Hindawi Publishing Corporation
Advances in Artificial Neural Systems
Volume 2009, Article ID 942697, 9 pages
doi:10.1155/2009/942697

Research Article

Design of Adaptive Filter Using Jordan/Elman Neural Network in a Typical EMG Signal Noise Removal

V. R. Mankar[1] and A. A. Ghatol[2]

[1] *Electronics Department, Government Polytechnic, Amravati, (M.S.) 444 604, India*
[2] *Technological University, Lonere, Dist. Raigarh, (M.S.), India*

Correspondence should be addressed to V. R. Mankar, vr_mankar@rediffmail.com

Received 28 July 2008; Revised 24 November 2008; Accepted 3 February 2009

Recommended by Yasar Becerikli

The bioelectric potentials associated with muscle activity constitute the electromyogram (EMG). These EMG signals are low-frequency and lower-magnitude signals. In this paper, it is presented that Jordan/Elman neural network can be effectively used for EMG signal noise removal, which is a typical nonlinear multivariable regression problem, as compared with other types of neural networks. Different neural network (NN) models with varying parameters were considered for the design of adaptive neural-network-based filter which is a typical SISO system. The performance parameters, that is, MSE, correlation coefficient, *N/P*, and *t*, are found to be in the expected range of values.

1. Introduction

The bioelectric signals associated with muscle fibers constitute the electromyogram (EMG). The contraction of skeletal muscle results in the generation of action potentials in the individual muscle fibers, a record of which is known as EMG. The activity is similar to that observed in cardiac muscle, but in skeletal muscle, repolarization takes place much more rapidly; the action potential lasting only a few milliseconds. The electrical activity of the underlying muscle mass can be observed by means of surface electrodes on the skin. However, it is usually preferred to record the action potentials from individual motor units for better diagnostic information [1, 2].

In voluntary contraction of skeletal muscle, the muscle potentials range from 25 μV to 5 mV, the duration ranges from 2 to 15 milliseconds, and primary signal frequency ranges from 5 to 2000 Hz. The values vary with the anatomic position of the muscle and the size and location of the electrode. In a relaxed muscle, there are normally no action potentials [3, 4]. Electromyograph is an instrument used for recording the electrical activity of the muscles to determine whether the muscle is contracting or not, displaying on the CRO and loudspeaker the action potentials spontaneously present in a muscle or induced by voluntary contractions as a means of detecting the nature and location of motor unit lesions, or recording the electrical activity evoked in a muscle by stimulation of its nerve. The instrument is useful for making a study of several aspects of neuromuscular function, neuromuscular condition, extent of nerve lesion, reflex responses, and so forth [5, 6].

Signal enhancement in noisy environment is a challenge problem for decades. Noise is added to a signal under measurement almost in an uncontrolled manner. Signal processing systems pick up "unwanted" noise signal along with desired signal. These noise signals result in performance degradation of those systems. Noise classification can be used to reduce the effect of environmental noises on signal processing tasks. Neural networks are proposed as alternative optimization techniques to handle problems in signal processing. Prior a neural network (NN) maps each input feature vector into output vector, it must have first learnt the classes of feature vectors through a process that partitions the set of feature vectors. This is called discrimination (or classification), which involves machines learning [7, 8].

2. Temporal Signal Processing

Temporal signal processing is important for various intelligent behaviors, including hearing, vision, speech, music, and motor control. Because there is an ever-changing environment around, an intelligent system, whether it be a human or a robot, must encode patterns over time and must recognize and generate temporal patterns. Time is embodied in a temporal pattern in two different ways.

(i) *Temporal order*. It refers to the ordering among the components of a sequence. For example, the sequence S-U-B is different from B-U-S. Temporal order may also refer to a syntactic structure, such as subject-verb-object, where each component may be any category of possible symbols.

(ii) *Time duration*. Duration can play a critical role for temporal processing.

Temporal pattern processing is a challenging topic because the information is embedded in time (i.e., inherently dynamic), not simultaneously available. Fundamentally different from static pattern processing, temporal processing requires that a neural network should have a capacity of short-term memory (STM) in order to maintain a component for some time. This is because a temporal pattern extends over a time period. Thus, how to encode STM becomes one of the criteria for classifying NNs for temporal processing [9].

The shared goal of all STM models is to make input history available simultaneously when recognition takes place. With an STM model in place, recognition is not much different from the recognition of static patterns. The architecture for this type of recognition is simply a two-layer network: the input layer that incorporates STM and the sequence recognition layer where each unit encodes an individual sequence. The recognition scheme is essentially template matching, where templates are formed through the following Hebbian learning:

$$Wij(t) = Wij(t-1) + Csi(t)[xj(t) - Wij(t-1)], \quad (1)$$

where Wij is the connection weight from unit xj in the input layer to sequence recognizer si in the recognition layer. Parameter C controls learning rate. Hebbian learning is applied after the presentation of the entire sequence is completed. Thus, the templates formed can be used to recognize specific input sequences. The recognition layer typically includes recurrent connections for selecting a winner by self-organization during training [10].

3. Neural Network Approach

There are numerous real life situations where the exactness of the measurements is required. In biomedical applications, due to complicated situations, the measurements are noisy. NNs can be used to obtain reasonably good accuracy in removal of noise or elegantly filtering out the desired signals. At a high level, the filtering problem is a special class of function approximation problem in which the function values are represented using time series. A time series is a sequence of values measured over time in the discrete or continuous time units. Literature survey revealed that the NNs can also be effectively used for solving the nonlinear multivariable regression problem [11, 12]. Also, there is a wide scope for an exact NN with the performance indices approaching to their ideal values, that is, MSE = 0, and correlation coefficient $r = 1$ [13].

Signal filtering from present observations is a basic signal processing operation by the use of filters. Conventional parametric approaches to this problem involve mathematical modeling of the signal characteristics, which is then used to accomplish the filtering. In a general case, this is relatively a complex task containing many steps for instance model hypothesis, identification and estimation of model parameters and their verification. However, using an NN, the modeling phase can be bypassed, and nonlinear and nonparametric signal filtering can be performed. As the thresholds of all neurons are set to zeros, unknown variables for one step ahead filtering are only the connection weights between the output neurons and the jth neuron in the second layer, which can be trained by available sample set [14, 15].

In the last decade, NNs have given rise to high expectations for model-free statistical estimation from a finite number of samples. The goal of predictive learning is to estimate or learn an unknown functional mapping between the input variables and the output variables, from the training set of known input output samples. The mapping is typically implemented as a computational procedure in software. Once the mapping is obtained from the training data, it can be used for predicting the output value, given only the values of the input variables [16]. In the research work referred, the several techniques for noise removal from biomedical signals like EMG [17, 18], EEG [19, 20], and ECG [21, 22] using signal processing techniques [23, 24] and NNs have been presented. Temporal whitening of individual surface EMG and spatial combination of multiple recording sites have separately been demonstrated to improve the performance of EMG amplitude estimation [25]. A systematic experimental study of the influence of smoothing window length on the signal-to-noise ratio of EMG amplitude estimates is described [26]. Because the relationship between EMG signals and muscle activations remains unpredictable, a new way to determine muscle activations from EMG signals by using a feedforward neural network using four layers is proposed [27]. An algorithm that generates EMG signals consistent with those acquired in a clinical setting is described [28].

3.1. Recurrent Networks. Recurrent networks are the proper NN to be selected when identifying a nonlinear dynamical process. Such networks are attractive with their capabilities to perform highly nonlinear dynamic mapping and their ability to store information for later use. Moreover, they can deal with time-varying input or output through their own natural temporal operation. There are two types of recurrent neural networks: fully recurrent neural networks and partially recurrent neural networks. Many learning algorithms have been developed. Partially recurrent networks

are back-propagation networks with proper feedback links. It allows the network to remember cues from the recent past. In these architectures, the nodes receiving feedback signals are context units. According to the kind of feedback links, two major models of partially recurrent networks are encountered as described below.

3.2. Jordan Network. Jordan network model is realized by adding recurrent links from the network's output to a set of context units C_i of a context layer and from the context units to themselves. Context units copy the activations of output node from the previous time step through the feedback links with unit weights. Their activations are governed by the differential equation

$$C_i'(t) = -\alpha C_i(t) + y_i(t), \tag{2}$$

where the y_i's are the activations of the output nodes, and α is the strength of the self-connections.

Despite the use of the Jordan sequential network to recognize and distinguish different input sequences with sequences of increasing length, this model of network encounters difficulties in discriminating on the basis of the first cues presented.

3.3. Multilayer Perceptron Approach. Jordan described the first MLP architecture with recurrent connections for sequence generation. The input layer has two parts: plan units representing external input and the identity of the sequence, and state units that receive one-to-one projections from the output layer, forming decay trace STM. After a sequence is stored into the network by back propagation training, it can be generated by an external input representing the identity of the sequence. This input activates the first component of the sequence in the output layer. This component feeds back to the input layer and, together with the external input, activates the second component, and so on. A particular component of a sequence is generated by the part of the sequence prior to the component; earlier components have lesser roles due to exponential decay. Elman later modified Jordan's architecture by having the hidden layer connect to a part of the input layer, called the context layer. The context layer simply duplicates the activation of the hidden layer in the previous time step. Elman used this architecture to learn a set of individual sequences satisfying a syntactic description and found that the network exhibits a kind of syntax recognition. This result suggests a way of learning high-level structures, such as natural language grammar.

3.4. Elman Neural Network. Elman neural network (ENN) is a type of partial recurrent neural network, which consists of two-layer back propagation networks with an additional feedback connection from the output of the hidden layer to its input layer. The advantage of this feedback path is that it allows ENN to recognize and generate temporal patterns and spatial patterns. This means that after training, interrelations between the current input and internal states are processed to produce the output and to represent the relevant past information in the internal states. As a result, the ENN has been widely used in various fields from a temporal version of the exclusive-OR function to the discovery of syntactic or semantic categories in natural language data. However, since ENN often uses back propagation (BP) to deal with the various signals, it has proved to be suffering from a suboptimal solution problem. At the same time, for the ENN, it is less able to find the most appropriate weights for hidden neurons and often get into the suboptimal areas because the error gradient is approximated.

In the ENN, after the hidden units are calculated, their values are used to compute the output of the network and all are also stored as "extra inputs" (called context unit) to be used when the network is operated next time. Thus, the recurrent contexts provide a weighted sum of the previous values of the hidden units as input to the hidden units. The activations are copied from hidden layer to context layer on a one-to-one basis, with fixed weight of 1 ($w = 1$). The forward connection weight is trained between hidden units and context units as well as other weights [29]. Both of the Jordan and Elman networks have fixed feedback parameters, and there is no recurrence in the input-output path. These networks can be trained approximately with straight back propagation. Elman's context layer receives input from the hidden layer, while Jordan's context layer receives input from the output as shown in Figure 1.

A typical nonlinear regression problem of removing noise from an EMG signal has been considered in this paper using a Jordan/Elman NN. Jordan and Elman networks extend the multilayer perceptron with context units, which are processing elements (PEs) that remember past activity. Context units provide the network with the ability to extract temporal information from the data. In the Elman network, the activity of the first hidden PEs is copied to the context units, while the Jordan network copies the output of the network. Networks which feed the input and the last hidden layer to the context units are also available. The training data is used to train a Jordan/Elman NN to remove the noise in the EMG signal. This contains 890 data samples in two variables.

4. Performance Measures

4.1. Mean Square Error (MSE). The formula for the mean square error is

$$\text{MSE} = \frac{\sum_{j=0}^{p} \sum_{i=0}^{n} (d_{ij} - y_{ij})^2}{NP}, \tag{3}$$

where P = number of output processing elements, N = number of exemplars in the data set, y_{ij} = network output for exemplar i at processing element j, and d_{ij} = desired output for exemplar i at processing element j.

Learning of an NN is a stochastic process that depends not only on the learning parameters but also on the initial conditions. Thus, if it is required to compare network convergence time or final value of the MSE after a number of iterations, it is necessary to run each network several times with random initial conditions and pick the best.

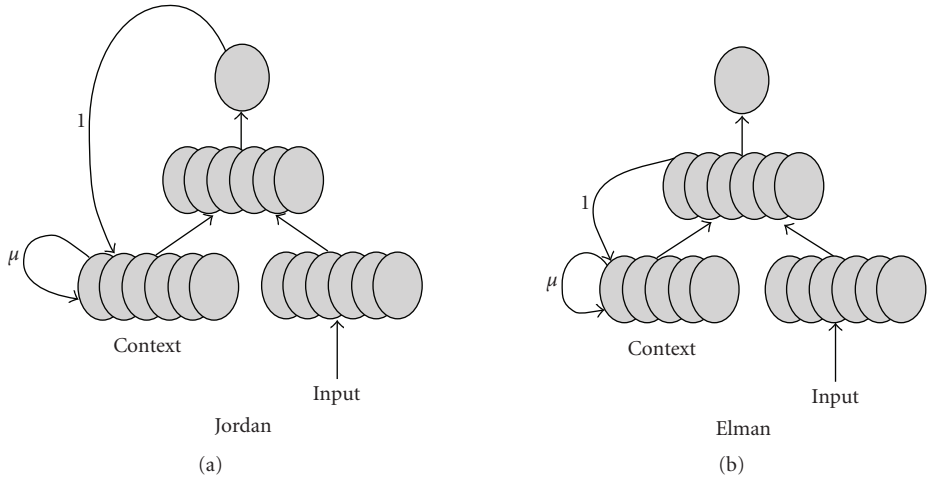

Figure 1: Jordan and Elman network.

4.2. Correlation Coefficient (r). The size of the mean square error (MSE) can be used to determine how well the network output fits the desired output, but it doesn't necessarily reflect whether the two sets of data move in the same direction. For instance, by simply scaling the network output, the MSE can be changed without changing the directionality of the data. The correlation coefficient (r) solves this problem. By definition, the correlation coefficient between a network output x and a desired output d is

$$r = \frac{\sum_i (x_1 - \bar{x})(d_i - \bar{d})/N}{\sqrt{\sum_i (d_i - \bar{d})^2/N}\sqrt{\sum_i (x_1 - \bar{x})^2/N}}. \quad (4)$$

The numerator is the covariance of the two variables, and the denominator is the product of the corresponding standard deviation, and N is the number of exemplars. The correlation coefficient is confined to the range $[-1, 1]$. When $r = 1$, there is a perfect positive linear correlation between x and d, that is, they covary, which means that they vary by the same amount. When $r = -1$, there is a perfectly linear negative correlation between x and d, that is, they vary in opposite ways. When $r = 0$, there is no correlation between x and d, that is, the variables are uncorrelated. Intermediate values reveal partial correlations (e.g., $r = 0.9$) which state that the fit of the linear model to the data is reasonably good [13].

Correlation coefficient (r) tells how much of the variance of d is captured by a linear regression on the independent variable x, and hence, r is a very effective quantifier of the modeling result. It has the greatest advantage with respect to the MSE as it is automatically normalized, while the MSE is not. But r is blind to the differences in means as it is a ratio of variances, that is, as long as the desired data and input covary, r will be small, in spite of the fact that they may be far apart in actual value. Hence, both parameters (r and MSE) are required when testing the results of regression.

4.3. The N/P Ratio. The N/P ratio describes the complexity of an NN and is given by

$$N/P = \frac{\text{Total Training Samples}}{\text{Number of connection weights}}. \quad (5)$$

4.4. Time Elapsed Per Epoch Per Exemplar (t). Time elapsed per epoch per exemplar (t) helps to calculate the speed of a network and is given by

$$t = \frac{\text{Time elapsed for } n \text{ samples}}{(n \text{ samples} \times \text{total training samples})}. \quad (6)$$

5. Database Descriptions

The noisy EMG input signal under consideration has 890 samples. Out of total 890 samples, 450 samples constitute the training set, 180 samples are used for cross-validation (CV), and 210 samples are chosen for testing set. The training set is used to train the NN. During the learning, the weights and biases are updated dynamically using the back propagation algorithm. The validation set is used to determine the performance of the NN on patterns that are not trained during learning. Its major goal is to avoid the over training during the learning phase. The testing set is used to check the overall performance of the network. The input PE and output PE were chosen to be one, as it is a single input (i.e., noisy EMG input) and a single output (i.e., desired or filtered EMG output), SISO system. The NN defined has the other parameters like context unit (time) = 0.8 microsecond, transfer function = TanhAxon, and learning rule = momentum. Termination criterion for experimentation is minimum MSE in both training and cross-validation stages with maximum epochs = 1000, and learning rate is fixed to 0.01.

6. Simulation

The results are obtained on neuro solutions platform and accordingly, simulations are carried out on noisy EMG input

Table 1: For hidden layers = 2, input PE = 1, output PE = 1, transfer function = TanhAxon, learning rule = momentum, maximum epochs = 1000, and threshold MSE = 0.01.

Serial no.	Type of ANN	Hidden layer variation H1 H2	Correlation coefficient (r)	MSE			N/P	t (μ sec)
				Training	Cross validation	Testing		
01	Jordan/Elman network	**09**,06	**0.782 021 085**	0.009 907 153	0.017 699 749	0.013 501 987	7.5	11.3
02	Jordan/Elman network	07,**05**	0.775 341 114	0.009 949 377	0.018 167 161	0.013 811 668	7.5	10.8
03	Jordan/Elman network	08,**05**	0.777 408 297	0.00 996 472	0.017 879 274	0.013 631 649	10	32.6

Table 2: For hidden layers = 3, input PE = 1, output PE = 1, transfer function = TanhAxon, learning rule = momentum, maximum epochs = 1000, and threshold MSE = 0.01.

Serial no.	Type of ANN	Hidden layer variation H1 H2 H3	Correlation coefficient (r)	MSE			N/P	t (μs)
				Training	Cross validation	Testing		
01	Jordan/Elman network	**08,**04,02	0.781 188 725	0.009 985 315	0.020 162 012	0.014 843 202	9.7826	6.4
02	Jordan/Elman network	08,**04**,02	**0.805 945 071**	0.009 983 631	0.021 040 779	0.015 119 372	9.7826	6.5
03	Jordan/Elman network	**08**,05,04	0.753 974 779	0.009 980 005	0.019 238 182	0.014 509 881	6.5217	23.8
04	Jordan/lman network	08,**05,**04	0.793 330 832	0.009 983 942	0.02 0150 538	0.014 729 364	6.5217	18.8

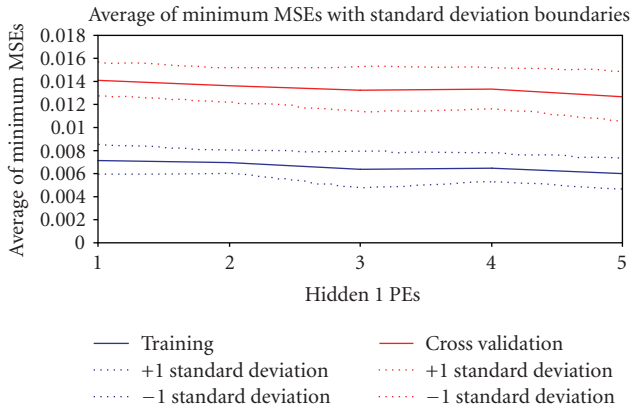

Figure 2

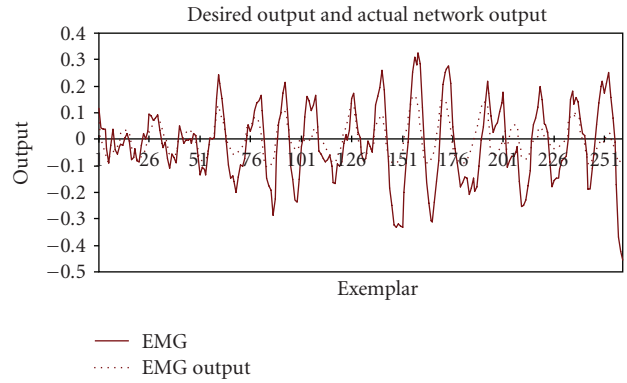

Figure 4

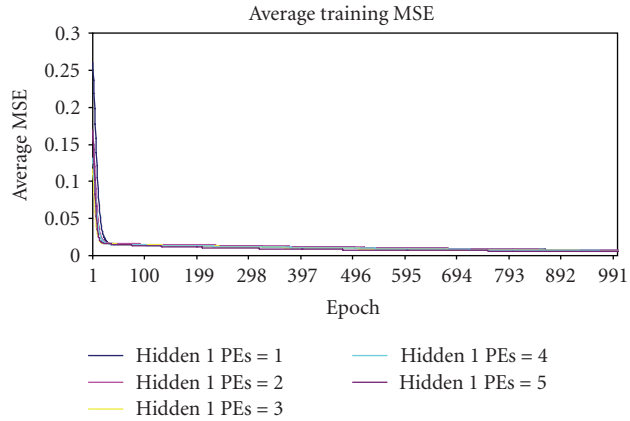

Figure 3

and desired EMG signal. The noisy EMG input was inputted to a Jordan/Elman NN with a number of hidden layers varying from 2 to 4. A Jordan/Elman NN with input, hidden, and output layer with varying parameters like processing elements, transfer function, learning rule, step size, and momentum were tested with maximum epoch value 1000.

After training the Jordan/Elman NN on a noisy input and desired output data values with 890 samples and under different (training, cross-validation, and testing samples swapped) conditions, the expected results were obtained with minimum MSE and maximum correlation coefficient around the estimated values as shown below. The EMG signal under consideration, having a total 890 samples, was divided into various tags, that is, 50% sample for training, 20% for cross validation, and 30% for testing. The hidden layers were

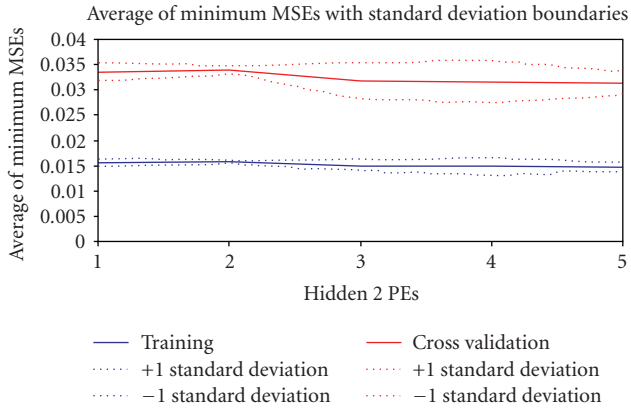

Figure 5

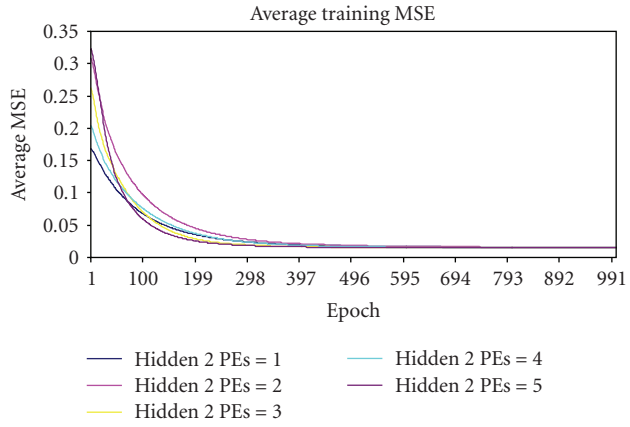

Figure 6

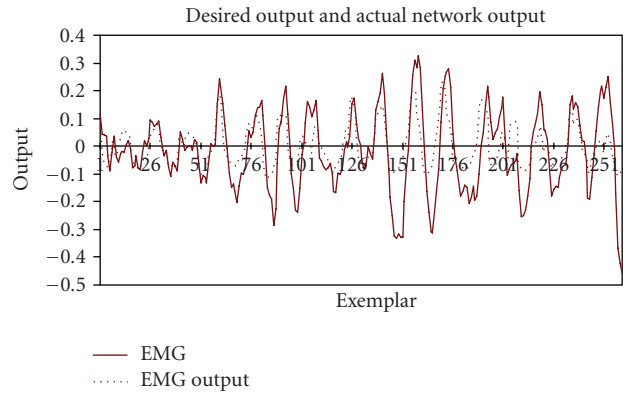

Figure 7

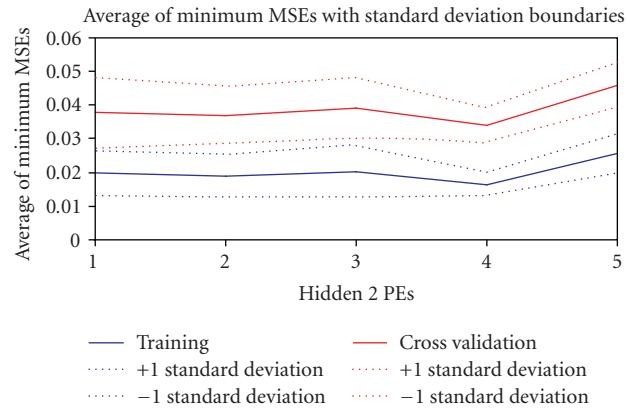

Figure 8

varied from 2 to 4 for experimentation. The other parameters like processing element per hidden layer, transfer function, and learning rule were also varied. The results for optimum parameters for SISO system under consideration are given in following tables.

Simulations Results and Discussion. The Jordan/Elman network was designed with two hidden layers, and processing elements per hidden layer were varied from minimum 1 to the maximum value as indicated in the table separately for hidden layer 1 and then for hidden layer 2, and a few sample results with optimal performance is as depicted in Table 1 with other parameters as described below. In Table 1, N denotes number of exemplars in the training data set, and P denotes total number of connection weights (free parameters) of the specific NN model. Time t denotes time elapsed per epoch per exemplar or instances.

The Jordan/Elman NN was trained for five times and the best performance with respect to MSE of training was observed during the third run at the end of 1000 epochs. Similarly, the best cross-validation performance was noticed during the fourth run at the end of 1000 epochs. Following graphs depict the results for simulation at serial number 1 in Table 1, for $r = 0.782\,021\,085$.

Figure 2 depicts the variation of average of minimum MSE for 5 runs versus the number of PEs in the first hidden layer. It is observed that for four processing elements in the first hidden layer, the MSE on CV attained its minimum value. When PEs are increased beyond 5, the MSE on CV was seen to increase. Therefore, 5 PEs are chosen for first hidden layer.

Figure 3 depicts the variation of average training MSE versus the number of epochs. Five different runs with new random initialization of connection weights of NNs are shown below. It is observed that for each run (training cycle), average MSE decreases as number of epochs increases. It is worthwhile to notice that this trend of decrease in MSE is consistent for all the five runs.

Figure 4 shows the variation of desired output and actual NN output versus the number of exemplars. The covariance between the desired output and actual NN output is indicated by correlation coefficient, $r = 0.782\,021\,085$.

The Jordan/Elman network was designed with three hidden layers, and processing elements per hidden layer were varied from minimum 1 to the maximum value as indicated in the table separately for hidden layer 1 and then for hidden layer 2, and a few sample results with optimal performance are depicted in Table 2 with other parameters as described below.

Table 3: For hidden layers = 4, input PE = 1, output PE = 1, transfer function = TanhAxon, learning rule = momentum, maximum epochs = 1000, and threshold MSE = 0.01.

Serial no.	Type of ANN	Hidden layer variation H1 H2 H3 H4	Correlation coefficient (r)	MSE			N/P	t (μs)
				Training	Cross validation	Testing		
01	Jordan/Elman network	**07**,11,12,05	0.771 344 418	0.006 039 548	0.013 371 967	0.012 071 403	1.5151	25.6
02	Jordan/Elman network	07,**11**,12,05	**0.78 827 949**	0.008 325 698	0.017 142 589	0.013 621 869	1.5151	23.2
03	Jordan/Elman network	**06**,09,09,05	0.785 489 978	0.009 970 153	0.019 572 975	0.014 487 733	2.21 675	35.7

Table 4

Serial no.	Type of ANN	Hidden layer variation H1 H2 H3	Correlation coefficient (r)	MSE		
				Training	Cross validation	Testing
01	Jordan/Elman network	08,04,02	**0.858 319 624**	0.005 840 824	0.020 745 749	0.02 014 514

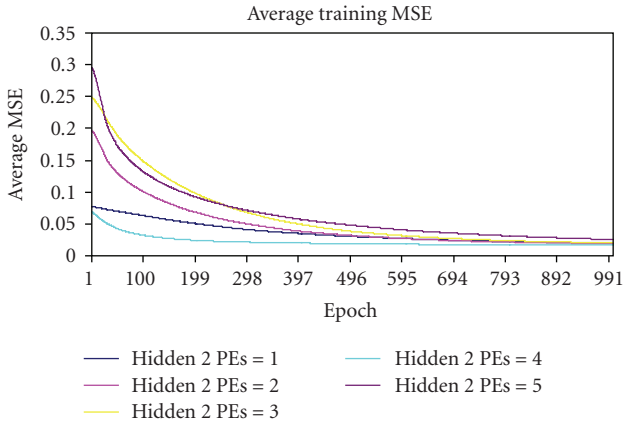

Figure 9

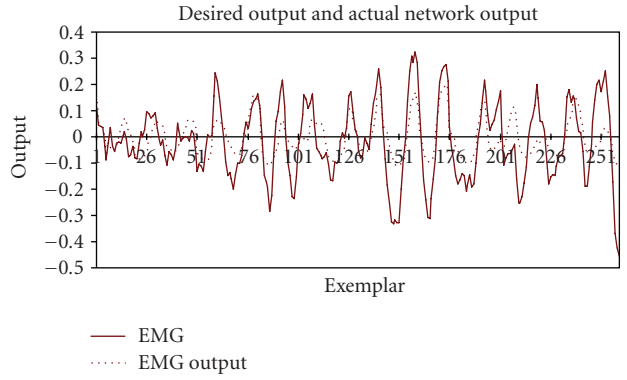

Figure 10

The Jordan/Elman NN was trained for five times, and the best performance with respect to MSE of training was observed during the second run at the end of 1000 epochs. Similarly, the best cross-validation performance was noticed during the second run at the end of 1000 epochs. Following graphs depict the results for simulation at serial number 2 in Table 2, for $r = 0.805\,945\,071$.

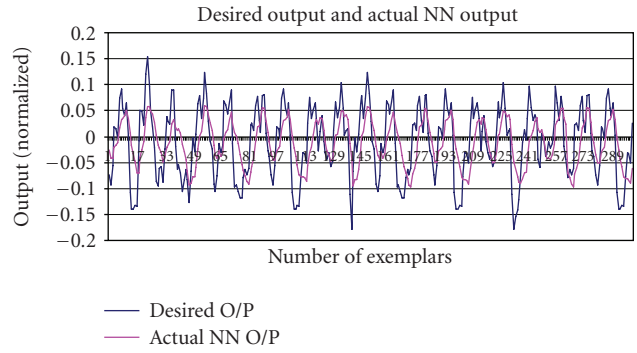

Figure 11

Figure 5 depicts the variation of average of minimum MSE for 5 runs versus the number of PEs in the second hidden layer. It is observed that for three processing elements in the second hidden layer, the MSE on CV attained its minimum value. When PEs are increased beyond 4, the MSE on CV was seen to increase. Therefore, 4 PEs are chosen for second hidden layer.

Figure 6 depicts the variation of average training MSE versus the number of epochs. Five different runs with new random initialization of connection weights of NNs are shown below. It is observed that for each run (training cycle), average MSE decreases as number of epochs increases. It is worthwhile to notice that this trend of decrease in MSE is consistent for all the five runs.

Figure 7 shows the variation of desired output and actual NN output versus the number of exemplars. The covariance between the desired output and actual NN output is indicated by correlation coefficient, $r = 0.805\,945\,071$.

The Jordan/Elman network was designed with four hidden layers, and processing elements per hidden layer were varied from minimum 1 to the maximum value as indicated in the table separately for hidden layer 1 and then for hidden layer 2, and a few sample results with optimal performance are depicted in Table 3 with other parameters as described below.

Table 5

Serial no.	Type of ANN	Hidden layer variation H1 H2 H3	Correlation coefficient (r)	MSE			N/P	t (μs)
				Training	Cross validation	Testing		
01	MLP	08,04,02	0.726 960 433	0.011 620 758	0.019 586 198	0.014 066 105	9.7826	6.1
02	Gen FF	08,04,02	0.738 815 896	0.011 513 017	0.018 595 822	0.01 154 671	9.7826	185.1
03	Mod NN	08,04,02	0.734 250 027	0.012 326 123	0.023 655 146	0.016 096 204	9.7826	37.9
04	RBF	08,04,02	0.710 938 982	0.011 711 991	0.020 447 456	0.014 686 126	9.7826	42.1
05	Time-lag Rec	08,04,02	0.703 300 064	0.009 955 203	0.023 340 290	0.016 268 503	9.7826	1.667
06	Recurrent NN	08,04,02	**0.739 407 946**	0.009 971 926	0.018 246 162	0.013 818 155	9.7826	493.8

The Jordan/Elman NN was trained for five times, and the best performance with respect to MSE of training was observed during the fifth run at the end of 1000 epochs. Similarly, the best cross-validation performance was noticed during the fifth run at the end of 1000 epochs. Following graphs depict the results for simulation at serial number 2 in Table 3, for $r = 0.788\,279\,49$.

Figure 8 depicts the variation of average of minimum MSE for 5 runs versus the number of PEs in the second hidden layer. It is observed that for four processing elements in the second hidden layer, the MSE on CV attained its minimum value. When PEs are increased beyond 4, the MSE on CV was seen to increase. Therefore, 4 PEs are chosen for second hidden layer.

Figure 9 depicts the variation of average training MSE versus the number of epochs. Five different runs with new random initialization of connection weights of NNs are shown below. It is observed that for each run (training cycle), average MSE decreases as number of epochs increases. It is worthwhile to notice that this trend of decrease in MSE is consistent for all the five runs.

Figure 10 shows the variation of desired output and actual NN output versus the number of exemplars. The covariance between the desired output and actual NN output is indicated by correlation coefficient, $r = 0.788\,279\,49$.

The optimum value for correlation coefficient, $r = 0.805\,945\,071$, is observed for a Jordan/Elman NN with three hidden layers as depicted in Table 2 (serial number 2). The NN was tested with leave-N-out mode using increasing MSE criterion over cross validation samples, with a window of as minimum as 30 samples in testing phase traversing over the complete range of 890 samples, with remaining samples (860) for training phase. Table 4 shows the values obtained during experimentation.

The graph in Figure 11 depicts the variation of desired output and actual network output as a function of number of exemplars. It is clearly seen that both outputs covary as it is evident from the value of $r = 0.858\,319\,624$. It is seen that the output of the Jordan/Elman NN closely follows the desired EMG signal output. Only 300 instances out of 3000 are shown in this graph as a scattered plot.

Simulations were carried out for comparison of other neural network's performance like multilayer perceptron NN (MLP), generalized feedforward NN, modular NN, RBF NN, time-lag recurrent network, and a recurrent NN with other parameters similar to Jordan/Elman NN, and the following results were obtained with maximum correlation coefficient (r) value as indicated in Table 5, that is, $r = \mathbf{0.739\,407\,946}$. Table 5 demonstrates the performance of six different NN configurations with respect to r, MSE, N/P, and t.

7. Conclusion

EMG signal is a very important biomedical signal associated with muscle activity, giving useful information about nerve system in order to detect abnormal muscle electrical activities that occur in many diseases and conditions like muscular dystrophy, inflammation of muscles, pinched nerves, peripheral nerve damages, amyotrophic lateral sclerosis, disc herniation, myasthenia gravis, and others. The detection and measurement of low-frequency and lower-magnitude EMG signals is noise-prone. Removal of noise from a typical EMG signal using Jordan/Elman NN has been studied in this paper. It is demonstrated that Jordan/Elman NN elegantly reduces the noise from the EMG signal. The difference between the noisy EMG signal and the desired EMG signal is computed as a performance measure (MSE) and is found to be in the expected range approaching to 0.01. The minimum MSE criterion is found satisfactory (0.0099–0.01) in trained Jordan/Elman NN and found to perform better during testing phase (0.01) as it is evident from all similar graphs as in Figure 2. The correlation coefficient (r) is also found to be in the desired range (0.805 945 071 at serial number 2 Table 2), so that the network output and the desired output covary, that is, varying by the same amount as depicted in Figure 7. This covariance is also observed in leave-N-out mode with improved performance (i.e., $r = 0.858\,319\,624$ as obvious from Figure 11 and Table 4). Moderately smaller values of N/P also show that the Jordan/Elman NN so designed is simpler to design and is capable of generalization. In addition, it is faster as it is evident from smaller values of t.

References

[1] J. D. Bronzino, Ed., *The Biomedical Engineering Handbook*, CRC Press, Boca Raton, Fla, USA, 1995.

[2] L. C. Brush, et al., *The Guide to Biomedical Standards*, Quest, Bera, Calif, USA, 20th edition, 1995.

[3] B. J. Cohen, *Medical Terminology: An Illustrated Guide*, Lippincott Williams & Wilkins, Philadelphia, Pa, USA, 2nd edition, 1995.

[4] L. Cromwell, F. J. Wiebell, and E. A. Pfeiffer, *Biomedical Instrumentation and Measurements*, Prentice-Hall, Englewood Cliffs, NJ, USA, 2nd edition, 1980.

[5] J. G. Webster, Ed., *Medical Instrumentation: Application and Design*, John Wiley & Sons, New York, NY, USA, 3rd edition, 2001.

[6] W. J. Tompkins, Ed., *Biomedical Digital Signal Processing*, Prentice-Hall, Englewood Cliffs, NJ, USA, 1999.

[7] L. Cromwell, F. J. Wiebell, and E. A. Pfeiffer, *Biomedical Instrumentation and Measurements*, Prentice-Hall, Englewood Cliffs, NJ, USA, 2004.

[8] E. C. Eugène, "Discriminating Noisy Sentences with Partially Recurrent Neural Networks," June 2008, http://www.cari-info.org/actes2006/59.pdf.

[9] D. Wang, "Temporal pattern processing," in *The Handbook of Brain Theory and Neural Networks*, pp. 1163–1167, MIT Press, Cambridge, Mass, USA, 2nd edition, 2003.

[10] D. L. Wang and B. Yuwono, "Incremental learning of complex temporal patterns," *IEEE Transactions on Neural Networks*, vol. 7, no. 6, pp. 1465–1481, 1996.

[11] Q. Xue, Y. H. Hu, and W. J. Tompkins, "Neural-network-based adaptive matched filtering for QRS detection," *IEEE Transactions on Biomedical Engineering*, vol. 39, no. 4, pp. 317–329, 1992.

[12] R. D. de Veaux, J. Schumi, J. Schweinsberg, and L. H. Ungar, "Prediction intervals for neural networks via nonlinear regression," *Technometrics*, vol. 40, no. 4, pp. 273–282, 1998.

[13] J. C. Principe, N. R. Euliano, and W. C. Lefebvre, *Neural and Adaptive Systems*, John Wiley & Sons, New York, NY, USA, 2000.

[14] S. Haykin, *Adaptive Filter Theory*, Prentice-Hall, Englewood Cliffs, NJ, USA, 1986.

[15] B. Widrow, J. R. Glover Jr., J. M. McCool, et al., "Adaptive noise cancelling: principles and applications," *Proceedings of the IEEE*, vol. 63, no. 12, pp. 1692–1716, 1975.

[16] R. S. Khandpur, *Handbook of Biomedical Instrumentation*, Tata McGraw-Hill, New Delhi, India, 2001.

[17] A. Zeghbib, F. Palis, N. Shoylev, V. Mladenov, and N. Mastorakis, "Sampling frequency and pass-band frequency effects on neuromuscular signals (EMG) recognition," in *Proceedings of the 6th WSEAS International Conference on Signal Processing, Robotics and Automation (ISPRA '07)*, pp. 107–114, Corfu Island, Greece, February 2007.

[18] U. Gündoğdu, A. Sayin, A. Akan, Y. Arslan, E. K. Orhan, and M. B. Baslo, "Investigation of muscle fatigue using temporal and spectral moments," in *Proceedings of the 5th WSEAS International Conference on Signal Processing (SIP '06)*, pp. 10–14, Istanbul, Turkey, May 2006.

[19] M. Cabrerizo, M. Adjouadi, M. Ayala, I. Yaylali, A. Barreto, and N. Rishe, "EEG analysis using neural networks for seizure detection," in *Proceedings of the 6th WSEAS International Conference on Signal Processing, Robotics and Automation (ISPRA '07)*, pp. 121–126, Corfu Island, Greece, February 2007.

[20] D. Coufal, "Classification of EEG signals by radial neuro-fuzzy system," in *Proceedings of the 4th WSEAS International Conference on Computational Intelligence, Man-Machine Systems and Cybernetics (CIMMACS '05)*, pp. 224–232, Miami, Fla, USA, November 2005.

[21] M. S. Chavan, R. Agarwala, and M. D. Uplane, "Use of Kaiser window for ECG processing," in *Proceedings of the 5th WSEAS International Conference on Signal Processing, Robotics and Automation (ISPRA '06)*, pp. 285–289, Madrid, Spain, February 2006.

[22] H. Husain and L. L. Fatt, "Effecient ECG signal classification using sparsely connected radial basis function neural network," in *Proceedings of the 6th WSEAS International Conference on Circuits, Systems, Electronics, Control & Signal Processing (CSECS '07)*, pp. 412–416, Cairo, Egypt, December 2007.

[23] M. Arezki, A. Benallal, P. Meyrueis, A. Guessoum, and D. Berkani, "Analysis of fast recursive least squares algorithms for adaptive filtering," in *Proceedings of the 11th WSEAS International Conference on Systems*, vol. 2, pp. 473–480, Crete Island, Greece, July 2007.

[24] C. Li and K.-H. Cheng, "Adaptive noise cancellation withcomputational-intelligence-based approach," in *Proceedings of the 5th WSEAS International Conference on Signal Processing (SIP '06)*, pp. 93–98, Istanbul, Turkey, May 2006.

[25] E. A. Clancy and N. Hogan, "Multiple site electromyograph amplitude estimation," *IEEE Transactions on Biomedical Engineering*, vol. 42, no. 2, pp. 203–211, 1995.

[26] Y. St-Amant, D. Rancourt, and E. A. Clancy, "Influence of smoothing window length on electromyogram amplitude estimates," *IEEE Transactions on Biomedical Engineering*, vol. 45, no. 6, pp. 795–800, 1998.

[27] L. Wang and T. S. Buchanan, "Prediction of joint moments using a neural network model of muscle activations from EMG signals," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 10, no. 1, pp. 30–37, 2002.

[28] A. Hamilton-Wright and D. W. Stashuk, "Physiologically based simulation of clinical EMG signals," *IEEE Transactions on Biomedical Engineering*, vol. 52, no. 2, pp. 171–183, 2005.

[29] Z. Zhang, Z. Tang, and C. Vairappan, "A novel learning method for Elman neural network using local search," *Neural Information Processing—Letters and Reviews*, vol. 11, no. 8, pp. 181–188, 2007.