Experimental Results

October 17, 2018

1 Effect of simulation fixed step size on the accuracy, architecture on neural network replacement of PID in dc motor speed control

1.1 Performance with 10 hidden neurons

Training Parameter: Tansig hidden layer, maximum epoch 150, stopping criteria epoch, minimum gradient, only the maximum epoch is changed everything else stays default. Such as input Delays = 1:2; feedback Delays = 1:2; hidden Layer Size = 10; With smaller step size, we have more data, and less error. Worst performance is shown in bold

Table 1: Effect of Fixed step size on Accuracy Testing Accuracy - Hidden Neurons 10

Fixed Step Size	Training Error	Test Error
0.001	2.4116e-06	1.9052e-06
0.01	2.3287e-06	1.8463e-06
0.1	7.4570e-05	6.1457e-05
0.2	**2.8739e-04**	**3.4440e-04**

1.2 Performance with 3 hidden neurons

Table 2: Effect of Fixed step size on Testing Accuracy - Hidden Neurons 3

Fixed Step Size	Training Error	Test Error
0.001	2.5779e-06	2.0565e-06
0.01	2.5016e-06	1.9960e-06
0.1	9.5324e-05	7.7856e-05
0.2	**3.6994e-04**	**4.0580e-04**

From the experiment we can conclude that a fixed step size of 0.01 will give better performance and the minimum allowed number of neurons in the hidden layer is 3.

2 Effect of Delay on the accuracy of neural network replacement of PID in DC motor speed control

Using the fixed step size obtained in 1, and the minimum hidden layer of 3 neurons, the following are the results for varying the delay. All other parameters are kept constant.

Table 3: Effect of Delay on Testing Accuracy

Delay	Training Error	Test Error
2	2.5016e-06	1.9960e-06
3	2.5152e-06	2.0026e-06
6	2.4874e-06	1.9868e-06
9	2.4872e-06	1.9871e-06
12	2.4856e-06	1.9859e-06

Observation: as the delay increases, the test accuracy gets better even though the difference is really small. However, the higher the value of the network delay, the more "complex" the architecture.

3 Write Up

The use of PID as a controller in control system applications is ubiquitous. Although the theory behind PID is really simple but the design and implementation of PID controllers are known to be time consuming and difficult. Moreover, PID are linear systems and can only learn a particular plant based on their predefined settings. NN on the other hand are characterised with learning opportunities and adaptability in nature. This means that a NN based controller is robust to several input types and even maybe different plant architectures as will be proved in this work.