VOL. 4, NO. 4, JUNE 2009 ISSN 1819-6608
ARPN Journal of Engineering and Applied Sciences


www.arpnjournals.com

ESTIMATING K, C VARIABLE PARAMETERS OF A WEIGHING SYSTEM USING RBF NEURAL NETWORKS

H.B. Bahar[1] and B. Sokouti[2]
[1]Faculty of Electrical and Computer Engineering, University of Tabriz, Tabriz, Iran
[2]Faculty of Engineering, Islamic Azad University-Tabriz Branch, Tabriz, Iran
E-Mail: b.sokouti@gmail.com

ABSTRACT

Neural networks are often used as a powerful discriminating estimator for tasks in system identification. This paper describes a neural-network-based method relies on the Radial Basis Function Network (RBF network), for estimating the variable damping factor C (n) and spring constant K (n) of a weighting platform. Firstly, the RBF network learns key properties of the step response of the weighting platform and then predicts the damping factor C (n) and spring constant K (n) of other systems with different step responses before the platform settled to the steady state. In the simulation and the experimental results, with the related applied masses, the correlation rates between the actual C(n), K(n) and estimated C(n) and K(n) are presented that shows the success of this method.

Keywords: nonlinear system, parameter estimation, artificial neural network, radial basis function (RBF), step response.

INTRODUCTION

The application of artificial neural networks in the control systems is vast and most of the theories have bought up to the practical environment [1, 2]. Radial Basis Functions (RBF) as a variant of artificial neural network in late 80's are known as good functional estimation [3]. It has also proved that any continues nonlinear system can be modeled up to a certain precision by a set of Radial Basis Functions (RBF) [4, 5, 6]. System Identification and neural network techniques involve simpler representation than the finite element method and when created, amount for errors [7] and offer a probabilistic framework to the representation [8]. For instance the relationship between the input and output is learned by lessening the least-square distance between the experiments and the neural network behavior. Nowadays Radial Basis Neural Networks are employed for these advantages: computational simplicity, supported by well-developed mathematical theory, and robust generalization, powerful enough for real-time real-life tasks [9, 10]. According to the recent results, RBF neural networks are effective for identifying a vast category of complex nonlinear systems when we don't have enough information about those systems [11]. This paper describes how neural network based on RBF is employed to identify the K(n), C(n) variable parameters of a nonlinear dynamic system (weighing platform model).This method can be used for finite data and also can be trained on experimental data, which intrinsically include the effects of any other dynamic modes of vibration [12 …17].

MATERIALS AND METHODS

Radial Basis Function (RBF) networks were originally proposed for the exact interpolation problem by Rosenblatt [18, 19] and used for discrimination by Broomhead and Lowe [20], and Moody and Darken [21]. Compared to the MLP model, the RBF network usually has much faster training speed, while maintaining the nonlinear classification ability. Therefore, it has gained popularity since it was originally proposed. The basic RBF network provides a nonlinear transformation of a pattern $x \in R^d \rightarrow R^c$ according to:

$$f_j(x) = b_j + \sum_{i=1}^{m} w_{ji}\varphi\left(\frac{|x-\mu_i|}{h}\right) \quad (1)$$

Where m is the number of basis functions, w_{ji} is a weight, b_j is a bias and $\mu_i \in R^d$ is called the center vector, and $h \in$ R is called the kernel width (or smoothing parameter). Formula (1) almost has the same mathematical form as the MLP model. The major difference is the logistic activation function is replaced by the radial basis function, whose value is usually the largest at its center, μ_i and decreases as x moves away from the center (One example is the Gaussian function $\varphi(x) = e^{\left(-\left(\frac{|x-\mu_i|}{h}\right)^2\right)}$.

Therefore, the RBF network may be simply viewed as putting a small "bump" at each center, and adding the bumps together to form the classification boundary.

Gaussian, multisquared, inverse multisquared, pseudo cubic and thin-plate-spline functions are examples of radial functions. Gaussian functions are normally used in RBFs because of its close connection with statistics. Some graphical examples of radial functions of Table-1 are presented in Figure-1.

Table-1. Commonly used RBFs.

Name	**Mathematical form of** φ (z), **z** = |x − μ| /**h**
Gaussian	exp (−z^2)
Exponential	exp (−z)
Quadratic	$\sqrt{z2 + az + ß}$
Inverse quadratic	$\frac{1}{\sqrt{(a^2 + z^2)}}$
Thin-plate spline	z^2 log (z)

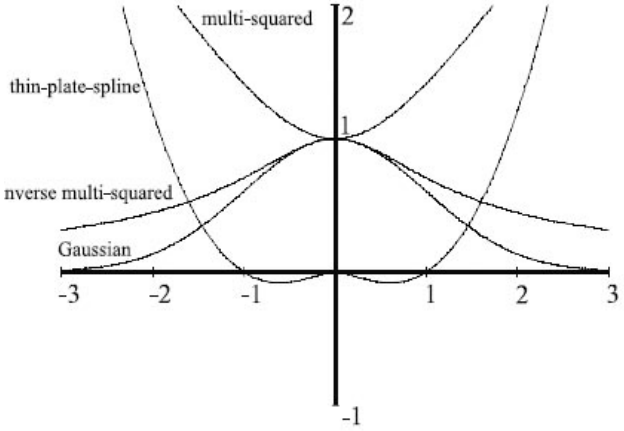

Figure-1. Format of some radial functions.

The parameters that define a radial function are, therefore, the radius and center. The center defines the spatial position of the function and the radius is a measure of how the function spreads around its center. For Gaussian RBFs, the center is associated with the density function mean and the radius with its variance. The two most popular forms are the Gaussian, $\varphi(z) = e^{-z^2}$, and the thin-plate spline, $\varphi(z) = z^2 \log(z)$. The Gaussian is more suitable for discrimination and density estimation, while the thin-plate spline is more suitable for curve fitting [22].

Now RBF can be taken in to consideration for system identification as the input x, include the sampling data from the related plant or controller, and $y(k-1), y(k-2),...y(k-N+1)$ in the time instant $k-1, k-2,...k-N+1$ and $x(k-1), x(k-2),...x(k-N+1)$ represents the system input at the time instant $k-1, k-2,...k-N+1$, from the network perspective the input vector x can be formed from :

$$x = f[y(k-1), y(k-2),...y(k-N+1) : x(k-1), x(k-2),...x(k-N+1)] \quad (2)$$

By appropriate selection of basis functions, center positions and weightings the network output y can be obtained to model the output system. For multiple output systems, several RBFs can be employed and combined to form a network as shown in Figure-2.

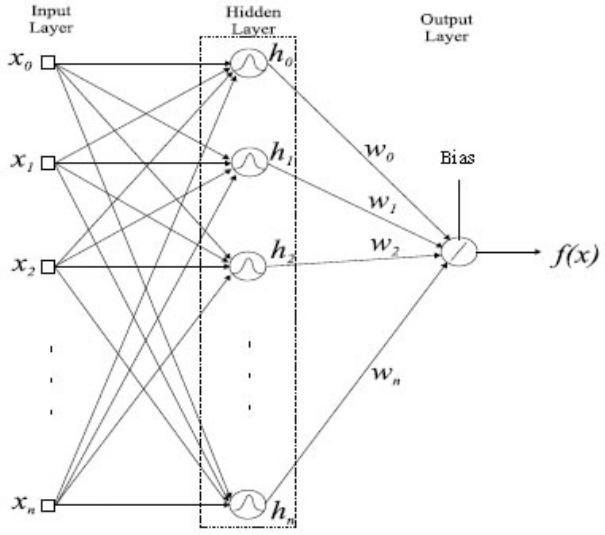

Figure-2. Radial basis function networks architecture.

There are various learning algorithms for the RBF networks [23, 24]. The basic algorithm employs a two-step learning strategy (hybrid learning): estimation of kernel positions and kernel widths using some unsupervised clustering algorithm, followed by a supervised Least Mean Square (LMS) [25] type of algorithm to determine the connection weights to the output layer. Since the output units are linear, a noniterative algorithm can be used. After this initial solution is obtained, a supervised gradient-based algorithm can be used to refine the network parameters. The learning algorithm builds an RBF network by finding out its parameter setting (it is unable to find out the number of hidden units of the RBF network). Therefore, the number of hidden units will be determined by trial and error.

In the supervised learning stage, the RBF network with fixed centers and widths can be interpreted as a case of multivariate linear regression on the training set:

$$y = Zw + e \quad (3)$$

Where $y = [y_1, y_2,..., y_p]^T$ is the desired output, Z is the *design matrix*, which is a matrix with the j th column $[z_j(x_1), z_j(x_2),..., z_j(x_p)]^T$, $w = [w_1, w_2,..., w_m]^T$ is the output layer weight vector and e is the error. The vector w is determined minimizing the sum of squared errors:

Find w that minimizes $SSE = e^T e$ (4)

A simple method to find the solution of this linear least squares problem may be obtained solving the well-known linear system (called normal equations):

$$(\overline{Z}^T Z)w = Z^T y \quad (5)$$

An ideal weighing platform can be modeled by a mass-spring-damping structure shown in Figure-3. It has a typical under damped ideal step response as showed in Figure-4.

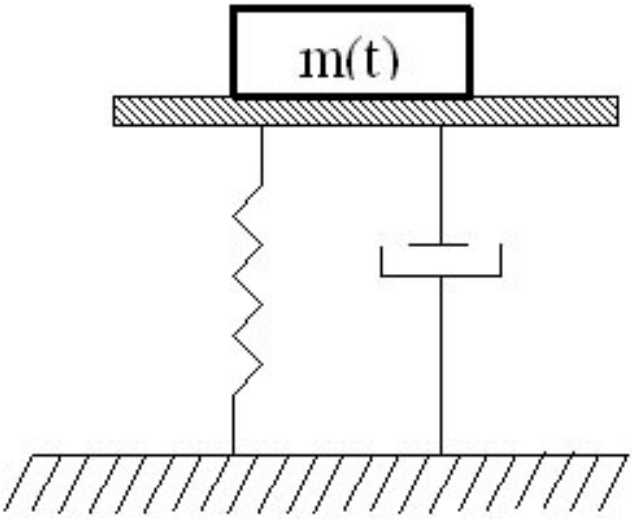

Figure-3. Weighing platform model with mass-spring-damping structure.

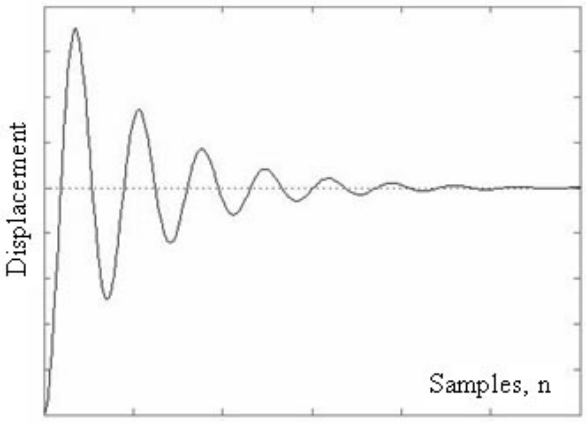

Figure-4. A typical under damped ideal step response.

It is governed by the solution of the following second order differential equation:

$$(m(t) + m_p)y''(t) + C(n)y'(t) + K(n)y(t) = gm(t) \quad (6)$$

Where $y(t)$, is the deflection signal obtained from the strain gauge on the weighing machine; $m(t)$ and mp are applied mass and the platform mass respectively; C(n) is the damping factor; K(n) is the spring constant and g is the gravitational constant.

For a general applied mass function $m(t)$, this is a nonlinear differential equation. However, for commonly faced situation $m(t)$ is a step function, which is assumed here. In this case the differential equation (6) is linear, for which the explicit solution is modeled by a constant term plus a transient term which can be under damped (u), critically damped (c), or over damped (o). Thus:

$$y(t) = q_0 + \{F_u(q_u, t), F_c(q_c, t) or F_o(q_o, t)\} \quad (7)$$

And the transient terms for under damped, critically damped and over damped cases are:

$$F_u(q_u, t) = e^{-q_{u1}t} q_{u2} \sin(q_{u3}t + q_{u4})$$

,

$$F_c(q_c, t) = e^{-q_{C1}t}(q_{C2} + q_{C3}t)$$

And

$$F_o(q_o, t) = e^{-q_{o1}t} q_{o2} + e^{-q_{o3}t} q_{o4}$$

Respectively, where the various q parameters are related to the initial platform displacement b_0, initial velocity b_1, the constant platform parameters C (n), K (n), mp and the applied mass $m(t)$ by the expressions given in the appendix.

These expressions have been used to generate data in the simulation study described in followings section.

Sampled data signals are assumed, for which, $t - nT$, where T is the sample interval. Thus $y(t)$ is written as $y(n)$.

RESULTS AND DISCUSSIONS

Before any simulation was performed, it is very important to initialize the data and the RBF Network with two outputs C(n) and K(n), represented in Figure 5 is used for simulation purposes.

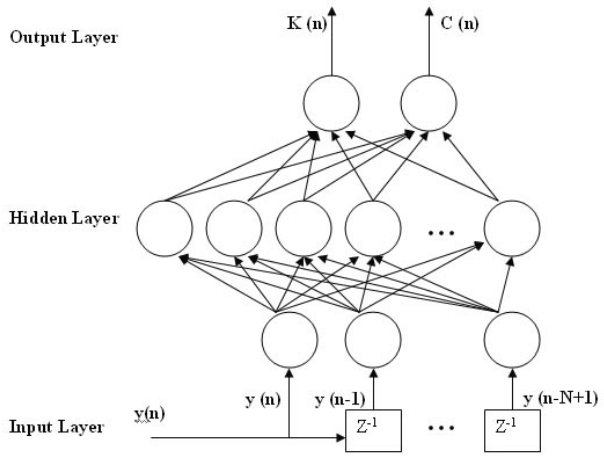

Figure-5. RBF network for platform parameters (K (n), C (n)) identification.

The proposed RBF network model shown in Figure-5 is the owner of the following characteristics:

- Number of input samples:

$$y(k-1), y(k-2),...y(k-N+1); N = 100$$

- Number of output parameters: C(n), K(n)
- Number of layers is 3: input layer, hidden layer and
- Output layer
- Total number of neurons used for this network is 202, where 100 for input neurons, 100 for our hidden neurons and 2 for the output neurons.

The RBF trained using the proposed algorithms based on moving the least squares algorithm was used to model the non-linear system. In this simulation, mass-spring-damping system (weighing platform system) was selected as the non-linear function of RBF network and the RBF centers were initialized to the first few samples of the input-output data. During calculating the mean squared error (MSE), the noise model was excluded from the model since the noise model will normally cause the MSE to become unstable in the early stage of training.

A data set of input samples $(y(k-1), y(k-2),...y(k-99))$ was taken from the MATLAB programming language to simulate the equation 7 in which only the under damped state is considered. The weighing platform parameters in all simulations are $m(t) = 5kg$, $m_p = 0kg$, $g = 10\,m/s^2$, sampling interval $t_s = 0.00004s$, initial platform displacement $b_0 = 0$, initial velocity $b_1 = 0\,m/s$. 100 data patterns were used to train the network; To be sure of the trained network performance, the recalling process includes two stages: the first stage is tested with the seen patterns and the next stage is tested with the unseen patterns to the network and are seen to the user, after these two stage has been completed we can be sure that the network is reliable upon the unseen patterns to both the network and the user. The 10 seen data patterns were used to test the fitted network model and the network model predicts two parameters (K (n), C (n)) reasonably over the training data sets. Since the model predicts reasonably and has good correlation tests [26] according to the Figure 6 and Figure 7, This is also done by the 10 unseen patterns shown in Figure 8, the network predicts two parameters (K(n),C(n)) reasonably over the recalling data sets Since the model predicts reasonably and has good correlation tests according to the Tables 2a and 2b and has a good correlation of 0.99 and a negligible error between desired and actual C(n) and K(n), the model can be considered as an adequate representation of the identified system.

Table-2a. Results of simulation of typical estimated 10 unseen platform parameters K (n), C (n).

Actual K(n),**N/m**	**Estimated K**(n),**N/m**	**Actual C**(n),**Kg/s**	**Estimated C**(n),**Kg/s**
5090	5089.4	11.9	11.897
5190	5189.9	12.9	12.899
5290	5289.3	13.9	13.896
5390	5387.7	14.9	14.889
5490	5489.1	15.9	15.9
5590	5593	16.9	16.909
5690	5693.9	17.9	17.91
5790	5791.9	18.9	18.907
5890	5892.2	19.9	19.916
5990	5985.2	20.9	20.87
Corr = 0.999963		Corr = 0.999991	

Table-2b. Results of simulation of typical estimated 10 unseen platform parameters K (n), C (n).

Absolute Error K(n)	**Absolute Error C**(n)
0.006	0.00003
0.001	0.00001
0.007	0.00004
0.023	0.00001
0.009	0.00000
0.03	0.00009
0.039	0.0001
0.019	0.00007
0.022	0.00016
0.048	0.0003
Rms=0.0257	Rms=1.2e-4

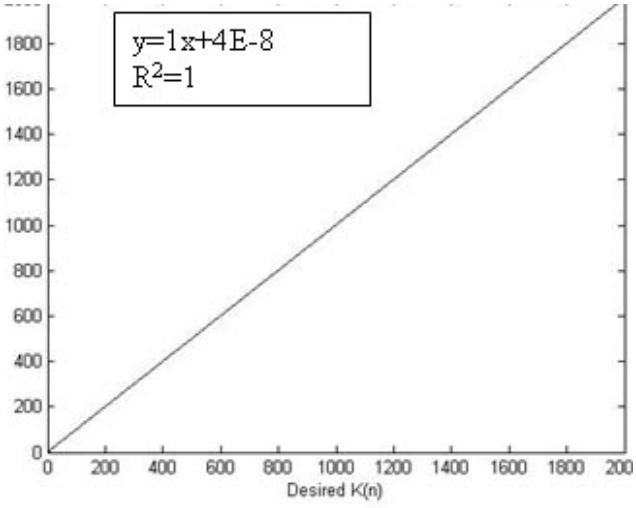

Figure-6. The simulation performance of trained RBF network indicates a linear relationship between the actual K(n) and estimated output K(n).

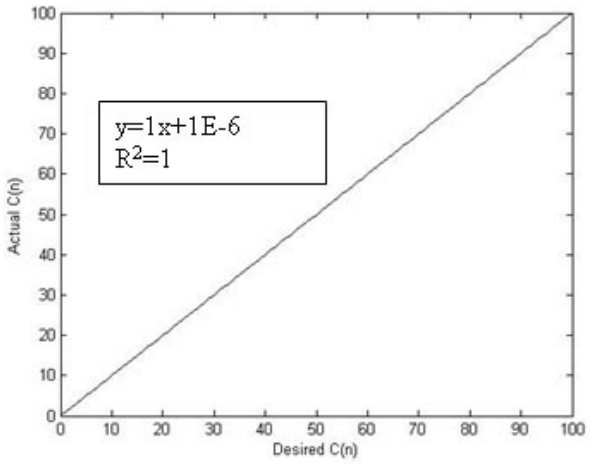

Figure-7. The simulation performance of trained RBF network indicates a linear relationship between the actual C(n) and estimated outputs C(n).

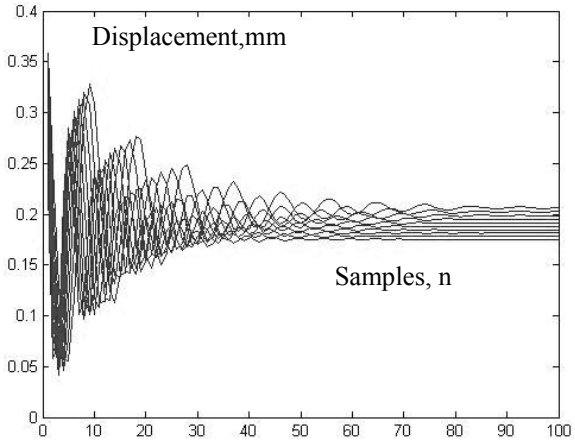

Figure-8. Time series step response with 10 various platform parameters, C(n), K(n).

According to the last two stages of the simulation results (1.training with seen data sets,2. recalling with unseen data set (unseen to the trained network but seen to the user); since the two stage were passed successfully, we would be confident of the results obtained by the trained network for the fully unseen patterns(unseen for both user and the network) but here the training and recalling stages will be done with the experimental data set that are surely not free noise (included noise of about %2) which were obtained from the industrial weighing platform. The weighing platform has the dimensions of 55.50 cm. 50.50 cm and 16.50 cm for length, width and height respectively with nominal full scale taken to be about 100kg. Any dynamic systems with fixed parameters K (n), C (n) can be easily written in the Laplace domain as follows:

$$H(s) = \frac{g}{x^2 + \frac{C(n)}{m(t)}s + \frac{K(n)}{m(t)}} \quad (8)$$

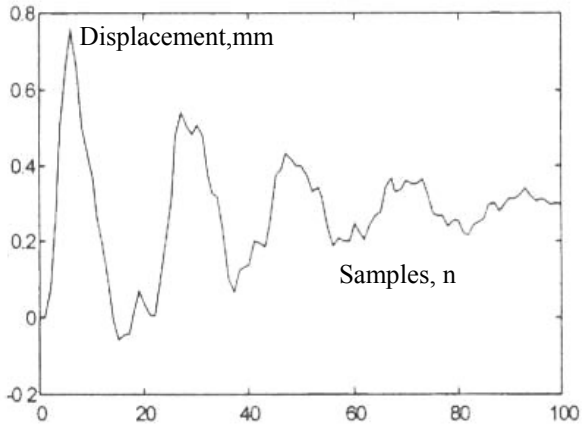

Figure-9. A typical experimental time series step response.

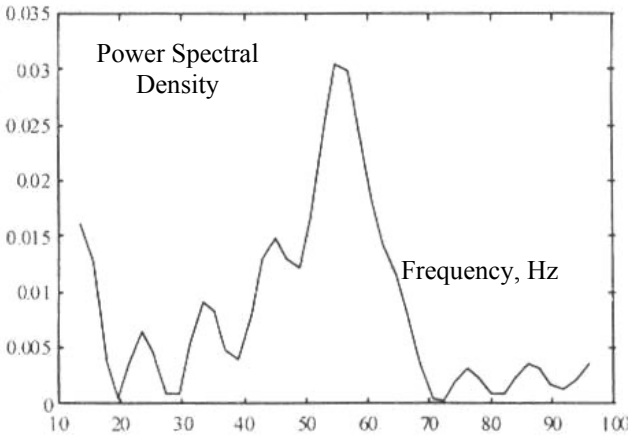

Figure-10. Power spectral density of response shown in Figure-9.

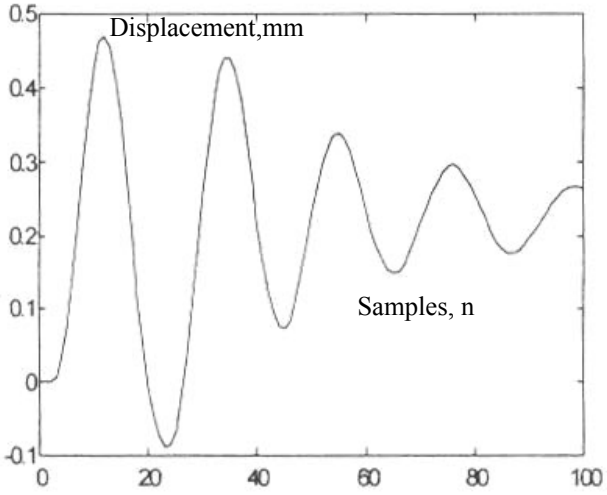

Figure-11. A typical filtered time series step response of Figure-9.

As the weighing system or any non-linear system has fixed parameters C(n) and K(n), for each sequence of applied mass, the coefficients of the Laplacian transfer function $\frac{C(n)}{m(t)}, \frac{K(n)}{m(t)}$ will be variable in equation 8; this means for any sequence of applied mass the coefficients will be estimated. A typical experimental time series step response obtained from the weighing platform system is illustrated in Figure-9 and Power spectral density of the typical response has shown in Figure-10 which represents that the obtained data set from the system has a noise at the frequency of 55 Hz. The source of noise may refer to random and unpredictable electrical signals naturally produced by the internal and external factors of the system. This kind of noise can not be eliminated from entering into the sampling process and filtering must be considered after the samples were obtained to reduce the noise contamination, but some of the noises can't be eliminated obviously; all noise frequency components that fall outside the weighing system respond band should be rejected by a filter such as a low-pass second order Chebyshev filter with BW=50Hz.. After applying the filter to the input samples the energy spectral density of the step response indicates the noise reduction shown in Figure-11 in comparison with the Figure 9 and the filtered step response is shown in Figure 10. Now the RBF network is trained using the new filtered experimental data samples with their corresponding $\frac{C(n)}{m(t)}, \frac{K(n)}{m(t)}$ parameters. For any sequence of applied mass, the first 100 samples in transient region were taken immediately within sampling interval of 0.004 s, so 0.4 s is required for parameters to be estimated.

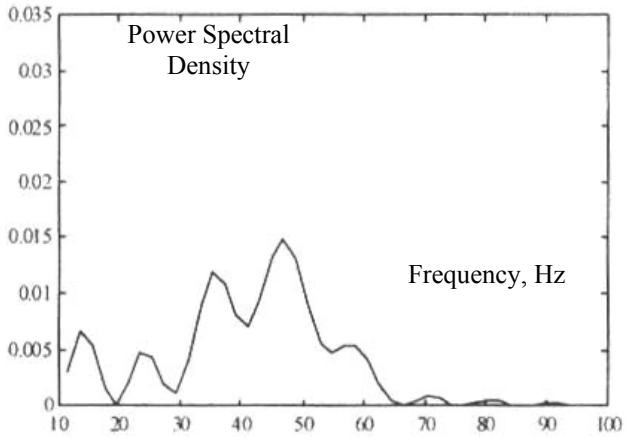

Figure-12. Power spectral density of response shown in Figure-11.

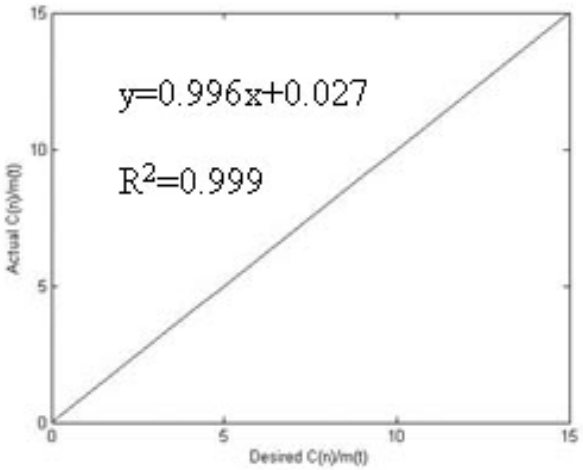

Figure-13. The performance of experimental RBF network for actual and desired parameter C (n)/m (t).

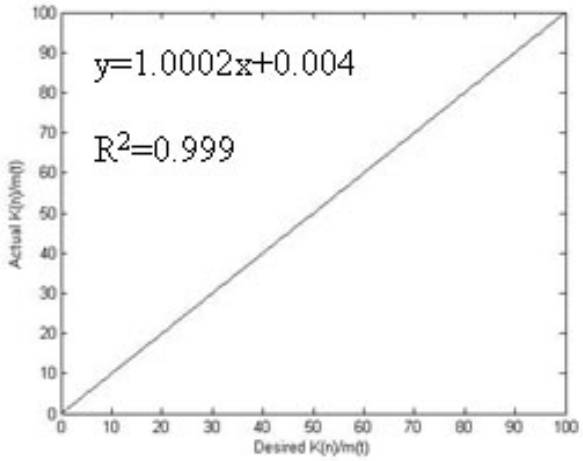

Figure-14. The performance of experimental RBF network for actual and desired parameter K (n)/m (t).

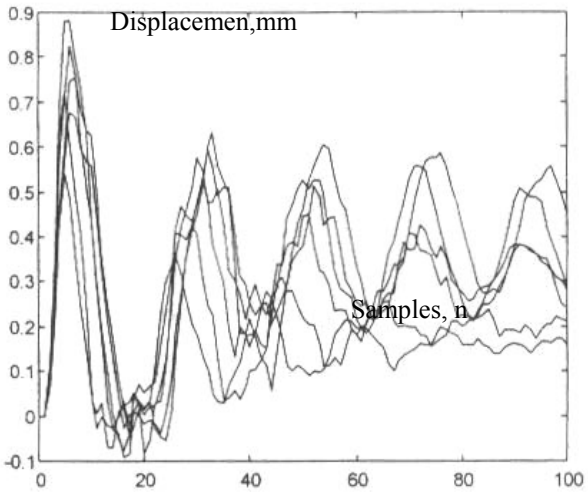

Figure-15. unfiltered step response of typical unseen time series experimental data with different real platform parameter.

For plotting the experiment performance of the trained RBF Network for $\frac{C(n)}{m(t)}, \frac{K(n)}{m(t)}$ the sequence of applied masses is taken from 10-50 Kg in step of 1 Kg as shown in Figures 13 and 14; in these Figures the linear relationship with slope of 1.0002 and 0.996 is assigned for $\frac{C(n)}{m(t)}, \frac{K(n)}{m(t)}$ respectively and the correlation rate (the square root of R) for both $\frac{C(n)}{m(t)}, \frac{K(n)}{m(t)}$ is $R^2 = 0.999$ that show a good estimation since the correlation rate is almost equal to +1. Noticing that for the ANN recalling stage the noise free data patterns were applied for input samples and the results obtained from the Figures 13 and 14 illustrates that the RBF network can model a non-linear system by mapping the input data samples to the corresponding variable parameter K(n) and C(n). The typical unseen patterns that are not used for training the RBF Network shown in Figures 15 and 16 (unfiltered and filtered experimental patterns) for testing the ability of the RBF Network are used at the recalling stage. In table 3 the typical applied masses ranging from 15 to 45 kg in step of 5 kg is applied and the results of the estimated and desired $\frac{C(n)}{m(t)}, \frac{K(n)}{m(t)}$ and the negligible errors between these values are shown clearly.

Table-3. Experimental results of typical desired and estimated $\frac{C(n)}{m(t)}, \frac{K(n)}{m(t)}$.

Applied	**Desired**	**Estimated**	**Desired**	**Estimated**	**Absolute error**	**Absolute error**
Mass (Kg)	$\frac{C(n)}{m(t)}$	$\frac{C(n)}{m(t)}$	$\frac{K(n)}{m(t)}$	$\frac{K(n)}{m(t)}$	$\frac{C(n)}{m(t)}$	$\frac{K(n)}{m(t)}$
15	3.3300	3.2532	66.6600	66.8533	0.0768	0.1933
20	2.5000	2.6532	50.0000	49.9525	0.1532	0.0475
30	1.6600	1.5532	33.3300	33.6366	0.1068	0.3066
35	1.4300	1.3405	28.5700	28.7800	0.0895	0.2100
40	1.2500	1.3946	24.3870	24.8290	0.1446	0.442
45	1.1100	1.0775	22.2200	22.2183	0.0325	0.0017

As the ANN is learning the characteristic of the defined non-linear system, any initial displacement will not have effect on the output response and the accuracy of the output depends on the training process and the sample data patterns we have used for training.

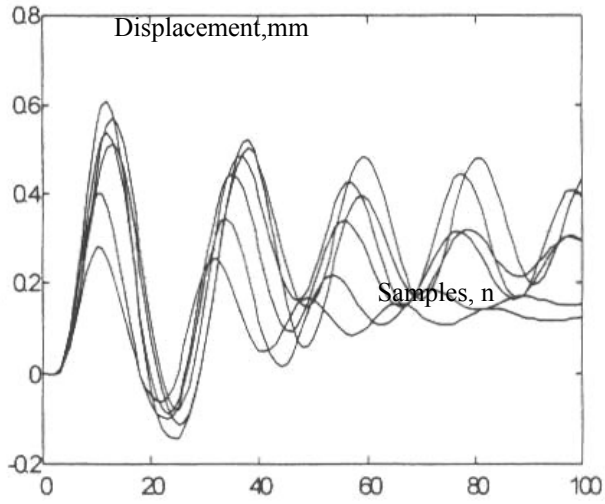

Figure-16. Filtered step response of Figure-15.

CONCLUSIONS

RBF NN algorithm proposed is good at rapid predicting the weighing platform system's K (n) and C (n) parameters before reaching the steady state. The simulation and experimental results and the correlation rate (almost equal to one) between the actual data and the desired data confirm that RBF NN algorithm is good for predicting the parameters accurately.

ACKNOWLEDGMENTS

The co-operation of W. and T. AVERY Ltd. in providing access to a weighing platform is gratefully acknowledged.

REFERENCES

[1] Hunt K.J., Sbarbaro, D., Zbiknowski R., and Gaqthrop P. J. 1992. Neural Networks for Control Systems = A Survey Auto matica. 28(6): 1083-1112.

[2] Bishop C.M., Haynes, P.S., Smith, M.E.V., Todd, T. N. and Trotman D.L. 1994. Real-times Contorl of a High Temprature Plasma Using a Hardware Neural Network" in "Neural Network, Neurao-fuzzy and other learning Systems for Engineering Application and Research. J. A. Powell, Proceedings of an EPSRC Conference, London. April 18th-19th. pp. 16-25.

[3] Tou J. T., Gonzalez R. C. 1974. Pattern Recognition. Reading, MA: Addison-Wesley.

[4] Hartman E.J., Keeler, J.D., and Kowalski J.M. 1990. Layered neural networks with Gaussian hidden units as universal approximations. Neural Computation. 2: 210-215.

[5] Park J. and Sandberg, J.W. 1991. Universal approximation using radial basis functions network Neural Computation. 3: 246-257.

[6] Poggio T. and Girosi F. 1990. Networks for approximation and learning. Proc. of IEEE. 78(9): 1481-1497.

[7] R. L.Riche. 2001. Neural Identification of non-Linear dynamic structures. Journal of Sound and vibration. 248: 247-265.

[8] G. F. Lin. A spatial interpolation method based on radial basis function networks incorporating a semivariogram.

[9] D. Pomerleau. 1994. Neural Network Perception for Mobile Robot Guidance, Kluwer Academic Publishing. Boston, MA.

[10] M. Rosenblum and L.S. Davis An improved radial basis function network for visual autonomous road following. IEEE Transactions on Neural Networks, 7(5): 1111-1120, 1996.

[11] Peng H. Ozaki,T., Ozaki, V.H., Toyoda Y. 2003. A Parameter Optimization Method for Radial Basis Function Type Models. IEEE Trans. Neural Networks. 14: 432-438

[12] Bahar H. B and Horroks D. H. 1998. Dynamic weight estimation using Artificial Neural Network. Artificial Intellegence in Engineering. 12. pp. 135-139.

[13] Tolle H. 1992. Neurocontrol: Learning Control Systems Inspired By Neural Architechtures and Human Problem Solving Strategies. Lecture Notes in Control and Identification Sciences. Springer-Verla. p.172.

[14] Brown M., Harris C. J. 1990. Neurofuzzy Adaptive Modelling and Control. Prentice-Hall.

[15] Girosi F. and Poggio T. 1990. Neural Networks and the Best Approximation Property Biol. Cybernetic. 63. pp. 169-176.

[16] Cichocki A. and Unbehauen R. 1993. Neural Networks for Optizimation and Signal Processing. John Wiley and Sons.

[17] Lowe D. 1994. Non Local Radial Basis Function for Forecasting and Density Estimation IEEE Intern. Conf. on neural networks.11: 1197-1198.

[18] D. Lowe. 1995. Radial basis function networks, In M. A. Arbib, editor, The Handbook of Brain Theory and Neural Networks. MIT Press. Cambridge, MA. pp. 779-782.

[19] F. Rosenblatt. 1992. Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms, Spartan Books. Washington, D.C.

[20] D. S. Broomhead and D. Lowe. 1998. Multi-variable functional interpolation and adaptive networks, Complex Systems. 2(3): 269-303.

[21] J. Moody and C. J. Darken, Fast learning in networks of locally-tuned processing units, Neural Computation. 1: 281-294.

[22] D. Lowe. 1995. Radial basis function networks, In M. A. Arbib, editor, The Handbook of Brain Theory and Neural Networks., MIT Press, Cambridge, MA.

[23] S. Haykin. 1994. Neural Networks: A Comprehensive Foundation. MacMillan College Publishing Company. New York.

[24] F. Schwenker, H. A. Kestler and G. Palm. 2001. Three learning phases for radial-basis-function networks. Neural Networks, 14: 439-458.

[25] Widrow B. and Stearns, S.D. (1985). Adaptive Signal Processing. Englewood Cliffs, NJ: Prentice Hall.

[26] Billings S. A. and W. S. F. Voon. 1986. Correlation based model validity tests for non-linear models.International Journal of Control. 44(1): 235-244.

APPENDIX

Model parameters of the weighing platform system with the parameters of C damping factor, K spring constant, and m (t) as the applied mass, mp as the platform mass, b0 as the platform displacement and b1 as the initial velocity are defined as below in three states of the step response: under damped (u), over damped (o) and critically damped(c):

Under damped (u)

$$q_0 = (m(t) + mp)g/K$$

$$q_1 = 0.5C/(m(t) + mp)$$

$$q_2 = \sqrt{B_1^2 + B_1^2}$$

$$q_3 = \omega_d = \sqrt{K(m(t) + mp)^{-1} - q_1^2}$$

$$q_4 = tan^{-1}(B_1/B_2)$$

$$B_1 = q_0 - b_0$$

$$B_2 = b_1 + B_1 q_1/q_3$$

Critical damped (c)

$$q_0 = (m(t) + mp)g/K$$

$$q_1 = 0.5C/(m(t) + mp)$$

$$q_2 = b_0 - q_0$$

$$q_3 = q_1 q_2 - b_1$$

$$\omega_d = 0$$

Over damped (o)

$$q_0 = (m(t) + mp)g/K$$

$$q_1 = 0.5C/(m(t) + m_p) - \omega_d$$

$$q_2 = -\frac{(q_0 - b_0)q_3 + b_1}{2\omega_d}$$

$$q_3 = 0.5C/(m(t) + mp) + \omega_d$$

$$q_4 = \frac{(q_0 - b_0)q_1 - b_1}{2\varpi_d}$$

$$\varpi_d = \sqrt{\left(\left[\frac{C}{2(m(t) + m_p)}\right]^2 - \frac{K}{m(t) + m_p}\right)}$$