Journal of Engineering and Applied Sciences 1 (1) : 1-6, 2006

Calibration Using Artificial Neural Networks

[1]H. Vasquez and [2]D.J. Fonseca
[1]Department of Mechanical Engineering, The University of Texas-Pan American
1201 W. University Drive, Edinburg, TX 78539
[2]Department of Industrial Engineering, The University of Alabama Box
870288 Tuscaloosa, AL 35487-0288. USA

Abstract: This study discusses the design and development of an Artificial Neural network (ANN) model to monitor the force applied to a strain-gage load cell. The reference voltage applied to a Wheatstone bridge formed by the strain gages, the amplification of the Wheatstone bridge's output voltage and the digitized value of the amplifier's output voltage acquired by a microprocessor represented the input to the ANN model. The output of the ANN was defined as the estimated value of the load acting on the load cell. In this study, a 5-3-1 neural network architecture proved to yield the best results, being the backpropagation Levenberg-Marquardt optimization algorithm the selected training paradigm. Based on the results obtained, it was concluded that neural networks offer a good option to calibrate an instrument, equipment, or system that operates under variable input conditions.

Key words: Artificial neural networks, load cell calibration, wheatstone bridge, levenberg-marquardt optimization

INTRODUCTION

A strain gage is an electrical device consisting of a flat coil with a nominal electrical resistance value of 120 or 350 Ω. The coil is built between thin insulating plastic layers and to assure electrical isolation between the gage wire and the metal on which the gage is installed, no less than 50 MΩ of electrical resistance must prevail after installation. The gage is installed on the surface of a mechanical part at the point and direction where the strain is to be measured.

An important property of strain gages is the gage factor, GF, which is the sensitivity of the gage. Such gage sensitivity is proportional to the ratio of the change in electrical resistance to the change in length or strain[1]. Knowing the gage factor, the strain is computed once the change in the gage's electrical resistance, ΔR, is measured. A Wheatstone bridge converts this change into resistance to a change in voltage. The bridge consists of four main resistors arranged in such a way that by measuring the output voltage, V_o and by knowing the input voltage V_i, the electrical resistance change, ΔR, in the strain gages can be estimated. In general, the output voltage, V_o, is not zero, even when there is not deformation of the strain gage because the resistors used to complete the bridge, or the gages themselves, are not always identical. The input voltage, V_i, must be low, approximately 5 volts, to reduce heat generation in the resistors[2]. There are ingenious ways to install and connect strain gages to a Wheatstone bridge to compensate for temperature changes or undesired cross sensitivity effects[1,3].

This study discusses the calibration of a load cell using a neural network model.The load cell was made with strain gages installed on a cantilever aluminum beam. The strain gages were connected to a Wheatstone bridge and the output signal of the bridge was amplified and collected using an analog-to-digital converter. The microprocessor transmitted the signal by infrared means to a computer. Although this type of weighing systems usually perform well, frequently, it is necessary some type of calibration because of inherent non-linearities and variation of parameters. These are some of the main reasons why a neural network was used for data correlation, as a counterpart of the traditional trial-and- error procedures currently in use. The neural network's most important task was to determine an accurate weight value regardless its relative location in the load cell range, considering any possible variations of its inputs. It wasthought by the authors of this study that such a neural network model might represent a viable means for calibrating such a system.

Corresponding Author: H. Vasquez, Department of Industrial Engineering, The University of Texas-Pan American 1201 W. University Drive, Edinburg, TX 78539, Tel: 956-292-7419 Fax: 965-3813527

The artificial neural network paradigm: In its most general form, an ANN can model the way in which the brain performs a particular function. The human brain is a highly complex, non-linear, and parallel information-processing system with the capability of performing certain computations (e.g., pattern recognition, perception and motor control) many times faster than the fastest digital computer in existence today. ANNs are simulated neurons interconnected in similar manner as the human brain's neurons.

There are three basic elements of an ANN model: a) a set of connecting links or *synapses*, each characterized by a *weight* of its own. Specifically, an input signal to the synapse j connected to neuron k is multiplied by the synaptic weight w_{kj}, b) an *adder*, which sums the input signals weighted by the respective synapses of the neuron and c) an *activation function,* which limits the permissible amplitude range of the output signal to some finite value. It defines the output of the neuron in terms of the induced local field, which is formed by the *linear combiner output* u_k and the *bias* b_k. This externally applied *bias* is used to increase or lower the net input of the activation function.

The ability to learn and improve its performance from examples is the ANN's fundamental trait. Haykin[4] defines as learning, in the ANNs context, the process by which the free parameters of the ANN are adapted through stimulation provided by the environment where the network is embedded. Instead of following a set of rules, ANNs are able to learn underlying relationships from a collection of training examples. ANNs are usually classified into two main categories: *recurrent* networks, in which loops occur because of feedback connections and *feed-forward* networks, in which the network structure has no loops. The choices of network architectures are intimately linked to the learning algorithm used in the training of the network.

Multilayered, feed-forward, non-linear network models are utilized for general-purpose and generalization applications. These types of networks are commonly known as Multilayer Perceptions (Maps). Maps have been successfully applied to solve diverse problems by training them in a supervised environment. The Back Error Propagation algorithm (BEEP) presented by[5], also known as the *Generalized Delta Rule,* is the most widely used supervised learning algorithm for MAPS. BEEP learns to generate a mapping from the input space to the output space by minimizing the error between the desired output and the actual output produced by the network. At each iteration, neurons slightly adjust their input connection weights in the direction that reduces their signal errors. This process is repeated for subsequent training patterns. Once trained, the MLP is able to identify features in input patterns and to produce meaningful outputs based on the detected presence (or absence) of those features in a new pattern. In short, BEEP is a *gradient-descent* technique, which is basically an iterative version of the simple Least-Squares Method (LSM), adapted to non-linear, multidimensional relationships.

ANN Applications in Systems Calibration: An important characteristic of an ANN is that all the neurons can be trained, under supervised or unsupervised learning, in an inter-dependent way, to associate, learn and/or classify information. In the past, neural networks have been used to calibrate, linearize, estimate, or fit data obtained from tracing the behavior of dynamic systems through sensors. A three-layered artificial neural network was used to calibrate a displacement sensor mechanism[6]. The results obtained were compared against curve fitting results and it was concluded that the ANN-based approach was the best alternative for the job.

Moreover, an ANN was used to monitor the behavior of a thermistor by linearizing its input-output relationship[7]. The developed network consisted of a feed-forward network with two hidden layers.

In another reported application of artificial neural networks to dynamic calibration, the force/torque sensors used by a robot to estimate the weight of an unknown payload was calibrated via a neural network[8]. This type of robotic calibration is critical to determine the correct grasping force to be applied by the robot's arm when picking up an object, so that slippage and product damage can be avoided.

A multi-layered ANN was also developed to calibrate a high-pressure measuring system with a non-monotonic behavior and which was greatly influenced by temperature changes. A better quality calibration than the one obtained through the conventional spline-based method of calibration was reported by the authors. The calibration obtained through the ANN was so effective that even for pressure reconstruction, temperature measurements were not necessary[9].

Finally, an ANN model was constructed for measurement calibration verification in power plants. As in many other industries, instruments used in power plants need to be re-calibrated on a periodic basis. An ANN was used to predict the reading of an instrument through readings from other dissimilar instruments. During this study, on-line parameter identification techniques and model-based observer methods were developed to assist in the training and testing of the constructed neural network model[7].

Most of the difficulty experienced during equipment calibration is due to inherent variability as well as the non-linear nature of the inputs and outputs involved in the process. The literature is clear in pointing out that ANNs can adequately be trained to learn and estimate non-linear data originated from measurements[10]. However, for the trained ANN to be useful, a high degree of repeatability of the system that generated the original data is required.

From the experiences found in the literature, it was determined that the problem at hand could be accurately solved using an ANN. An extension of this effort could be applicable to many other similar calibration situations. After facing some problems with calibration because of non-linearities and variations, it is possible that a particular piece of equipment can operate properly, with good sensitivity and repeatability, but not well calibrated. However, good signal processing would be required to compensate for its "bad" calibration. In this study, a neural network was used to tackle such a situation.

Description of the problem: The study involved the calibration of a load cell consisting of two strain gages installed on an aluminum cantilever beam with a rectangular cross sectional area. One of the strain gages was placed on the top surface while the other at the bottom surface of the beam, which was loaded in bending. Figure 1 shows the load, P, the cantilever beam and the top side strain gage. Calibration of the load cell was performed via a neural network. The main objective of the study was to design and train a neural network to correctly estimate the weight, or load P, applied to the load cell. Calibration data collected experimentally as well as from the mathematical formulation of the system was used to train the neural network.

Figure 1, it can be noted that when one of the gages is in tension, the other one is in compression; however, the strain on each gage is expected to reach the same magnitude. These strain gages, R_T and R_C in Fig. 2, are connected to a Wheatstone bridge. The bridge is completed by adding two resistors, R, with nominal values equal to those of the strain gages. The reference voltage of the Wheatstone bridge is V_i, which represents one of the inputs to the neural network model. The Wheatstone bridge's output voltage, V_o, depends on the deformation of the strain gages and on the input voltage to the bridge, V_i.

An electronic circuit was designed and implemented to condition and process the signal output from the bridge. First, an amplifier with a gain, K_{amp}, was introduced to make the signal more manageable by other

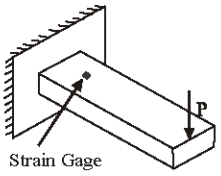

Fig. 1: Cantilever beam used as aload cell

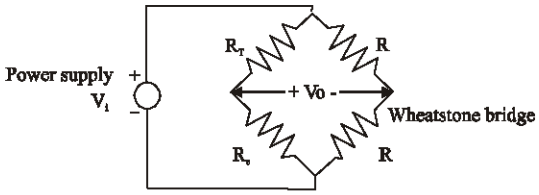

Fig. 2: Wheatstone bridge connections

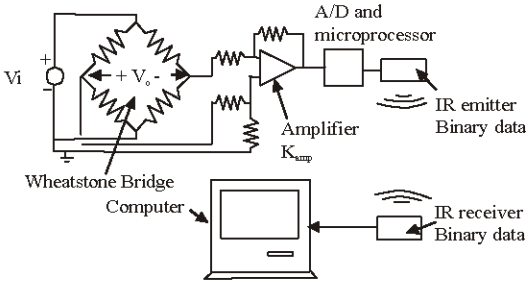

Fig. 3: Load cell signal acquisition and processing

devices. The gain, K_{amp}, is another of the ANN inputs since it can be adjusted and frequently measured to verify its value. This gain is not supposed to vary that much; but, when it changes, it has an important effect on the results. There was an analogical to digital (A/D) converter following the amplifier, which converted the amplified voltage to an integer value between 0 and 255 (one byte). This digitized value was another of the inputs to the neural network.

Figure 3 shows a more complete illustration of the load cell system where the neural network's inputs and output can be identified. The first input to the network is the amplification value, which is called K_{amp}; the second input is the Wheatstone Bridge's reference voltage, V_i; and the third input is the digitized amplifier's output

voltage value. The amplifier's output is basically the Wheatstone bridge's output voltage, V_o, multiplied by K_{amp}. This digitized value was obtained with an A/D converter driven by a microprocessor. The microprocessor sends the information serially via infrared to a computer located at a certain distance, 3 or 4 meters, from the measuring system. After that, the computer uses the trained neural network to compute the weight or load acting on the load cell.

In short, the Wheatstone bridge converts the weight acting on the load cell to a voltage, which is then amplified and digitized. Theoretically, it is expected that the digitized value be proportional to the product of the weight applied to the load cell, the amplification value and the Wheatstone bridge's reference voltage.

Description of the ANN model: The weighing system described above is made of several components, each introducing some uncertainty and error. Thus, the weight being measured is difficult to determine accurately throughout the load cell's operating range. This occurs because of variations of the reference voltage applied to the Wheatstone bridge, variations in the amplifier's gain and due to other nonlinear effects that are inherent to the system.

Inputs to the neural network: There are three major parameters that were selected as inputs to the neural network model:

1- Amplification: K_{amp}
2- Input Voltage to the Wheatstone bridge: V_i
3- Binary information serially transmitted: V_d

The inputs to the ANN were formalized as shown in Eq. 1:

$$\text{input\_vector} = \begin{bmatrix} K_{amp} \\ V_i \\ V_d \\ K_{amp} \times V_i \\ K_{amp} \times V_i \times V_d \end{bmatrix} \quad (1)$$

The input vector is 5x1 in size and its last two terms are a product combination of the first three; therefore, only the first three inputs are needed since the model's code takes care of adding the other two terms as well as normalizing the all input vectors. The reason behind including these last two terms originates from previous experimentation where the results obtained with only the first three terms were not satisfactory. Furthermore, from a theoretical stand, it is expected that the output from the ANN be proportional to the mutual product of the first three input terms. The input vectors were normalized by dividing each one by its corresponding magnitude.

Outputs from the neural network: The network has a single output, which is the value of the weight or load applied to the load cell, which obviously is a function of the three main inputs:

$$\text{weight} = f\{K_{amp}, V_i V_d\} \quad (2)$$

The output values generated from the training of the neural network were linearly scaled to values between 0 and 1. As it was later determined, the function in Eq. 2 is a combination of addition and multiplication of the three inputs. The main purpose of the neural network was to determine the best function to map the weight surface as a function of the inputs.

Training examples: Training examples were experimentally determined through the use of several different known weights, combined with four different reference voltages, V_i, applied to the Wheatstone bridge and several amplification gains, K_{amp}. A total of 154 training input-output pairs were generated, as shown in Table 1. All the sample data were used to train the network.

Test Samples: Once the training of the neural network model was completed, test input cases were fed into the network to evaluate its performance. These test samples are presented in Table 2.As it was done for the input training examples, any input vector presented to the neural network required two additional terms as defined in Eq. 1. The vectors were also normalized before being fed into the neural network.

Neural Network Architecture and Training: Sigmoidal transfer functions were chosen for the middle layers while a linear transfer function was used in the output layer of the network. The training examples were normalized to have proper values to work with these transfer functions. Training was performed using back propagation with the Levenberg-Marquardt (BPLM) algorithm as numerical optimization technique for relatively fast convergence[11]. The advantage of the Levenberg-Marquardt technique is that the results always go along the minimization of the error surface. However, the most significant disadvantage of this technique is the computation of the modified pseudo-inverse of the Jacobian, $(J^TJ + \mu_k I)^{-1}$, for every batch iteration. Where J is the Jacobian and μ_k is a variable coefficient that is changed to adapt the algorithm to the progress accomplished after every iteration. In some instances, this size of the matrix is so large that computing its inverse becomes extremely complicated.

Table 1: Training examples

	Kamp=200	Kamp=300	Kamp=401	Kamp=250	Kamp=200	Kamp=300	Kamp=200
	Vin=5.01V	Vin=9.04	Vin=5.01V	Vin=12.05	Vin=7.45	Vin=12.05	Vin=9.04
Weighty (1b)	Vd	Vd	Vd	Vd	Vd	Vd	Vd
0.00	30	74	60	79	41	62	63
0.51	33	82	66	88	46	68	70
1.10	36	91	73	99	50	75	79
1.62	39	98	79	1.6	54	82	85
2.07	42	105	84	115	58	87	92

to be Countinue

	Kamp=200	Kamp=800	Kamp=401	Kamp=200	Kamp=300	Kamp=40	Kamp=300
	Vin=9.04	Vin=5.01V	Vin=9.04	Vin=12.05	Vin=12.05	Vin=7.45	Vin=5.01
Weighty (1b)	Vd	Vd	Vd	Vd	Vd	Vd	Vd
2.58	45	113	89	124	62	93	99
3.10	48	121	95	133	66	99	106
3.62	51	129	101	140	70	105	112
4.14	54	136	106	150	74	111	120
4.66	56	144	112	158	78	117	126
5.11	59	151	117	165	81	122	132

Table 2: Test sample

Test#	1	2	3	4	5	6	7	8	9	10
Kamp	250	200	350	350	250	350	250	350	300	401
Vi	5.01	6.23	9.04	7.45	5.01	9.04	7.45	7.45	9.75	8.25
Vd	38	35	106	88	53	141	92	143	100	120
Weight (1b)	0.00	0.00	1.10	1.10	2.07	3.10	4.14	5.11	1.68	1.72

Table 3: Neural network result to the Test Sample for a 5-3-1

ANN result	Expected result	%Error
-0.088	0	
-0.010	0	
1.15	1.10	5.2
1.07	1.10	-1.9
2.04	2.07	-1.1
3.17	3.10	2.3
4.08	4.14	-1.2
4.98	5.11	-2.5
1.32	1.39	-4.9
1.61	1.72	-6.4

For example, for a 3-9-1 network, the corresponding matrix has a size of 46x46. Fortunately, nowadays, the computational power required to perform such massive computations is widely available to researchers.

The backpropagation algorithm with momentum and variable learning rate, VLBP, was originally implemented in the model. However, it proved to be inadequate for the problem at hand due to the required adjustments of parameters which led to convergence to stationary points that were not the desired minimum[11]. Thus, the Levenberg-Marquardt algorithm was mostly applied to optimize the convergence of the network. The training data shown in Table 1 was fed into the network in a row by row sequence. The total squared error was computed per each epoch, triggering the start of the back propagation learning.

RESULTS

A small number of neurons, between 4 and 5, was used at first in a single middle layer to train the neural network. However, progressively, more or fewer neurons were used as further tests were performed. The number of hidden layers was also constantly modified as the training progressed. While training was being performed, the results were displayed every one hundred epochs by means of a set of three graphs as depicted in Figure 4. The graph at the top displayed the convergence of the network's results to the training examples, the graph in the middle showed the squared error per epoch and the bottom graph provided data on the convergence of the network to the test samples.

The best results were obtained with a 5-3-1 neural network with sigmoidal middle layer transfer functions and a linear transfer function for the output layer. Figure 4 shows the network's results for epochs 5,100 through 5,400.

The minimum squared error per epoch was 0.025. The Jacobian matrix started getting close to be singular and the program could not compute its pseudo-inverse. It was determined that the training examples as well as the

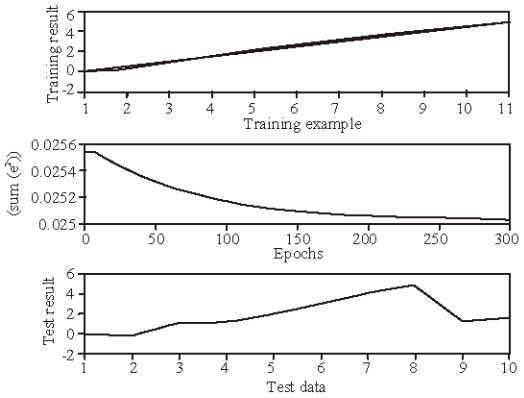

Fig. 4: Neural network 5-3-1

test examples were well approximated. The top graph of Fig. 4 represents the convergence of the neural network results to the training examples listed in columns 2 and 12 of Table 1. The horizontal axis of this graph represents the input vector according to its position in both columns. The "x" marks represent the expected weight values (Column 1 values). The top graph reveals that the two curves converged to the same values of weights in several regions of the graph. The bottom graph presents the results obtained using the test samples from Table 2. The horizontal axis of this graph represents the input vector according to its position in the columns of Table 2. The "x" marks represent the expected weight value associated with the values of Row 5 of Table 2. Again, the neural network generated an acceptable approximation of the test samples, with errors under 6.4% for all of the test points, as shown in Table 3. This error is acceptable, but there is always a desired to reduce it.

A final step was taken to decrease the number of neurons in the second layer to 2 instead of 3. However, the results obtained were not as good as the ones produced by the 5-3-1 neural network.

CONCLUSIONS

The paper described the designing, training and testing of a neural network model to calibrate a load cell made with strain gages installed on an aluminum beam. Two strain gages and two external resistors formed a Wheatstone bridge and the output signal from the bridge was first amplified and later acquired using an analog-to-digital converter, which was controlled by a microprocessor. The inputs to the neural network were the reference voltage applied to the Wheatstone bridge, the amplification gain used for applied to the Wheatstone bridge's output voltage and the digitized voltage value acquired by a microprocessor. The network's output was the estimated value of the weight applied to the load cell.

The network's main objective was to learn an accurate input-output relationship of the variables involved in the load cell system. The best results were achieved using a 5-3-1 ANN configuration. The backpropagation Levenberg-Marquardt (BPLM) algorithm was used to train the network and a minimum squared error per epoch of 0.025 was achieved for the 154 training examples. The network was tested and errors under 6.4% were obtained for all the test samples. Training of the neural network via a BPLM algorithm took approximately 25 minutes on a Pentium III Personal Computer. Based on the results obtained in this study, it was concluded that neural networks offer a good viable means to calibrate and correlate input-output data of equipment that operate under multiple variable inputs.

REFERENCES

1. Dally, J. and W. Riley, 1991. Experimental Strees Analysis 3ed Edition. New York: McGraw Hill.
2. Shaw, M., 1984. Metal Cutting Principles. New York: Oxford.
3. Joo, J.W., K.S. Na. and D.I. Kang, 2002. Design and Evaluation of a Six-Component Load Cell. Measurement. 32: 125-133.
4. Haykin, S., 1999. Neural Networks. New Jersey, NY: Prentice-Hall.
5. Rumelhart, D., G. Hinton and R. Williams, 1996. Learning Representations by Error Propagation. Parallel Distributed Processing, 1, 318-362.
6. Masory, O. and A. Aguirre, 1990. Neural network calibrates a displacement sensor. sensors, 7: 48-54.
7. Masory, O. and A. Aguirre, 1990. Sensor Calibration methods – a performance study. International society for optical engineering. Applications of Neural Networks.12: 490-501.
8. Attari, M. and F. Boudjema, 1995. Linearizing a thermistor characteristic in the range of 0 to 100 degree c with two- layered ANN. IEEE Transactions on Instrumentation and Measurement. 4: 119-122.
9. Garg, D. and S. Ananthraman, 1991. Sensor Integration for Payload Weight Estimation in a Robotic Work Cell. *ASME*, 30: 25-29.
10. Ipakchi, A. and M. Khadem, 1991. Neural Network applications to measurement calibration verification in power plants. Proceedings Instrumentation in the Power Industry. pp: 501-511.
11. Poopalasingam, S., C.R. Reeves and N.C. Steele1994. Application of neural networks for sensor performance improvement, Proceedings of the Ieee Workshop, pp: 633-640.
12. Hagan, M., H. Demuth and M. Beale, 1995. Neural Network Design. Boston: PWS Publishing.