

System Identification of Non-linear Inverted Pendulum with neural networks

Jaydeep Nandi, Asim Das, Priyanku Goswami

May 5, 2018

Abstract

System identification

1 Introduction

Hi

2 Inverted Pendulum System

3 Multi-layer Perceptrons

3.1 Introduction

A multi-layer perceptron (MLP) is a class of feed-forward artificial neural network. An MLP consists of at least three layers of nodes. Except for the input nodes, each node is a neuron that uses a non-linear activation function. MLP utilizes a supervised learning technique called backpropagation for training. Its multiple layers and non-linear activation distinguish MLP from a linear perceptron. It can distinguish data that is not linearly separable.

The general architecture of a MLP is shown in figure 1 It contains one output neuron, but the number of output neurons can easily be extended to be more than one.

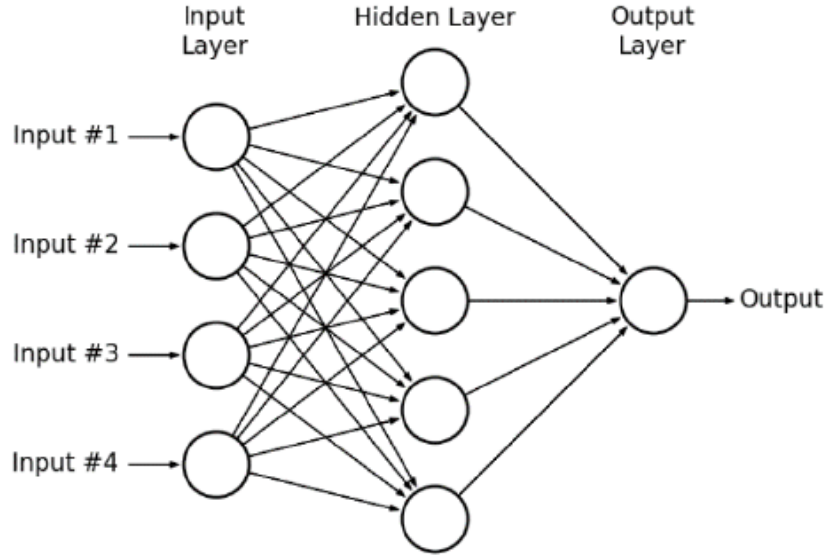


Figure 1: Architecture of Multi-layered perceptron

3.2 Activation function

The activation function of a neuron maps the output of a neuron based on the total input that the neuron has received (called the induced field of the neuron). The activation function can be linear or non-linear in nature.

Mathematically, the output y of j^{th} neuron is related to it's activation function $\phi(.)$ as:

$$y_j = \phi\left(\sum_{i=0}^n w_{ij}x_i\right) \quad (1)$$

Where:

$$w_{ij} = \text{weight of } j^{th} \text{ neuron connecting } i^{th} \text{ input}$$

$$x_i = i^{th} \text{ input in the input vector}$$

Different activation functions are used in practice. Some of the typical ones include:

3.2.1 Linear

Here,

$$\phi(x) = x$$

. This is used typically in the input layer, and also in output layer for regression tasks.

3.2.2 Sigmoid

Here

$$\phi(x) = \frac{1}{1 + e^{-\lambda x}}$$

. This lies in range $[0, 1]$. It is typically used in hidden layers in pattern classification tasks.

3.2.3 Hyperbolic Tangent

Here

$$\phi(x) = \tanh(\lambda x)$$

. It permits negative output in contrast to sigmoid, and is used in cases where a negative output is expected. This is the one we used in out hidden layers as activation function.

The characteristics of different activation functions are shown in figure 2

3.3 Learning

Learning refers to the process, where the neural network learns to update it's weights based on given input pattern set. Three distinct learning procedures exist:

3.3.1 *Supervised Learning*

Here, the network is presented with a pattern set $\tau\{X^k, D^k\}$, with input vectors X^k and corresponding target output vectors D^k , where k is the iteration index. The network learns the functional mapping


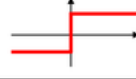
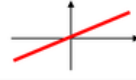
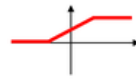


Activation function	Equation	Example	1D Graph
Unit step (Heaviside)	$\phi(z) = \begin{cases} 0, & z < 0, \\ 0.5, & z = 0, \\ 1, & z > 0, \end{cases}$	Perceptron variant	
Sign (Signum)	$\phi(z) = \begin{cases} -1, & z < 0, \\ 0, & z = 0, \\ 1, & z > 0, \end{cases}$	Perceptron variant	
Linear	$\phi(z) = z$	Adaline, linear regression	
Piece-wise linear	$\phi(z) = \begin{cases} 1, & z \geq \frac{1}{2}, \\ z + \frac{1}{2}, & -\frac{1}{2} < z < \frac{1}{2}, \\ 0, & z \leq -\frac{1}{2}, \end{cases}$	Support vector machine	
Logistic (sigmoid)	$\phi(z) = \frac{1}{1 + e^{-z}}$	Logistic regression, Multi-layer NN	
Hyperbolic tangent	$\phi(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$	Multi-layer NN	

Figure 2: A small list of activation functions

$f(X^k) = D^k$ through the learning algorithm. The learning process proceeds as follows:

1. The network is initialized with random weights, and the input pattern X_k is presented to the network, and the corresponding output y_k is noted.
2. The mean-squared error $e_k = \frac{1}{2} \sum_k (D_k - y_k)^2$ is calculated from target vector D_k .
3. The weights are updated using backpropagation algorithm, by propagating the error backwards. The details of the algorithm can be found in and is not discussed in detail here.

3.3.2 Reinforcement Learning

Here, the network is given a set of input vectors, and a reward, based on the action taken by the network. The network tries to maximize

the reward, through learning algorithm, typical one being Q-learning.

3.3.3 Unsupervised Learning

Unsupervised learning forces the network to learn from input data alone. Typically, the network is trained to cluster the inputs into different sets, by a distance measure.

MLPs are universal function approximators as showed by Cybenko's theorem, so they can be used to create mathematical models by regression analysis. As classification is a particular case of regression when the response variable is categorical, MLPs make good classifier algorithms.

3.4 Training a MLP in Matlab

Matlab R2017a provides a built-in `feedforwardnet` class, which can be used to create our simple two layer MLP network. The default activation functions are **Hyperbolic Tangent** in hidden layer, and **Linear** in output layer. The sample implementation we used is shown below:

```
1  % feedforwardnet is the net class
2  % 50 is the number of hidden layer neurons
3  net = feedforwardnet(50);
4  net.divideParam.trainRatio = 0.7;
5  net.divideParam.valRatio = 0.3;
6
7  %Train for 50 epochs
8  net.trainParam.epochs = 50;
9  net.trainParam.max_fail = 50;
10 net.trainParam.min_grad = 0;
11
12 % X is input vector; D is target vector
13 net = train(net, X', D');
```

After the training is over, the simulink block is formed by gensim command as follows

```
gensim(net)
```

The Simulink block can now be used for testing in simulink models.

4 Elman networks

Asim's part

5 System Identification

System identification is the process of developing a mathematical model of a dynamic system based on the input and output data from the actual process. This means it is possible to sample the input and output signals of a system and using this data generate a mathematical model. An important stage in control system design is the development of a mathematical model of the system to be controlled. In order to develop a controller, it must be possible to analyse the system to be controlled and this is done using a mathematical model. Another advantage of system identification is evident if the process is changed or modified. System identification allows the real system to be altered without having to calculate the dynamical equations and model the parameters again.

System identification is concerned with developing models. The inverted pendulum system is a non-linear process. To adequately model it, non-linear methods using neural networks must be used.

5.1 System Identification Procedure

Basically system identification is achieved by adjusting the parameters of the model until the model output is similar to the output of the real system. The basic procedure is modelled in figure 3

The main steps in system identification are:

1. The first step is to generate some experimental input/output data from the process we are trying to model. In the case of the

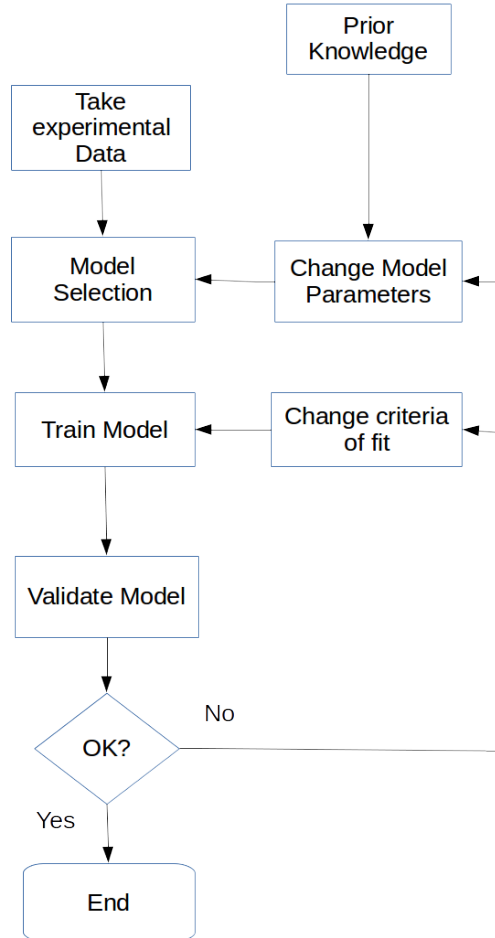


Figure 3: System Identification Procedure

inverted pendulum system this would be the input force on the cart and the output pendulum angle. The data is split into 70% training set and 15% validation set.

2. The next step is to choose a model structure to use. We shall use multi-layer perceptrons, and elman networks. The parameters are initialized to random values.
3. Next, the model is trained in a supervised fashion from the input/output data generated using backpropagation algorithm.

4. The network is validated against a validation set. If validation accuracy is good enough, then the model is accepted, else, the model parameters are changed, and the training is performed again.

5.2 Non-Linear Identification of the Process with neural networks

This section discusses the different methods of identifying the pendulum process using neural networks. The most common method of neural network identification is called forward modelling (shown in figure 4). During training both the process and ANN receive the same input, the outputs from the ANN and process are compared, this error signal is used to update the weights in the ANN. This is an example of supervised learning- the teacher (pendulum system) provides target values for the learner (the neural network).

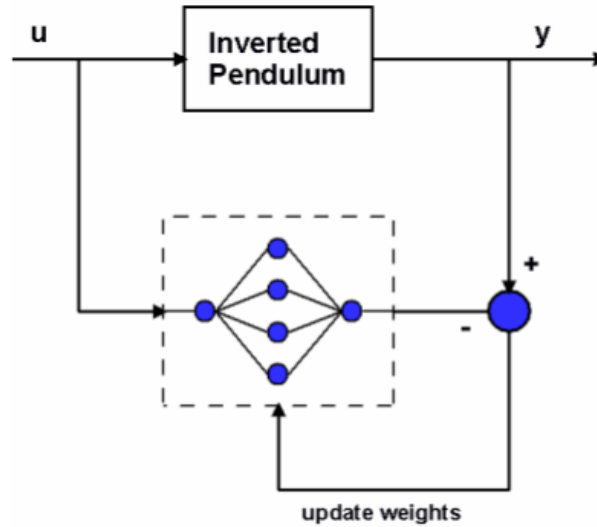


Figure 4: Forward modelling procedure (Taken from Callinan etal)

5.2.1 Generation of training data

The Simulink model of Pendulum+ Feedback Linearized controller is excited from a sine-wave, to which band-limited noise has been added.

This forms the input vector X . The Plant produces 4 state vectors: Position x , Velocity \dot{x} , Pendulum angle θ , Pendulum angular velocity $\dot{\theta}$. In the single output identification case, we consider only the angle θ as the output vector. The Simulink model of the process is shown below:

Thus, we get our training sample $\tau = X, D$. This will be used in training of the neural nets for system identification.

5.2.2 Identification with MLP

5.2.3 Identification with Elman nets

5.2.4 Multi-output Identification

6 Conclusion