterns were removed from the data using seasonal decomposition, and a Savitzky-Golay filter was applied to reduce noise. Time-series data were organized into overlapping sequences with a look-back window of 24 months for training the LSTM layers.

We used Adam optimizer for its adoptive learning capabilities, and dropout layers were included in the model architecture to reduce overfitting.

2.2.2 Neural-ODE

We assume that the time evolution of ocean temperature, atmosphere temperature, precipitation, and wind speed can be described by Eq. (2), for some unknown $\overrightarrow{f}(\overrightarrow{x})$. We train a neural-ODE, letting $\overrightarrow{f}$ be equal to a neural network, and we train the neural network using the adjoint method. Due to the complicated nature of ENSO, we try out three state vectors:

- The projection coefficients for the first mode of the PODs of the ocean temperature and precipitation data, and their derivatives, with dimension 4. The neural-ODE model for this state vector will be referred to as ODE-OP.
- The projection coefficients for the first mode of the PODs of the ocean temperature and wind speed data, and their derivatives, with dimension 4. The neural-ODE model for this state vector will be referred to as ODE-OW.
- The projection coefficients for the first mode of the PODs of the ocean temperature, precipitation, wind speed data, and their derivatives, with dimension 6. The neural-ODE model for this state vector will be referred to as ODE-OPW.

We choose to omit the atmosphere temperature dataset since it aligns too closely with the ocean temperature dataset, and we choose to test combinations of precipitation and wind speed projection coefficients as the feature importance analysis was inconclusive as to whether precipitation or wind speed was more important in predicting SST. The precipitation projection coefficient was deemed more important than the wind speed projection coefficient, but the derivative of the wind speed projection coefficient was deemed more important than the derivative of the precipitation projection coefficient.

We use a neural network with an input size and output size equal to the dimension of the state vector, and three hidden layers, with ReLU activation function after each layer, leading to a neural network with 33668 trainable parameters for ODE-OP and ODE-OW, and a neural network with 33926 trainable parameters for ODE-OPW. We train on the first 80 % of the time series data for 500 epochs with batch size of 256. We use torchdyn (Poli et al., 2020) to create and train our neural-ODE using the adjoint method, a tsit5 solver (Tsitouras, 2011), and absolute and relative tolerances of 10^{-3}.

2.2.3 Neural-DDE

ENSO has been modeled in the past using delay differential equations (Ghil et al., 2008). Delay differential equations are of the form

$$\frac{d\overrightarrow{x}}{dt} = \overrightarrow{f}(\overrightarrow{x}, \overrightarrow{x}_t, t) \tag{3}$$

where $\overrightarrow{x}_t = \{x(\tau), \tau < t\}$. We assume that the system does not depend on the entire history of $\overrightarrow{x}$, and can be described by

$$\frac{d\overrightarrow{x}}{dt} = \overrightarrow{f}(\overrightarrow{x}, \overrightarrow{x}_a) \tag{4}$$

where $\overrightarrow{x}_a = \{x(\tau), t - a \leq \tau < t\}, a > 0$, and we have assumed the system is autonomous. We wish to approximate $\overrightarrow{f}$ using a neural network, a task which becomes more complicated since $\overrightarrow{f}$ is now a function of $\overrightarrow{x}_a$, not just $\overrightarrow{x}$. However, we can work around this problem by writing Eq. (4) in the form of Eq. (2) and creating a neural-ODE to approximate $\overrightarrow{f}$.

The data is evenly spaced in time, with spacing Δt, and we assume a is given by some integer number of time steps $a = jk\Delta t, j \in \mathbb{N}, k \in \mathbb{N}$. We consider the vector

$$\overrightarrow{y}(t) = \begin{pmatrix} \overrightarrow{x}(t) \\ \overrightarrow{x}(t - j\Delta t) \\ \overrightarrow{x}(t - 2j\Delta t) \\ . \\ . \\ . \\ \overrightarrow{x}(t - jk\Delta t) \end{pmatrix} \tag{5}$$

we note that since our dataset is discrete in time, $\overrightarrow{y}$ contains the information from both $\overrightarrow{x}$ and $\overrightarrow{x_a}$. Using $\overrightarrow{y}$ as the state vector we get

$$\frac{d\overrightarrow{y}}{dt} = \overrightarrow{f_1}(\overrightarrow{y}) \tag{6}$$

where $\overrightarrow{f_1}(\overrightarrow{y}) = \overrightarrow{f}(\overrightarrow{x}, \overrightarrow{x}_a)$. This is the same form as Eq. (4) and we can solve for $\overrightarrow{f_1}$ using a neural-ODE.

We create three neural-DDE models, by taking the state vectors from ODE-OP, ODE-OW, and ODE-OPW, and computing the corresponding state vectors from Eq. (5), with $j = 1$, $k = 5$, and $\Delta t = 0.1\,\mathrm{a}$. We take only every 10 data points (this is why $\Delta t = 0.1\,\mathrm{a}$) and set $j = 1$, allowing the model to learn the simple shifting pattern of the state vector at the cost of increasing the time step. These neural-DDE models are referred to as DDE-OP, DDE-OW, and DDE-OPW respectively. The same neural network as in Section 2.2.2 is used (with more neurons in the hidden layers), as well as the train/test split, training epochs, solution methods, and solver tolerances. The larger dimension of the state vectors for the neural-DDEs and the increased neurons in the hidden layers means that the DDE-OP and DDE-OW models have 101524 trainable parameters and DDE-OPW has 102814 trainable parameters.

3 Results

3.1 Neural-ODE

We can evaluate the local success of the ODE by allowing it to advance the system one time step for each data point in the training and test sets and computing the MSE between the predicted and actual data. These MSEs are given in Table 1. In all three neural-ODE models there is a high degree of bias, with testing MSE more than an order of magnitude higher than training MSE. This is suggestive of the model overfitting to the training data.

Model	**MSE train**	**MSE test**
ODE-OP	$8.96 \cdot 10^{-5}$	$1.14 \cdot 10^{-3}$
ODE-OW	$9.37 \cdot 10^{-5}$	$1.77 \cdot 10^{-3}$
ODE-OPW	$5.72 \cdot 10^{-5}$	$1.02 \cdot 10^{-3}$
DDE-OP	$1.41 \cdot 10^{-3}$	$3.54 \cdot 10^{-2}$
DDE-OW	$1.13 \cdot 10^{-4}$	$6.38 \cdot 10^{-4}$
DDE-OPW	$9.83 \cdot 10^{-4}$	$2.20 \cdot 10^{-2}$

Table 1: Mean squared error of neural-ODE and neural-DDE models on training and test data.

We use the models to predict the last 20 % of the time series with an initial condition of the first data point in the test set. In all three models, the predicted solutions begin on the correct trajectory, but they quickly diverge from this trajectory. The models are unable to predict the onset of shifts between El Niño and La Niña, as well as the magnitude to which these variations deviate from normal SSTs. Furthermore, ODE-OW predicts oscillations of the projection coefficients for SSTs and precipitation that are on timescales smaller than that of ENSO and with smaller amplitude, and ODE-OP oscillates at timescales larger than expected for ENSO. Though the ODE-OPW solution quickly diverges from the test data, the solution remains bounded and oscillatory for the duration of the simulation, suggesting that ODE-OPW does a better job at learning the dynamics of the system.

3.2 Neural-DDE

As with the case of the neural-ODE, we can evaluate the local performance of the neural-DDE via calculating the MSE of predicted and actual data in the training and test sets. However, for the neural-DDE we only compute the MSE of the components of the state vector corresponding to the current time of the state vector. The MSEs for the neural-DDEs are given in Table 1. The MSE in the training data for the neural-DDEs is within an order of magnitude of the MSE for the neural-ODEs, except in the case of ODE-OP and DDE-OP. However, the time step for the neural-DDEs is ten times larger than the time step for the neural-DDEs and so DDE-OW and DDE-OPW outperform ODE-OP and ODE-OPW locally on the training data. The neural-DDEs perform similar on the test data than the neural-ODEs given the larger time step.

Like with the neural-ODEs, we use the neural-DDEs to predict the last 20 % of the timeseries data. Unlike ODE-OP, and ODE-OW, the neural-DDEs solutions remain oscillating and bounded over the course of the simulations, and DDE-OPW does a considerably better job at predicting the shifts in ENSO phases than any of the neural-ODE models.

3.3 RNN

Each figure compares the historical data (blue lines) with the RNN's predictions for the next 24 months (red lines), providing insight into the model's ability to forecast trends. The predictions shows us that RNN effectively predictions short-term trends for each variables. In all the cases, we observe that the model maintains great alignment with the historical data, indicating the robust

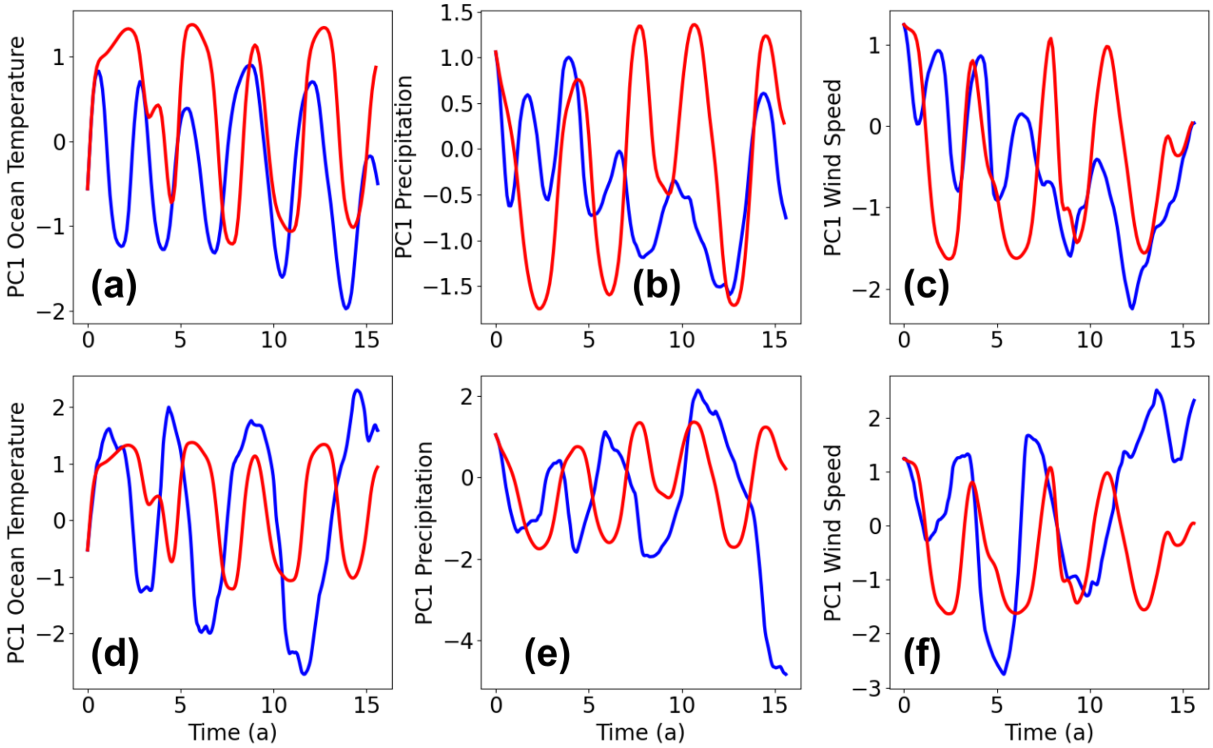


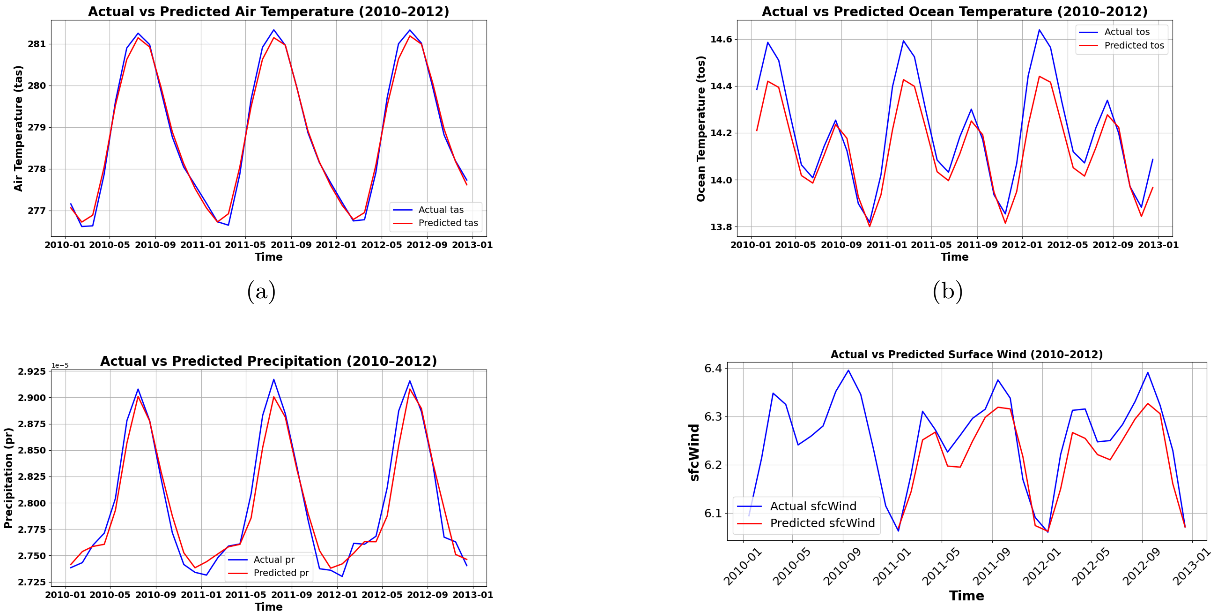
Figure 1: Predicted solutions on test data for ODE-OPW and DDE-OPW, predicted solutions in blue and test data in red. **(a)** first principal component of ocean temperature for DDE-OPW; **(b)** first principal component of precipitation for DDE-OPW; **(c)** first principal component of wind speed for DDE-OPW; **(d)** first principal component of ocean temperature for ODE-OPW; **(e)** first principal component of precipitation for ODE-OPW; **(f)** first principal component of wind speed for ODE-OPW.

temporal continuity of these variables. This demonstrates the RNN's strength in processing time-series data.

4 Discussion

The SST/wind speed relationship explored in ODE-OW and DDE-OW was only able to capture the qualititative behaviour of the ENSO dynamics (shift in ENSO phases, oscillatory nature, bounded solutions) in DDE-OW, meaning that wind speed and SST may evolve on different timescales. Given that ODE-OPW performed better than the other neural-ODE models and DDE-OPW performed better than the other neural-DDE models in terms of capturing the dynamics of the system, we conclude that there is an important relationship between wind speed and precipitation that helps to drive ENSO. All the models considered in this project included derivatives of the projection coefficients in the state vectors, meaning that the system was assumed to be second order.

In this work we use the neural-ODE framework to approximate delay differential equations through augmentation of the state vector, as opposed to a neural delay differential equation frame-



(a)

(b)

(c)

(d)

Figure 2: Actual vs Predicted values for various climate variables during 2010–2012. (a) Air Temperature, (b) Ocean Temperature, (c) Precipitation, and (d) Surface Wind.

work (Zhu et al., 2021) which extends the adjoint method while maintaining the same state vector. When j from Eq. (5) is set equal to 1, the model is able to learn the shifting nature of the state vector $\overrightarrow{y}$ and the historic data is readily available for prediction of the state vector at the next time step. With this approach, our neural-DDE models were able to outperform our neural-ODE models. However, the model is only able to learn the shifting nature of the state vector when $j = 1$, which limits the minimum time step of the model. Though the framework presented here is functional and easily implementable using the torcychdyn library (Poli et al., 2020), it is likely most useful for systems where the delay is small relative to the timescale of the system dynamics. However, we used a delay of 6 months, with oscillations between El Niño and La Niña occurring on timescales of as little as one year, and this produced decent results, showcasing the potential versatility of this method.

5 Conclusions

In this study, we explored the use of RNNs, Neural-ODEs, and Neural-DDEs to predict future trends of ENSO (El Niño Southern Oscillation) by analyzing key variables such as ocean surface temperature, air temperature, precipitation, and wind speed. We observed that all models effectively predicted short-term trends in ENSO. However, long term trends predicted by the neural-ODE and neural-DDE models diverged from the test data. Including wind speed and precipitation data in the neural-DDE framework allowed for a model that can capture the qualitative behavior of ENSO (bounded nonperiodic oscillatory motion) over the course of a 15 year simulation. Future work is needed to accurately predict long term ENSO trends.

References

Berkooz, G., Holmes, P., & Lumley, J. L. (1993). The proper orthogonal decomposition in the analysis of turbulent flows. *Annual review of fluid mechanics*, *25*(1), 539-575.

Brunton, S. L., Proctor, J. L., & Kutz, J. N. (2016). Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the National Academy of Sciences - PNAS*, *113*(15), 3932-3937.

Chen, R. T. Q., Rubanova, Y., Bettencourt, J., & Duvenaud, D. (2019). *Neural ordinary differential equations.* Retrieved from https://arxiv.org/abs/1806.07366

Eyring, V., Bony, S., Meehl, G. A., Senior, C. A., Stevens, B., Stouffer, R. J., & Taylor, K. E. (2016). Overview of the coupled model intercomparison project phase 6 (cmip6) experimental design and organization. *Geoscientific Model Development*, *9*(5), 1937–1958. Retrieved from https://gmd.copernicus.org/articles/9/1937/2016/ doi: 10.5194/gmd-9-1937-2016

Ghil, M., Zaliapin, I., & Thompson, S. (2008). A delay differential model of enso variability: parametric instability and the distribution of extremes. *Nonlinear processes in geophysics*, *15*(3), 417-433.

Koza, J. R. (1992). *Genetic programming : on the programming of computers by means of natural selection.* Cambridge, Mass: MIT Press.

Poli, M., Massaroli, S., Yamashita, A., Asama, H., & Park, J. (2020). Torchdyn: A neural differential equations library.

Pontriagin, L. S. L. S. (1964). *The mathematical theory of optimal processes.* Oxford: Pergamon Press; [distributed in the Western Hemisphere by Macmillan, New York].

Sarachik, E. S., & Cane, M. A. (2010). *The el nino-southern oscillation phenomenon.* Cambridge: Cambridge University Press.

Tsitouras, C. (2011). Runge–kutta pairs of order 5(4) satisfying only the first column simplifying assumption. *Computers mathematics with applications (1987)*, *62*(2), 770-775.

Zhu, Q., Guo, Y., & Lin, W. (2021). *Neural delay differential equations.* Retrieved from https://arxiv.org/abs/2102.10801