www.editada.org

International Journal of Combinatorial Optimization Problems and Informatics, 16(4), Sep-Dec 2025, 259-272. ISSN: 2007-1558. https://doi.org/10.61467/2007.1558.2025.v16i4.936

Nonlinear Systems Modeling Using Polynomial Neural Networks: Comparison

Guillermo Ronquillo Lomeli
Centro de Ingeniería y Desarrollo Industrial
gronquillo@cidesi.edu.mx

Abstract. Non-linear systems have not been extensively utilised in engineering research. This is a consequence of the lack of effective analytical approaches analogous to those developed for linear systems, which are well understood and readily analysable. This paper presents the application of two artificial neural networks for the online identification of non-linear systems as an alternative to linear systems theory. To illustrate this, the non-linear model of a single inverted pendulum on a cart was identified using experimental data obtained from a prototype. The model structure employed in this study was based on the Non-linear Autoregressive model with Exogenous Inputs (NARX). The networks were trained using a set of experimental data, and the validity of these models was confirmed through additional experimental recordings. The neural networks incorporating Chebyshev polynomials demonstrated marginally superior performance. The model of pendulum angular position achieved an R^2 value of 0.9984.

Keywords: Basic function; Nonlinear systems; Neural networks; Volterra polynomials; Chebyshev polynomials.

Article Info
Received Dec 26, 2016
Accepted Dec 11, 2017

1 Introduction

It is widely acknowledged that the inverted pendulum represents a diverse array of naturally unstable systems and serves as a benchmark problem in the fields of identification and control (Sarit & Kaustav, 2011; Wang, 2011; Y. X. Zhang et al., 2011; Zupančič & Sodja, 2013). Its applications extend beyond these domains, encompassing areas such as human body self-equilibrium analysis (AbuAli & Sabir, 2024; Kuo, 2007; Milton et al., 2009) and robot design technology (Erbatur & Kurt, 2009; Vanderborght, 2010; Xie et al., 2024), among others.

Despite the extensive development and application of the theory of nonlinear systems in the modeling of dynamic systems, they are, in general, complex and nonlinear (Jamsheed & Iqbal, 2023; Todorovic & Klan, 2006). The application of linearization methods is constrained to a limited operational range, necessitating the use of nonlinear modeling techniques. However, the identification of nonlinear systems is a significantly more challenging process than that of linear systems, as evidenced by the findings of (Liu, 2001). A review of the literature reveals that artificial neural networks (ANNs) have been utilized in nonlinear modeling for a considerable period of time due to their inherent characteristics, particularly their learning ability and their ability to accurately approximate nonlinear dynamics. The utilization of neural networks for nonlinear model identification has already been explored in several studies, including those by (Liu, 2001; Purwar et al., 2007; Ranković & Nikolić, 2008; Z. Zhang & Zhang, 2022).

The identification of a single inverted pendulum on a cart (SIPC) system has already been studied using neural networks in (Folgheraiter et al., 2023; Mohandas & Paritala, 2006) and (Sutradhar et al., 2010). The study by (Ronquillo-Lomeli et al., 2016) examines the multilayer perceptron (MLP) training method known as backpropagation, the networks were developed using Volterra polynomials in conjunction with the orthogonal least squares method for parameter estimation. In the study by (Orozco et al., 2015), the performance of MLP and Volterra polynomial basic function (VPBF) networks was compared. It should be noted, however, that the aforementioned papers employ only offline identification.

The applications of Volterra (Alessandrini et al., 2022; Liu, 2001; Liu et al., 1998; Patrikar & Provence, 1996) and Chebyshev (Li & He, 2010; Purwar et al., 2007; Z. Zhang & Zhang, 2022) polynomial expansions in neural networks have been the subject of considerable research. Each of these works presents a distinct approach to constructing the polynomial neural network. Consequently, it is challenging to ascertain which polynomial expansion (Volterra or Chebyshev) is more suitable for a given application.

A straightforward and straightforwardly implementable polynomial neural network structure for online system identification it is necessary. The nonlinear autoregressive with external input (NARX) model structure is a mathematical tool that is capable of modeling nonlinear relations and estimating the current output value through the use of a regression input vector (Billings, 2013; Liu, 2001; Norgaard et al., 2003). In the NARX model structure, the predictions are inherently stable due to the pure algebraic relationship between the prior measurements of the input and the output. This is of particular importance in the nonlinear case, as stability issues are inherently complex to analyze.

The present paper sets forth a comparison of two nonlinear model structures for the online modeling of a SIPC. The experimental data employed for the training and testing of the identified neural network models were obtained from a laboratory SIPC prototype. The proposed polynomial neural network structure is predicated on the input configuration of the linear ARX model, which is designated as NARX. In addition, the proposed structures were evaluated through the implementation of two distinct polynomial expansion methodologies: the Volterra and Chebyshev methodologies. These methodologies can be generated and implemented while maintaining the same structure, linear in their parameters and nonlinear in the inputs. This enables the consistent application of the same learning method. The learning algorithm of the proposed network is the least squares method in any of its variants. In this study, the recursive least squares (RLS) method is employed to introduce the exponential forgetting technique and to implement the algorithm in scenarios where parameters undergo temporal changes.

The remainder of this paper is organized as follows. In Section 2, a classical data-driven modeling methodology is presented. The methodology is comprised of the following phases: experimentation, selection of model structure, model train, and validation. As delineated in Section 3 of this text, the configuration of the model is established, thereby establishing the inputs and the structure of the nonlinear relationship. In Section 4, two neural network schemes are developed, with the basic of these schemes being nonlinear basic functions. Section 5 provides a comprehensive examination of the SIPC experimental prototype and the data acquisition system employed in the experimental context. The results of the implementation of the proposed methodology and the model performance measurement parameters can be found in section 6.

The polynomial neural network structure presented in this work, being a single-layer neural network, exhibits lower computational complexity compared to MLP and is suitable for online learning. The primary contribution of this article is the development of a methodology for modeling nonlinear systems with simple structures, thereby enhancing the accuracy of classical neural network schemes.

2 Methodology

While all systems are inherently nonlinear, the majority of extant literature on system modeling focuses on linear systems. The fundamental rationale underlying this phenomenon is that the underlying assumptions can become exceedingly limiting due to the intricacy of the process, compelling the designer to resort to rigorous simplifications or to rectify model components. Furthermore, the implementation of novel processes designed to improve performance, in conjunction with the presence of heterogeneous local environments, often leads to substantial discrepancies between two plants that appear to be analogous.

The experimental approach to process modeling is known as identification. The approach is comprised of the following steps: experimentation, model structure selection, training or parameter estimation, and model validation. The identification process, as based on experimental data, is illustrated schematically in Fig. 1.

In order to obtain data that accurately reflects the process's behavior, an experimental design must be performed across the entire range of the system's operation. In summary, it is imperative to ascertain the physical support values throughout the input range and to assess the impact on the output. Subsequently, a model is formulated from the measured input and output data. The experimental phase necessitates the delineation of several parameters, encompassing the formulation of test preparation, the determination of sampling time, the conceptualization of suitable experiments, and the undertaking of data preprocessing.

The process of defining the model framework is referred to as model structure selection. In order to select an appropriate model structure, it is necessary to consider the input and output signals of the model in question, as well as the internal interactions that occur within the model. The model framework is derived from a prior background. The selection of a model framework requires the identification of the model structure and the definition of the appropriate number of parameters for a given problem.

Subsequent to the selection of the model structure, it is imperative to estimate the values of the unknown parameters of a parameterized model framework. In most cases, the model that offers the most optimal performance for the intended application is selected. A multitude of methods exist for articulating design specifications. Therefore, it is recommended that the formulation of the model be considered at the outset, in conjunction with the application context. The selection of the model is made on the basic of the most accurate model predictions, with the criterion of the statistical properties of the error between the observed result and the model estimate being the primary consideration. The methodology for estimating the model parameters of the selected model framework is developed according to statistical theory and is referred to as estimation. In analogous processes within the framework of ANN theory, the analogous process is frequently referred to as training or learning.

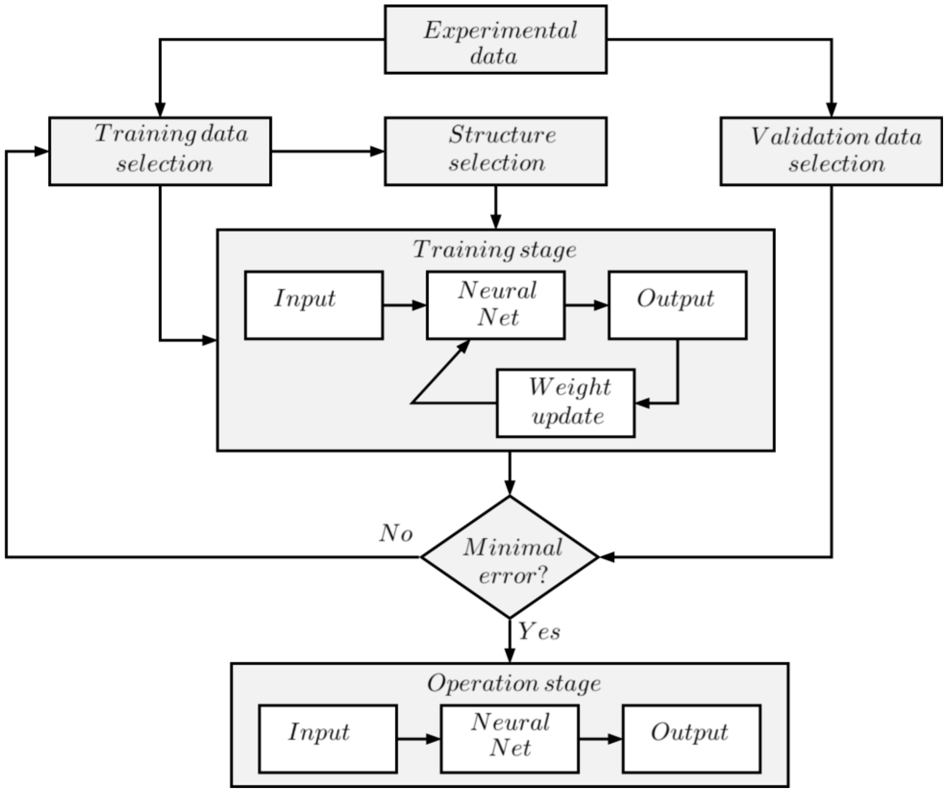


Fig. 1. Data-based approach for modeling.

Following the construction of the complete model, it is imperative that it undergoes rigorous testing to ascertain its conformity with the stipulated application requirements. This entails evaluating the model's accuracy, robustness, convergence, and good generalization (interpolation) abilities. The validation process of a model is inextricably linked to its practical application. The objective of this study is to facilitate a straightforward comparison of two model structures that are not contingent on data flow or operating conditions.

3 Structure of the NARX model

The ANN-based identification process is contingent upon the input data representing the system across its entire operational range and the learning algorithm incorporating the information contained in the data into the neural network model. In this context, the determination of the neural network is based less on the data flow or operational conditions and more on the selection of the neural network model structure, incorporation of variables, and selection of information for training and validation of the model. To illustrate, consider the nonlinear discrete system described by

$$X_{t+1} = G(X_t, u_t), \tag{1}$$

$$y_t = h(X_t, u_t), \tag{2}$$

where $G(\cdot)$ is a nonlinear vector function, $h(\cdot)$ a nonlinear function, X_t the state vector, y_t the output and u_t the input. In consideration of the input-output relationship of the system, the aforementioned nonlinear discrete system can be expressed by a nonlinear autoregressive model with external input (NARX) (Ranković & Nikolić, 2008), that is

$$y_t = f\left(y_{t-1}, y_{t-2}, \cdots, y_{t-n_y}, u_{t-1}, u_{t-2}, \cdots, u_{t-n_u}\right). \tag{3}$$

In this context, $f(\cdot)$ denotes a nonlinear function, with n_y and n_u denoting the corresponding maximum delays. The nonlinear function $f(\cdot)$ described in eq. (**3**) is approximated by a one-layer neural network comprising a linear combination of the basic functions ϕ_t

$$\hat{f}(x_t) = \sum_{k=1}^{N} \theta_k \varphi_k(x_t), \tag{4}$$

where $\mathrm{x_t} = [\mathrm{y_{t-1}}, \mathrm{y_{t-2}}, \cdots, \mathrm{y_{t-n_y}}, \mathrm{u_{t-1}}, \mathrm{u_{t-2}}, \cdots, \mathrm{u_{t-n_u}}]$, φ_k are the nonlinear basic functions, θ_k the weights of each connection, $\hat{f}(\cdot)$ the estimation of $f(\cdot)$ y N the nonlinear basic functions number.

The nonlinear function f(x) can be derived through the utilization of these polynomial network structures as follows

$$f(x_t) = \hat{f}(x_t) + \epsilon_t, \tag{5}$$

where ϵ_t is the approximation error.

The nonlinear identification scheme, which employs artificial neural networks, is depicted in Fig. 2.

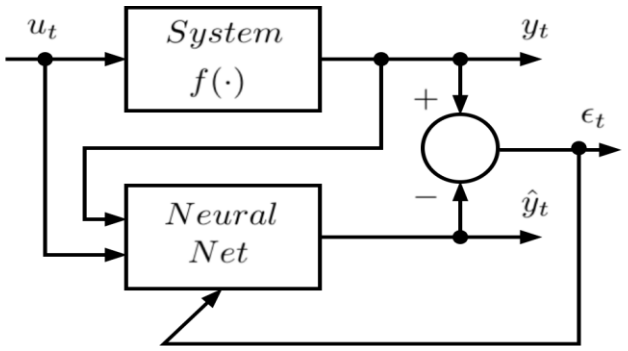


Fig. 2. Block diagram of nonlinear modeling using neural networks.

4 Neural network with polynomial basic functions

In accordance with the universal approximation theorem (Haykin, 1999), a finite number of basic functions exist for a neural network to accurately approximate the nonlinear function $f(\cdot)$. It has been demonstrated that any requisite approximation accuracy can be attained through the utilization of an adequate number of independent nonlinear basic functions (Yu, 2004). In this study, the basic functions employed are the Volterra and Chebyshev polynomials.

Based on the eq. (**4**) the relationship between the input and the output can be expressed in a compact vector form

$$y_t = \phi_t^T \Theta + \epsilon_t, \tag{6}$$

where the basic function vectors $\phi_t \in \mathbb{R}^N$ and parameter vector $\Theta \in \mathbb{R}^N$ are given by:

$$\phi_t = [\varphi_1(x_t), \varphi_2(x_t), \cdots, \varphi_N(x_t)]^T, \tag{7}$$

$$\Theta = [\theta_1(x_t), \theta_2(x_t), \cdots, \theta_N(x_t)], \tag{8}$$

the number of polynomial basic functions, denoted by N, is dependent upon the specific type of polynomial being considered, whether it be Volterra or Chebyshev.

The proposed structure employs a classical weight update law that is independent of the polynomial expansion utilized.
A graphical representation of eq. (**6**) is presented in Fig. 3.

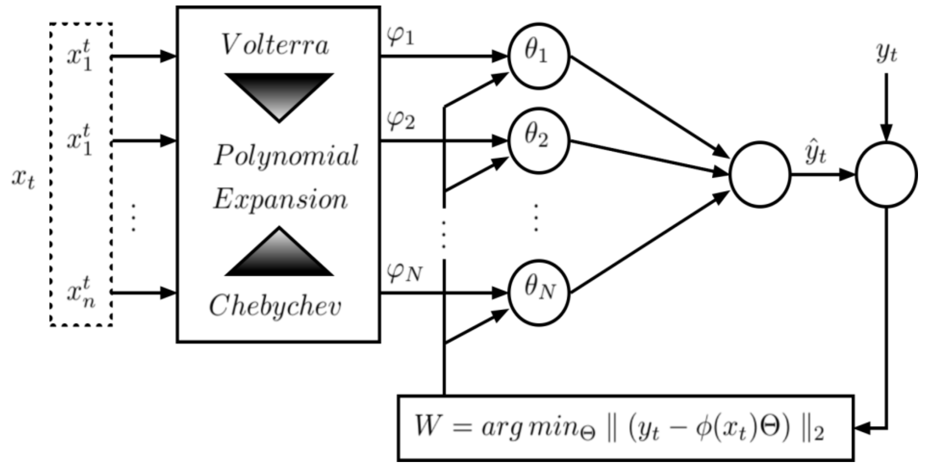


Fig. 3. Neural network model with polynomial basic functions.

The recursive least squares with exponential forgetting technique is employed to update the weights associated with Θ online, thus obviating the necessity to preprocess the input data for the neural network. (Purwar et al., 2007) presents stability and convergence in the Lyapunov sense for this method. The recursive MMC algorithm, as outlined by (Åström & Wittenmark, 1994), is

$$\epsilon_t = y_t - \phi_t^T \hat{\Theta}_{t-1}, \tag{9}$$

$$\hat{\Theta}_t = \hat{\Theta}_{t-1} + K_t \epsilon_t, \tag{10}$$

$$K_t = P_{t-1}\phi_t(\lambda I - \phi_t^T P_{t-1}\phi_t)^{-1}, \tag{11}$$

$$Pt = \frac{(I - K_t \phi_t^T)P_{t-1}}{\lambda}, \tag{12}$$

where $P_t \in \mathbb{R}^{N\times N}$ is the gains matrix and $K_t \in \mathbb{R}^{N\times 1}$ is the weight factor, λ the forgetting factor and I is the identity matrix. For $t = 1,2,\cdots, M$ with M the number of total samples.

Once the neural network has been defined, it is necessary to generate the functions by polynomial expansion. As previously indicated, in this work the Volterra polynomial and Chebyshev polynomials were used as basic functions.

4.1 Volterra polynomial basic functions

The VPBF is contingent upon the order o of the desired polynomial expansion and the number of entries, which is the sum of the number of inputs and outputs, $n = n_u + n_y$. The Volterra basic functions are determined from the expansion of the Volterra series as follows,

$$\begin{aligned}\phi_t &= \left[\varphi_1, \varphi_2, \varphi_3, \cdots, \varphi_{n_y+1}, \varphi_{n_y+2}, \cdots, \varphi_{n_y+n_u+1}, \varphi_{n_y+n_u+2}, \varphi_{n_y+n_u+3}, \cdots, \varphi_N\right](x_t) \\ &= \left[1, y_{t-1}, y_{t-2}, \cdots, y_{t-n_y}, u_{t-1}, \cdots, u_{t-n_u}, y_{t-1}^2, y_{t-1}y_{t-2}, \cdots, \varphi_{t-n_u}^o\right],\end{aligned} \tag{13}$$

where the number of basic functions N is determined by

$$N = \frac{(n_y + n_u + o)!}{o!\,(n_y + n_u)!}. \tag{14}$$

Considering that $\delta = x_t$ where $\delta = [\delta_1, \delta_2, \cdots, \delta_n]$ represents the state vector at a given time point t. The procedure for constructing the VPBFs Neural Network is shown in Algorithm 1.

Algorithm 1. Pseudo code for constructing the VPBFs Neural Network.

```
$n = n\_u + n\_y$
$k = 1$
$\varphi_k = 1$
$\Delta = x_t$
if $o \geq 1$ then
    for $i_1 = 1, \cdots, n$
        $k = k + 1$
        $\varphi_k = \delta_{i_1}$
    end for
end if
if $o \geq 2$ then
    for $i_1 = 1, \cdots, n$
        for $i_2 = i_1, \cdots, n$
            $k = k + 1$
            $\varphi_k = \delta_{i_1}\delta_{i_2}$
        end for
    end for
end if
if $o \geq 3$ then
    for $i_1 = 1, \cdots, n$
        for $i_2 = i_1, \cdots, n$
            for $i_3 = i_2, \cdots, n$
                $k = k + 1$
                $\varphi_k = \delta_{i_1}\delta_{i_2}\delta_{i_3}$
            end for
        end for
    end for
end if
and so, on
```

The vector of basic functions ϕ_t is concatenated as defined in eq. (**7**).

4.2 Chebyshev polynomial basic functions

Considering a scalar input δ the Chebyshev polynomial basic function (CPBF) is determined by the following recursive formula (Purwar et al., 2007).

$$T_{i+1}(\delta) = 2\delta T_i(\delta) - T_{i-1}(\delta), \tag{15}$$

where $T_0(\delta) = 1$, $T_1(\delta) = 1$ and each $T_i(x_t)$ is a Chebyshev polynomial, and the fundamental functions are determined in the following manner

$$\phi_t = [1, T_1(\delta), T_2(\delta), \cdots, T_o(\delta)], \tag{16}$$

in the case of n entries, the general form of the delta vector is given by $\Delta = [\delta_1, \delta_2, \cdots, \delta_n]$ and the polynomial Chebyshev expansion is

$$\phi_t = [1, T_1(\delta_1), T_2(\delta_1), \cdots, T_o(\delta_1), \cdots, T_1(\delta_n), T_2(\delta_n), \cdots, T_o(\delta_n)], \tag{17}$$

where $T_i(\delta_j)$ is a Chebyshev polynomial, the term i is used to denote the selected polynomial order and $j = 1, \cdots, n$ the position of the model input. The possible values of $T_1(\delta)$ are δ, 2δ, 2δ-1 and $2\delta + 1$. In the present study, the first-order Chebyshev polynomial $T_1(\delta)$ was selected as δ.

Assigning $\delta = x_t$ where $\delta = [\delta_1, \delta_2, \cdots, \delta_n]$ represents the state vector at time t.

The procedure for constructing the CPBF Neural Network is shown in Algorithm 2.

Algorithm 2. Pseudo code for constructing the CPBFs Neural Network.

```
$n = n_u + n_y$
$k = 1$
$T_0(x_t) = 1$
$\varphi_k = T_0(x_t)$
$\Delta = x_t$
for $j = 1, \cdots, n$
    $k = k + 1$
    for $i = 1, \cdots, o$ do
        if $i = 1$ then
            $T\_i(\delta_j) = \delta_j$
        else
            $T\_i(\delta_j) = 2\delta_j T_{i-1} - T_{i-2}$
        end if
        $\varphi_k = T_i(\delta_j)$
    end for
end for
```

The final number of CPBFs is

$$N = 1 + (n_y + n_u)o, \tag{18}$$

and vector of Chebyshev polynomial basic functions, denoted by ϕ_t, is constructed in accordance with the definition given in eq. (**7**).

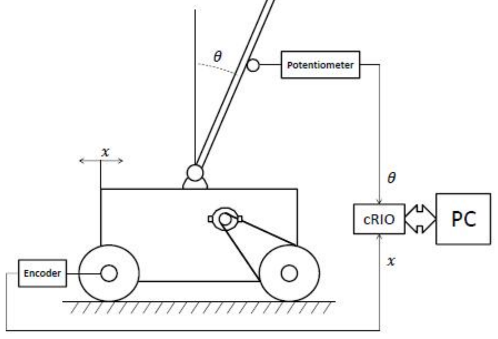
5 Experimental prototype

The experimental prototype for the implementation of the proposed methods is a mobile cart driven by a direct current (DC) motor. The motor is coupled by means of a belt and pulleys to generate mechanical traction and apply a force transversely on the x-axis. The x-axis is the input to the system. As illustrated in Fig. 4, the mobile cart is affixed with an inverted pendulum.

The system under consideration is a laboratory prototype model RT-124 manufactured by G.U.N.T. This model features a rotary potentiometer with the requisite voltage-angle position relationship, enabling it to function as a position and angular velocity sensor at the base of the inverted pendulum.

Additionally, the system incorporates an incremental encoder to measure the position and linear velocity of the carriage. The actuator is a direct current (DC) motor. The maximum deflection angle of the pendulum is constrained by physical limitations. The input force data and state variables $\theta, x, \dot{\theta}, \dot{x}$, which correspond to angular and linear position and angular and linear velocity, respectively, were acquired through a National Instruments cRIO-9074 and *LabVIEW* with embedded software for real-time acquisition.

The sampling interval was set at 10 milliseconds, a value consistent with Shannon's theorem. The mechanical time constant functioned as a reference point for the configuration under consideration. The data were stored in a text file on the hardware to ensure real-time data acquisition.



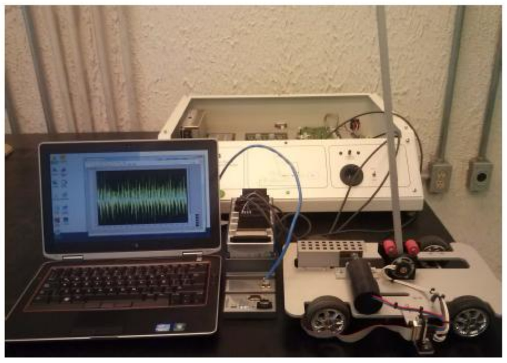


Fig. 4. Data acquisition system, hardware and software: Prototype schema (left) and Experimental prototype (right).

In order to identify the system based on data, it is essential to ensure that the excitation conditions are persistent. Furthermore, it is imperative that the system be managed in close proximity to its unstable operating point, that is to say, in an inverted position.

The incorporation of a fuzzy logic controller within the RT-124 control circuit enables the system to achieve both the operating range and the persistent excitation condition without the need for manual perturbations. As illustrated in Fig. 4 a) prototype schematic, b) the RT-124 system, NI cRIO, and the associated software are depicted.

The experimental training and test data acquired for modeling are presented in Fig. 5. Two data sets were utilized in the construction of the model. The initial 6,000 samples were utilized for model training, while the final 1,500 samples were employed for model validation.

6 Results

This section presents the results of the learning or training process in terms of mean squared error (MSE) and mean absolute error (MAE). The data presented in Fig. 5 for the time interval 0-60 seconds were applied to the VPBF and CPBF polynomial neural networks.

A total of eight identification experiments were conducted for each neural network, wherein the input-output delays ($n_u = n_y$) and other parameters specific to each neural network were modified. In the context of the VPBF and CPBF polynomial neural networks, the modified parameters were the polynomial order (o) and the forgetting factor (λ) and the input-output delays ($n_u = n_y$).

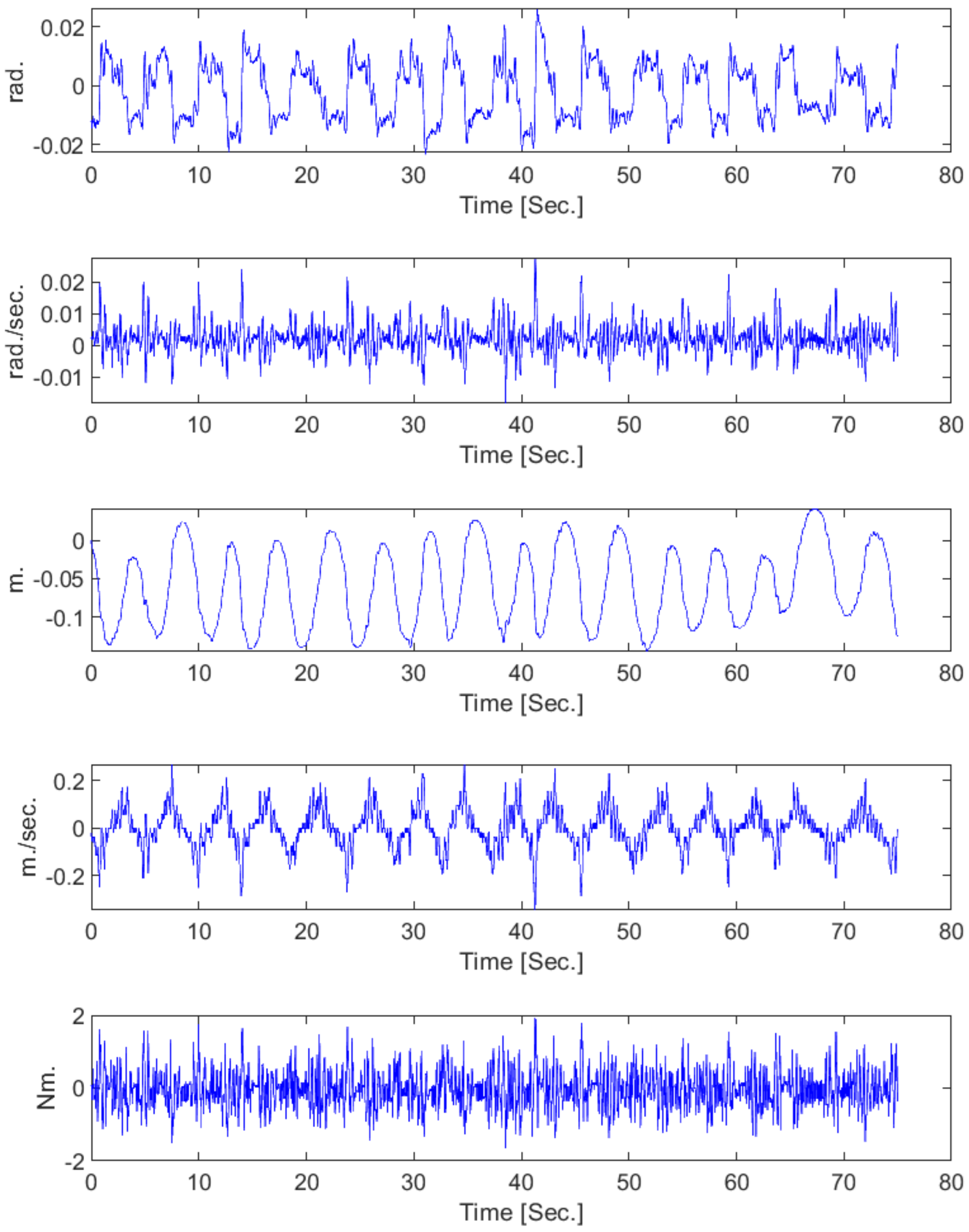


Fig. 5. Experimental data for models training from 0 to 60 seconds and testing from greater than 60 to 75 seconds: angular position, angular velocity, linear position, linear velocity, and linear force from top to bottom respectively.

In order to identify a simple model, it is proposed to evaluate combinations of models of order 2 and 3 with input lags of 2 and 4, in order to maintain simplicity. However, a model complexity balance between accuracy and complexity analysis is reserved for subsequent research endeavors.

The MSE and MAE results are presented in Table 1 and Table 2 for VPBF and the MSE and MAE results are presented in Table 3 and Table 4 for CPBF. It can be observed that the polynomial neural network, with CPBF polynomial expansions, exhibited superior performance. The computational complexity analysis of each network for the optimal performance (in bold) is presented in Table 5.

The best performance, with regard to computational complexity, is indicated in bold for each state variable. As evidenced in Table 1, Table 2, Table 3 and Table 4, the most favorable performance for each neural network was observed with two input-output delays.

Table 1. MSE of the VPBF neural network

Error		**Polynomial Order:** o			
MSE		2		3	
Delays:		**Forgetting factor:** ⋋		**Forgetting factor:** ⋋	
$n_u = n_y$		0.990	0.999	0.990	0.999
2	θ	**3.11**$\times10^{-7}$	2.37$\times10^{-7}$	1.21$\times10^{-6}$	2.44$\times10^{-7}$
	x	**1.07**$\times10^{-5}$	1.33$\times10^{-5}$	1.06$\times10^{+0}$	1.03$\times10^{-5}$
	$\dot{\theta}$	**9.24**$\times10^{-7}$	1.47$\times10^{-6}$	1.09$\times10^{-6}$	1.46$\times10^{-6}$
	$\dot{x}$	1.95$\times10^{-4}$	**1.82**$\times10^{-4}$	1.99$\times10^{-4}$	1.83$\times10^{-4}$
4	θ	4.26$\times10^{-7}$	3.05$\times10^{-7}$	3.68$\times10^{-5}$	2.87$\times10^{-7}$
	x	8.28$\times10^{-4}$	1.04$\times10^{-5}$	1.51$\times10^{+0}$	1.08$\times10^{-5}$
	$\dot{\theta}$	9.86$\times10^{-7}$	1.28$\times10^{-6}$	2.94$\times10^{-6}$	1.24$\times10^{-6}$
	$\dot{x}$	1.00$\times10^{+0}$	1.63$\times10^{-4}$	1.03$\times10^{+0}$	1.54$\times10^{-4}$

Table 2. MAE of the VPBF neural network

Error		**Polynomial Order:** o			
MAE		2		3	
Delays:		**Forgetting factor:** ⋋		**Forgetting factor:** ⋋	
$n_u = n_y$		0.990	0.999	0.990	0.999
2	θ	**3.33**$\times10^{-4}$	3.45$\times10^{-4}$	3.59$\times10^{-4}$	3.52$\times10^{-4}$
	x	**5.56**$\times10^{-4}$	8.01$\times10^{-4}$	1.48$\times10^{-1}$	7.78$\times10^{-4}$
	$\dot{\theta}$	**7.30**$\times10^{-4}$	8.90$\times10^{-4}$	7.85$\times10^{-4}$	8.92$\times10^{-4}$
	$\dot{x}$	9.40$\times10^{-3}$	**8.70**$\times10^{-3}$	9.40$\times10^{-3}$	8.80$\times10^{-3}$
4	θ	3.55$\times10^{-4}$	3.95$\times10^{-4}$	7.66$\times10^{-4}$	3.80$\times10^{-4}$
	x	1.60$\times10^{-3}$	7.34$\times10^{-4}$	1.50$\times10^{+1}$	7.18$\times10^{-4}$
	$\dot{\theta}$	7.53$\times10^{-4}$	8.36$\times10^{-4}$	1.10$\times10^{-3}$	8.32$\times10^{-4}$
	$\dot{x}$	4.14$\times10^{+0}$	8.90$\times10^{-3}$	4.25$\times10^{+0}$	8.70$\times10^{-3}$

Table 6 presents the Pearson's linear correlation coefficient and 95% confidence intervals for each neural network for performance evaluation of $\theta, x, \dot{\theta}, \dot{x}$ selected models.

The resultant correlation coefficients thus demonstrate the satisfactory performance of the state variables in relation to the validation data. The 95% confidence interval obtained indicates that the data dispersion is negligible among the four models.

Prior studies on the modeling of a SIPC using data-driven methodologies have yielded complex models that incur high computational costs and exhibit marginally lower accuracy compared to the results reported in this article. (Folgheraiter et al., 2023; Mohandas & Paritala, 2006) presents a model based on ARX, NNARX, and FNN structures with 200 inputs.

The optimal model demonstrates an MSE of 0.0048 in the angular position model and utilizes a few hundred basic functions. (Sutradhar et al., 2010) present a neural network of considerable complexity that demonstrates satisfactory performance, albeit with simulated data. The findings of (Ronquillo-Lomeli et al., 2016) and (Orozco et al., 2015) demonstrate results that are analogous results to those obtained in this study, albeit with 168 basic functions. The angular position model that has been determined to be optimal in the present study is characterized by an MSE of 2.20 x 10^{-7} and a total of nine basic functions.

The results demonstrate that the CPBF neural network exhibits the most optimal performance in terms of computational complexity, mean square error, mean absolute error, and correlation coefficient. The plots of the measured and estimated state variables with the CPBF neural network using validation data are presented in Fig. 6.

Table 3. MSE of the CPBF neural network

Error		**Polynomial Order:** o			
MSE		2		3	
Delays:		**Forgetting factor:** ⋋		**Forgetting factor:** ⋋	
$n_u = n_y$		0.990	0.999	0.990	0.999
2	θ	**2.20**$\times 10^{-7}$	2.38$\times 10^{-7}$	2.22$\times 10^{-7}$	2.22$\times 10^{-7}$
	x	1.40$\times 10^{-5}$	**1.03**$\times 10^{-5}$	1.30$\times 10^{-5}$	9.36$\times 10^{-6}$
	$\dot{\theta}$	**9.16**$\times 10^{-7}$	1.47$\times 10^{-6}$	9.38$\times 10^{-7}$	1.01$\times 10^{-6}$
	$\dot{x}$	1.87$\times 10^{-4}$	**1.82**$\times 10^{-4}$	1.91$\times 10^{-4}$	1.82$\times 10^{-4}$
4	θ	2.18$\times 10^{-7}$	3.07$\times 10^{-7}$	2.17$\times 10^{-7}$	2.25$\times 10^{-7}$
	x	1.23$\times 10^{-5}$	1.03$\times 10^{-5}$	1.22$\times 10^{-5}$	1.17$\times 10^{-5}$
	$\dot{\theta}$	8.53$\times 10^{-7}$	1.32$\times 10^{-6}$	9.01$\times 10^{-7}$	8.50$\times 10^{-7}$
	$\dot{x}$	1.74$\times 10^{-4}$	1.62$\times 10^{-4}$	1.81$\times 10^{-4}$	1.61$\times 10^{-4}$

Table 4. MAE of the CPBF neural network

Error		**Polynomial Order:** o			
MAE		2		3	
Delays:		**Forgetting factor:** ⋋		**Forgetting factor:** ⋋	
$n_u = n_y$		0.990	0.999	0.990	0.999
2	θ	**3.30**$\times 10^{-4}$	3.46$\times 10^{-4}$	3.38$\times 10^{-4}$	3.37$\times 10^{-4}$
	x	5.61$\times 10^{-4}$	**5.34**$\times 10^{-4}$	7.99$\times 10^{-4}$	5.72$\times 10^{-4}$
	$\dot{\theta}$	**7.28**$\times 10^{-4}$	8.89$\times 10^{-4}$	7.38$\times 10^{-4}$	7.39$\times 10^{-4}$
	$\dot{x}$	9.10$\times 10^{-3}$	**8.60**$\times 10^{-3}$	9.40$\times 10^{-3}$	8.70$\times 10^{-3}$
4	θ	3.25$\times 10^{-4}$	3.96$\times 10^{-4}$	3.25$\times 10^{-4}$	3.32$\times 10^{-4}$
	x	4.19$\times 10^{-4}$	7.21$\times 10^{-4}$	4.46$\times 10^{-4}$	4.40$\times 10^{-4}$
	$\dot{\theta}$	7.04$\times 10^{-4}$	8.46$\times 10^{-4}$	7.28$\times 10^{-4}$	6.88$\times 10^{-4}$
	$\dot{x}$	9.30$\times 10^{-3}$	8.80$\times 10^{-3}$	9.80$\times 10^{-3}$	8.90$\times 10^{-3}$

Table 5. Comparison of the computational complexity between VPBF and CPBF

Number of	**VPBF**	**CPBF**
Basic Functions	15	9
Weights	15	9
Net inputs	4	4

The findings of this study corroborate the robustness of the proposed model and demonstrate that structural complexity can be substantially minimized without compromising precision, and indeed, with the potential to enhance it. This renders the technology well-suited for deployment in real-time control applications or in settings with limited computational capacity.

As was initially proposed, the evidence demonstrates that the polynomial neural network structure, being a single-layer neural network, presents lower computational complexity compared to conventional multi-layer neural network architectures. This reduction in complexity can be attributed to the simplified topology of the structure.

In addition, the employment of polynomial basis functions provides the network with adequate representational capacity to comprehend the nonlinear dynamics of the system. This eliminates the requirement for excessively complex structures. The developed architecture is especially suitable for real-time processing and computational resource-constrained applications due to the balance it achieves between simplicity and accuracy.

Table 6. Model performance parameters

Variable	Correlation Coefficients		95% Confidence Interval	
	VPBF	CPBF	VPBF	CPBF
θ	0.9983	0.9984	[-7.59×10^{-5},3.13×10^{-5}]	[-8.52×10^{-5},4.12×10^{-5}]
x	0.9989	0.9998	[-3.98×10^{-5},1.47×10^{-5}]	[-4.24×10^{-5},1.24×10^{-5}]
$\dot{\theta}$	0.9580	0.9578	[-6.04×10^{-5},6.21×10^{-5}]	[-6.05×10^{-5},6.20×10^{-5}]
$\dot{x}$	0.9805	0.9806	[-9.34×10^{-4},4.34×10^{-4}]	[-9.16×10^{-4},4.50×10^{-4}]

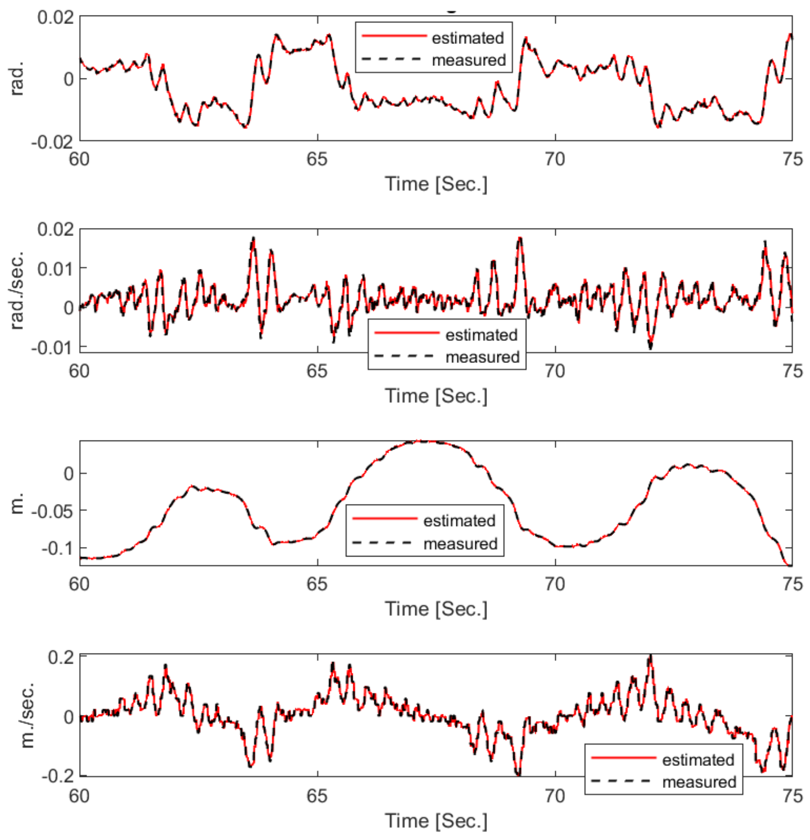


Fig. 6. State variables measured and estimated with the CPBF neural network: angular position, angular velocity, linear position, and linear velocity from top to bottom respectively.

7 Conclusions

In this study, two neural networks with polynomial basic functions were proposed for comparison. The model structure implemented is NARX. The CPBF neural network exhibited superior performance in terms of approximation error, computational complexity, and correlation when using the recursive MMC parameter estimation algorithm in comparison to the VPBF neural network. The proposed method allows for the interchangeability of the basic functions of the polynomial expansion without necessitating alterations to the network structure or the introduction of additional parameters. Furthermore, it incorporates a rapid and online weight update method. The proposed algorithm is straightforward to implement and requires minimal computational resources. The preprocessing of system data is not necessary, and no a priori knowledge is required. Neural networks with polynomial expansions as basic functions have been demonstrated to be efficacious. While MLP remains the most prevalent neural network, the present study demonstrates that a polynomial neural network can enhance results in engineering applications where the dynamics are complex, nonlinear, and change with time. Subsequent endeavors will entail the development of a methodology for online adaptation of the model structure. The objective of this methodology is to enhance or sustain model performance by incorporating or eliminating base functions or model inputs.

References

AbuAli, N., & Sabir, Z. (2024). Designing a heuristic computing structure to solve the human balancing model. *Journal of King Saud University - Computer and Information Sciences*, *36*(1). https://doi.org/10.1016/j.jksuci.2023.101890

Alessandrini, M., Falaschetti, L., Biagetti, G., Crippa, P., & Turchetti, C. (2022). Nonlinear Dynamic System Identification in the Spectral Domain Using Particle-Bernstein Polynomials. *Electronics (Switzerland)*, *11*(19). https://doi.org/10.3390/electronics11193100

Åström, K. J., & Wittenmark, B. (1994). *Adaptive Control* (2nd ed.). Addison-Wesley.

Billings, S. A. (2013). *Nonlinear System Identification: NARMAX Methods in the Time, Frequency, and Spatio-Temporal Domains*. Wiley.

Erbatur, K., & Kurt, O. (2009). Natural {ZMP} Trajectories for Biped Robot Reference Generation. *{IEEE} Transactions on Industrial Electronics*, *56*(3), 835–845. https://doi.org/10.1109/tie.2008.2005150

Folgheraiter, M., Yskak, A., & Yessirkepov, S. (2023). One-Shot Bipedal Robot Dynamics Identification With a Reservoir-Based RNN. *IEEE Access*, *11*, 50180 – 50194. https://doi.org/10.1109/ACCESS.2023.3277977

Haykin, S. (1999). Neural networks: a comprehensive foundation. In *The Knowledge Engineering Review* (second, Vol. 13, Issue 4). Prentice Hall International. https://doi.org/10.1017/S0269888998214044

Jamsheed, F., & Iqbal, S. J. (2023). Simplified artificial neural network based online adaptive control scheme for nonlinear systems. *Neural Computing and Applications*, *35*(1), 663 – 679. https://doi.org/10.1007/s00521-022-07760-x

Kuo, A. D. (2007). The six determinants of gait and the inverted pendulum analogy: A dynamic walking perspective. *Hum Mov Sci*, *26*(4), 617–656. https://doi.org/10.1016/j.humov.2007.04.003

Li, M., & He, Y. (2010). Nonlinear system identification using adaptive Chebyshev neural networks. *2010 {IEEE} International Conference on Intelligent Computing and Intelligent Systems*. https://doi.org/10.1109/icicisys.2010.5658578

Liu, G. P. (2001). Nonlinear Identification and Control. In *Advances in Industrial Control*. Springer London. https://doi.org/10.1007/978-1-4471-0345-5

Liu, G. P., Kadirkamanathan, V., & Billings, S. A. (1998). On-line identification of nonlinear systems using Volterra polynomial basis function neural networks. *Neural Networks*, *11*(9), 1645–1657. https://doi.org/10.1016/s0893-6080(98)00100-2

Milton, J., Cabrera, J. L., Ohira, T., Tajima, S., Tonosaki, Y., Eurich, C. W., & Campbell, S. A. (2009). The time-delayed inverted pendulum: implications for human balance control. *Chaos*, *19*, 26110.

Mohandas, K. P., & Paritala, S. (2006). SIMULATION OF INVERTED PENDULUM USING NEURAL NETWORKS FOR IDENTIFICATION. *Proceedings of the 4thFaculty of Architecture and Engineering Symposim*, *30*, 621–627.

Norgaard, M., Ravn, O., Poulsen, N. K., & Hansen, L. K. (2003). *Neural Networks for Modelling and Control of Dynamic Systems: A Practitioner's Handbook*. Springer London.

Orozco, L. M. L., Lomeli, G. R., Moreno, J. G. R., & Perea, M. T. (2015). Identification Inverted Pendulum System using Multilayer and Polynomial Neural Networks. *{IEEE} Latin America Transactions*, *13*(5), 1569–1576. https://doi.org/10.1109/tla.2015.7112017

Patrikar, A., & Provence, J. (1996). Nonlinear system identification and adaptive control using polynomial networks. *Mathematical and Computer Modelling*, *23*(1–2), 159–173. https://doi.org/10.1016/0895-7177(95)00225-1

Purwar, S., Kar, I. N., & Jha, A. N. (2007). On-line system identification of complex systems using Chebyshev neural networks. *Applied Soft Computing*, *7*(1), 364–372. https://doi.org/10.1016/j.asoc.2005.08.001

Ranković, V. M., & Nikolić, I. (2008). Identification of nonlinear models with Feedforward Neural Network and Digital Recurrent Network. *FME Transactions*, *36*(2), 87–92.

Ronquillo-Lomeli, G., Ríos-Moreno, G. J., Gómez-Espinosa, A., Morales-Hernández, L. A., & Trejo-Perea, M. (2016). Nonlinear identification of inverted pendulum system using Volterra polynomials. *Mechanics*

Based Design of Structures and Machines, *44*(1–2), 5–15. https://doi.org/10.1080/15397734.2015.1028551

Sarit, K. Das, & Kaustav, K. P. (2011). Robust compensation of a Cart–Inverted Pendulum system using a periodic controller: Experimental results. *Automatica*, *47*, 2543–2547.

Sutradhar, A., Sengupta, A., & Challa, V. R. (2010). Identification of servo-driven inverted pendulum system using neural network. *IEEE India Conference*, 1–4. https://doi.org/10.1109/INDCON.2010.5712589

Todorovic, N., & Klan, P. (2006). State of the Art in Nonlinear Dynamical System Identification using Artificial Neural Networks. *2006 8th Seminar on Neural Network Applications in Electrical Engineering*. https://doi.org/10.1109/neurel.2006.341187

Vanderborght, B. (2010). *Dynamic Stabilisation of the Biped Lucy Powered by Actuators with Controllable Stiffness*. Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-13417-3

Wang, J. J. (2011). Simulation studies of inverted pendulum based on PID controllers. *Simulation Modelling Practice and Theory*, *19*, 440–449.

Xie, Z., Wang, Y., Luo, X., Arpenti, P., Ruggiero, F., & Siciliano, B. (2024). Three-dimensional variable center of mass height biped walking using a new model and nonlinear model predictive control. *Mechanism and Machine Theory*, *197*. https://doi.org/10.1016/j.mechmachtheory.2024.105651

Yu, W. (2004). Nonlinear system identification using discrete-time recurrent neural networks with stable learning algorithms. *Information Sciences*, *158*, 131–147. https://doi.org/10.1016/j.ins.2003.08.002

Zhang, Y. X., Han, Z. J., & Xu, G. Q. (2011). Expansion of solution of an inverted pendulum system with time delay. *Applied Mathematics and Computation*, *217*, 6476–6489.

Zhang, Z., & Zhang, J. (2022). Chebyshev Functional Link Spline Neural Filter for Nonlinear Dynamic System Identification. *IEEE Transactions on Circuits and Systems II: Express Briefs*, *69*(3), 1907 – 1911. https://doi.org/10.1109/TCSII.2021.3111919

Zupančič, B., & Sodja, A. (2013). Computer-aided physical multi-domain modelling: Some experiences from education and industrial applications. *Simulation Modelling Practice and Theory*, *33*, 45–67.