Solving Heat Transfer Equation by Using Feed Forward Neural Networks

Received : 13\3\2014 **Accepted : 14\5\2014**

Alaa Kamel Jaber
Email: alaa77_math@yahoo.co.uk

AL-Qadisiyah University- College of Education - Mathematics Department

Abstract

The aim of this paper is to design feed forward neural network (FFNN) to solve the heat equation. Using a multi-layer with 7 hidden units (neurons) and one linear output unit, the sigmoid activation function of each unit in hidden layer is *tansig* function, where the Levenberg – Marquardt training algorithm is used to train the network .The existence of the proposed solution was proved. The suggested networks have been studied intensively for a few decades and have provided an option for modeling complex systems. Therefore this option was utilized to reduce the computation of solution, and finally the method is demonstrated through illustrative examples.

Keywords : Feed Forward Neural Network, Boundary-initial value problems, heat equation.

1- Introduction

Artificial neural networks have been studied over the last decades and are an excellent option for modeling complex systems. Some of the thermal systems that have been studied with this technique are the prediction of heat transfer coefficients (Jambunathan et al., 1996) [7], estimation of heat transfer in the transition region of a circular tube (Ghajar et al., 2004) [2], friction and heat transfer in helically finned tubes (Zdaniuk et al., 2007) [15], the modeling of evaporative air coolers (Hosoz et al., 2008) [3], the characterization of compact heat exchangers (Ermis,2008) [1], the estimation of thermal performance of plate and tube heat exchangers (Peng and Ling, 2009) [10], performance of finned tube evaporators (Zhao et al., 2010) [16], indirect evaporative cooling (Kiran and Rajput, 2011) [9] and prediction of convective heat transfer in evaporative units (Romero-Mendez et. al.) [11].

The direct problem under consideration consists of a transient heat conduction problem in a slab with adiabatic boundary condition, with an initial temperature profile denoted by $\phi(x)$. Mathematically, the problem can be modeled by the following heat equation:-

$$U_t = \alpha^2 U_{xx} \qquad \text{……………………..(1)}$$

Where $x \in [a,b]$ and $t \geq 0$ with

BCs $\{ U(a,t)=A \quad , \quad U(b,t)=B \}$ and IC $\{ U(x,0)=\phi(x) \}$

We wont to approximate U(x,t) in (1) by using FFNN.

2- Artificial Neural Network [14]

An Artificial neural network (Ann) is a simplified mathematical model of the human brain. It can be implemented by both electric elements and computer software. It is a parallel distributed processor with large numbers of connections, it is an information processing system that has certain performance characters in common with biological neural networks. Ann have been developed as generalizations of mathematical models of human cognition or neural biology, based on the assumptions that:

i. Information processing occurs at many simple elements called neurons that is fundamental to the operation of Ann's.
ii. Signals are passed between neurons over connection links.
iii. Each connection link has an associated weight which, in a typical neural net, multiplies the signal transmitted.
iv. Each neuron applies an action function (usually nonlinear) to its net input (sum of weighted input signals) to determine its output signal.

The units in a network are organized into a given topology by a set of connections, or weights, shown as lines in a diagram .

Ann is characterized by:

i. Architecture: its pattern of connections between the neurons.
ii. Training algorithm : its method of determining the weights on the connections.
iii. Activation function.

Ann are often classified as single layer or multilayer. In determining the number of layers, the input units are not counted as a layer, because they perform no computation. Equivalently, the number of layers in the net can be defined to be the number of layers of weighted interconnects links between the slabs of neurons. This view is motivated by the fact that the weights in a net contain extremely important information.

2.1- Multilayer Feed Forward Neural Network Architecture [14]

In a layered neural network the neurons are organized in the form of layers. We have at least two layers: an input and an outputlayer. The layers between the input and the output layer (if any) are called hidden layers, whose computation nodes are correspondingly called hidden neuronsor hidden units. Extra hidden neurons raise the network’s ability to extract higher-order statistics from (input) data.

The Ann is said to be fullyconnectedin the sense that every node in each layer of the network is connected to every other node in the adjacent forward layer , otherwise the network is called partially connected. Each layer consists of a certain number of neurons; each neuron is connected to other neurons of the previous layer through adaptable synaptic weights w and biases b .

2.2- Training Feed Forward Neural Network [14]

Training is the process of adjusting connection weights w and biases b. In the first step, the network outputs and the difference between the actual (obtained) output and the desired (target) output (i.e., the error) is calculated for the initialized weights and biases (arbitrary values). During the second stage, the initialized weights in all links and biases in all neurons are adjusted to minimize the error by propagating the error backwards (the back propagation algorithm). The network outputs and the error are calculated again with the adapted weights and biases, and the process (the training of the Ann) is repeated at each epoch until a satisfied output y_k(corresponding to the values of the input variables x) is obtained and the error is acceptably small. In most of the training algorithms a learning rate is used to determine the length of the weight update (step size) .

3- The Method

To illustrate the method we will write the approximate solution as :

$$\mu(x,t)=\frac{t^{b-x}(x-a)B+t^{x-a}(b-x)A}{b-a}+[(x-a)(b-x)]^t\phi(x)+t(x-a)(b-x)N(x,t) \quad (2)$$

Where N(x,t) is the output of a FFNN with two input units for x and t.

It's clear that μ satisfy the BC and IC of (1).

Our goal is to design a FFNN, N(x,t) such that μ Fit Informatics unknown function U(x,t) in any accuracy .

Now rewrite (2) to be as follows:-

$$\mathrm{N}(\mathrm{x},\mathrm{t})=\frac{\mu(x,t)-\frac{t^{b-x}(x-a)B+t^{x-a}(b-x)A}{b-a}-[(x-a)(b-x)]^t\phi(x)}{t(x-a)(b-x)} \quad , \quad t\neq 0 \quad , \quad x\neq a,b \quad (3)$$

The right side of (3) is unknown function of two variables , denoted by G(x,t), and the needed FFNN in (2) is the same network required to approximate the function G(x,t), which means that problem (1) has been converted from differential equation problem to approximation function problem by FFNN, which will discuss in the next section.

4- The Existence [6]

One of the earliest works on FFNN with ridge activation functions is in Hecht-Nielson. The author used an improved version of Kolmogorov’s theorem due to Sprecher which states that:

Every continuous function $f:[0,1]^N \rightarrow R$ can be written as:

$$f(x) = \sum_{h=1}^{2N+1}\phi_h\left(\sum_{k=1}^{N}\lambda^h\psi(x_k+\varepsilon h)+h\right) \quad (4)$$

where the real λ and the continuous monotonically increasing function ψ are independent of f, the constant ε is a positive number and the continuous function φ_h, $1 \leq h \leq 2N+1$, depends on f. This equation can be interpreted as a three-layered neural network where the h^{th} hidden node computes the function

$$z_h = \sum_{k=1}^{N} \lambda^h \psi(x_k + \varepsilon h) + h,$$

and the output nodes compute $\sum_{h=1}^{2N+1} \phi_h(z_h)$. However, this is not the network architecture commonly used in practice.

One of the most elegant approaches to prove universal approximation is proposed by Cybenko. By using the Hahn-Banach Theorem and the Riesz Representation Theorem, he showed that if the ridge activation function, σ, is a continuous sigmoid, then the set of $\sum_{i=1}^{N} c_i \sigma(\theta_i^T x + b_i)$ is dense in C(K), where K is a compact set of R^N, with respect to uniform norm. Later, his approach was adopted by many authors to prove their results.[6]

Chui and Li adopted another approach to prove universal approximation. They showed that if the ridge activation function σ, is continuous sigmoid and the direction vector θ satisfies some interpolation conditions, then the set of $\sum_{i=1}^{N} c_i \sigma(\theta_i^T x + b_i)$ is dense in C(K) with respect to uniform norm. They constructed their proof by showing that it is possible to realize *polynomials* as a sum of ridge activation functions.[6]

Since *polynomials* are dense in C(R^N), it follows that the three-layered neural networks are dense in C(R^N) with respect to uniform norm.

In Chen et. al., [6] showed that the continuity assumption usually imposed on the sigmoid functions is unnecessary. Instead, they proved that if the ridge activation function σ, is a bounded sigmoid, then the set of $\sum_{i=1}^{N} c_i \sigma(\theta_i^T x + b_i)$ is dense in C(K) with respect to uniform norm. They also pointed out that in order to prove the neural network in the n-dimensional case, all one needs to do is to prove the case for one dimension .

In Hornik, the author adopted Cybenko's approach to prove universal approximation. He showed the sigmoid assumption usually imposed on the ridge function is unnecessary. Instead, he proved that if the ridge activation function σ, is continuous, bounded and non-constant, then the set of $\sum_{i=1}^{N} c_i \sigma(\theta_i^T x + b_i)$ is dense in C(K) with respect to uniform norm. At the same time, he proved that if the ridge activation function σ, is bounded and non-constant, then the set of $\sum_{i=1}^{N} c_i \sigma(\theta_i^T x + b_i)$ is dense in $L_p(\mu)$ with respect to L_p norm for $1 \le p < \infty$ and a finite measure μ.

Leshno et. Al. provides one of the most general results. They showed that if the ridge activation function σ, is continuous almost everywhere, locally essentially bounded, and not a *polynomial*, then the set of $\sum_{i=1}^{N} c_i \sigma(\theta_i^T x + b_i)$ is dense in C(K) with respect to uniform norm.

4.1- Theorem [6]

Standard Feed Forward Networks with only a single hidden layer can approximate any continuous function uniformly on any compact set and any measurable function to any desired degree of accuracy.

Therefore from the above theorem we have the following:-

1- To approximate any function on R^N we want to determine the number of the hidden nodes , activation functions to hidden layer and training functions.

2- The parameters to this approximation are the weights and biases of nodes in the layers which can calculate by training the FFNN.

3- Any lack of success in applications must arise from inadequate training, insufficient number of hidden units, or the lack of a deterministic relationship between the input and the target.

5 Examples

Now in this section we give some example which illustrate the suggested network

5.1 Example 1

Let us have the following heat conduction problem

$$4U_t = U_{xx}, \quad x \in [0,2] \text{ and } t \geq 0 \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (5)$$

with BCs: $U(0,t)=0$, $U(2,t)=0$ and IC: $U(x,0) = \phi(x) = 2\sin\left(\frac{\pi x}{2}\right) - \sin(\pi x) - \sin(2\pi x)$

The exact solution of Eq. (5) is given by:

$$U(x,t) = 2\sin\left(\frac{\pi x}{2}\right) e^{\frac{-\pi^2 t}{16}} - \sin(\pi x)\, e^{\frac{-\pi^2 t}{4}} - \sin(2\pi x) e^{-\pi^2 t} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (6)$$

Our solution to equation (5) by using (2) is :

$$\mu(x,t) = (2x-x^2)^t \phi(x) + t(2x-x^2)N(x,t) \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (7)$$

Then the FFNN N(x,t) in (7) is the same as to approximate the following function :-

$$G(x,t) = \frac{\mu(x,t) - (2x-x^2)^t \phi(x)}{t(2x-x^2)}, \; t \neq 0 \text{ and } x \neq 0,2 \; \ldots\ldots\ldots (8)$$

To design FFNN which approximate G(x,t) by Theorem 4.1, we choose 7 nodes to hidden layer and the activation function is *'tansig'*, then to training the FFNN we use the function *'trainlm'*.

After we training the FFNN we obtain the parameters illustrated in Table1 and the result is giving in Table 2.

Table1: Training parameter of the suggested network for example 1

Input Weight		Hidden Bias	Hidden Weight	Output Bias
4.943193995	-0.597431269	-2.161127886	-1.032560586	5.868012882
-1.141369003	-2.758773341	1.192524815	-1.637086984	
2.349301891	-3.434468698	0.839751595	3.128773222	
3.301744604	3.775302719	0.867191145	0.744595153	
-3.495976255	-3.407837799	-2.033098557	-0.757759082	
-2.246370758	2.892763372	-3.003247227	-5.291028890	
-1.329680000	-1.193236665	-2.281200040	15.548818803	

Table 2: The results of the example1 using suggested network

X	T	U-exact	U-approximate	$(U_e\text{-}U_p)^2$
0.05	0.1	-0.089870961	-0.112890042	0.000529878
0.15	0.2	0.023159983	-0.049668986	0.005304059
0.25	0.3	0.246995214	0.193612744	0.002849688
0.35	0.4	0.468805803	0.476189268	0.000054516
0.45	0.5	0.664322864	0.743432968	0.006258409
0.55	0.6	0.826453566	0.960426098	0.017948639
0.65	0.7	0.949711452	1.087862662	0.019085757
0.75	0.8	1.030200603	1.127100867	0.009389661
0.85	0.9	1.067074077	1.114970548	0.002294072
0.95	1	1.062705465	1.037712234	0.000624662
1.05	1.1	1.021950790	0.959671257	0.003878740
1.15	1.2	0.951156786	0.906248385	0.002016765
1.25	1.3	0.857272960	0.817980282	0.001543915
1.35	1.4	0.747185424	0.697317576	0.002486802
1.45	1.5	0.627275786	0.550804726	0.005847823
1.55	1.6	0.503167533	0.426107871	0.005938192
1.65	1.7	0.379613397	0.328191230	0.002644239
1.75	1.8	0.260480149	0.211407382	0.002408136
1.85	1.9	0.148794441	0.105483304	0.001875855
1.95	2	0.046821666	0.045806704	0.000001030
			MSE	0.092980837

5.2 Example 2

Let us have the following heat conduction problem

$100 \quad U_t \quad = \quad U_{xx}$ ……………………………………..……………………………………. (9)

Where $x \in [0,\pi]$ and $t \geq 0$ with

BCs: $U(0,t)=0$, $U(\pi,t)=0$

IC: $U(x,0)=\phi(x)= 3\sin(5x)$

And the exact solution to (9) is: $U(x,t) = 3\sin(5x)\, e^{\frac{-t}{4}}$ ……………….…...…………(10)

Our solution to equation (9) by using (2) is given by:

$\mu(x,t)=(\pi x-x^2)^t\phi(x)+t(\pi x-x^2)N(x,t)$ ……………………………………………………..……(11)

Then the FFNN $N(x,t)$ in (11) is the same as to approximate the following function :-

$G(x,t) = \frac{\mu(x,t)-(\pi x-x^2)^t\phi(x)}{t(\pi x-x^2)}$, $t \neq 0$ and $x \neq 0,2$ ………………………………..………(12)

To design FFNN which approximate G(x,t) by Theorem 4.1, we choose 7 nodes to hidden layer and the activation function is *'tansig'*, then to training the FFNN we use the function *'trainlm'*.

After we training the FFNN we obtain the parameters illustrated in Table 3, and the result is giving in Table 4.

Table3: Training parameter of the suggested network for example 2

Input Weight		Hidden Bias	Hidden Weight	Output Bias
-2.884195443	2.324487664	3.702832077	0.514918866	1.641235145
-3.156701897	-2.117427554	1.034811997	5.615858228	
-1.634109484	2.860559826	-0.243589930	5.603264883	
-2.447982616	-0.583445966	0.074654273	-7.004209469	
-3.788825753	1.421502940	-1.790173712	-3.367507905	
2.352653427	1.686907637	1.179311517	-6.039890252	
-2.865290397	-4.380471379	-6.453552673	3.050140174	

Table 4: The results of the example2 using suggested network

X	T	U-exact	U-approximate	$(U_e-U_p)^2$
0.1	0.2	1.368131038	1.368149651	0.000000000
0.3	0.4	2.707712365	2.763524945	0.003115044
0.5	0.6	1.545329244	1.416088904	0.016703066
0.7	0.8	-0.861591049	-0.860353544	0.000001531
0.9	1	-2.283903663	-2.276725712	0.000051523
1.1	1.2	-1.568031386	-1.573409343	0.000028922
1.3	1.4	0.454777480	0.474085428	0.000372797
1.5	1.6	1.886280563	1.858509202	0.000771248
1.7	1.8	1.527413585	1.555377139	0.000781960
1.9	2	-0.136744376	-0.152456116	0.000246859
2.1	2.2	-1.522620906	-1.350978778	0.029461020
2.3	2.4	-1.441375021	-1.433929308	0.000055439
2.5	2.6	-0.103869199	-0.117239402	0.000178762
2.7	2.8	1.197442601	1.204082428	0.000044087
2.9	3	1.324839464	1.321854474	0.000008910
			MSE	0.051821170

6- References

[1] Ermis K., (2008), "ANN Modeling of Compact Heat Exchangers". International Journal of Energy Research, volume 32, 2008: 581-594.

[2] Ghajar A.J., Tam L.M., Tam S.C., (2004), "Improved Heat Transfer Correlation in the Transition Region for a Circular Tube with Three Inlet Configurations Using Artificial Neural Networks" . Heat Transfer Engineering, volume 25: 30-40.

[3] Hosoz M., Ertunc H.M., Ozguc A.F.,(2008), "Modelling of a Direct Evaporative Air Cooler Using Artificial Neural Network". International Journal of Energy Research, volume 32:83-89.

[4] Hussain Eman Ali, Al_Saif Nahdh. S. M., (2013), " Design feed forward neural network for solving two dimension singularly perturbed integro-differential and integral equation", International Journal of Applied Mathematical Research, 2 (1) : 134-139.

[5] Hwang S. Deng, Y., (2006), " Applying neural networks to the solution of forward and inverse heat conduction problems", International Journal of Heat and Mass Transfer 49 : 4732–4750.

[6] Jabber , A. K., (2009), "On Training Feed Forward Neural Networks for Approximation Problem", MSc Thesis, Baghdad University, College of Education (Ibn Al-Haitham).

[7] Jambunathan K., Hartle S.L., Ashforth F.S., Fontama V.M., (1996), "Evaluating Convective Heat Transfer Coefficient Using Neural Networks". International Journal of Heat and Mass Transfer, vol. 39 (11): 2329-2332.

[8] John H. Lienhard IV, John H. Lienhard IV, (2000), "A Heat Transfer Textbook", Third Edition.

[9] Kiran T.R., Rajput S.P.S., (2011), "An Effectiveness Model for an Indirect Evaporative Cooling (IEC) System: Comparison of Artificial Neural Networks (ANN), Adaptive Neuro-Fuzzy Inference System (ANFIS) and Fuzzy Inference System (FIS) Approach". Applied Soft Computing, vol. 11: 3525-3533.

[10] Peng H., Ling X., (2009), "Neural Networks Analysis of Thermal Characteristics on Plate-Fin Heat Exchangers with Limited Experimental Data". Applied Thermal Engineering, volume 29: 2251-2256.

[11] Ricardo Romero-Mendez, Martín Durán-García Héctor, Manuel Hidalgo-López Juan, Arturo Pacheco-Vega, (2014), "Use of Artificial Neural Networks for Prediction of Convective Heat Transfer in Evaporative Units", Ingeniería Investigación y Tecnología, volumen XV (número 1), enero-marzo 2014: 93-101 ISSN 1405-7743 FI-UNAM.

[12] Tawfiq Luma N. M., Hussein Ashraf A. T., (2013), " Design Feed Forward Neural Network to Solve Singular Boundary Value Problems", ISRN Applied Mathematics, Volume 2013, Article ID 650467, 7 pages.

[13] Tawfiq L. N. M., Ibraheem G .H., (2012), " Designing Feed Forward Neural Network for Solving Linear VolterraIntegro-Differential Equations", Ibn Al-Haitham Journal for Pure and Applied Science , NO. 3, Vol. 25:316-324.

[14] Tawfiq L. N. M., Oraibi Yaseen A., (2013), "Fast Training Algorithms for Feed Forward Neural Networks", Ibn Al-Haitham Journal for Pure and Applied Science , NO. 1, Vol. 26:275-280.

[15] Zdaniuk G.J., Chamra L.M., Walters D.K., (2007), "Correlating Heat Transfer and Friction in Helically-Finned Tubes Using Artificial Neural

Networks". International Journal of Heat and Mass Transfer, volume 50: 4713-4723.

[16] Zhao L.X., Yang L., Zhang C.L., (2010), "Network Modeling of Fin-and-Tube Evaporator Performance Under Dry and Wet Conditions". ASME Journal of Heat Transfer, volume 132, Art. 074502.

حل معادلة التوصيل الحراري باستخدام الشبكات العصبية ذات التغذية التقدمية

تاريخ الاستلام : 2014\3\13 تاريخ القبول : 2014\5\14

علاء كامل جابر

Email: alaa77_math@yahoo.co.uk

جامعة القادسية – كلية التربية – قسم الرياضيات

الخلاصة

الهدف من البحث هو تصميم شبكة عصبية ذات تغذية تقدمية لحل معادلة التوصيل الحراري. تم استخدام شبكة متعددة الطبقات ذات طبقة خفية واحدة تحتوي على 7 وحدات (عصبونات) ووحدة اخراج خطية واحدة، دالة التنشيط المستخدمة لكل وحدة في الطبقة المخفية كانت الدالة (*tansig*) وتم استخدام خوارزمية التدريب (*trainlm*). تم اثبات وجود الحل للشبكة المقترحة. تم دراستها بشكل مكثف منذ بضعة عقود حيث قدمت خيارا لنمذجة الانظمة الصعبة. لذلك هذا الخيار استخدم لتقليل الحسابات في اثناء الحل ، واخيرا تم توضيح الشبكة المقترحة من خلال الأمثلة التوضيحية.

الكلمات المفتاحية : الشبكات العصبية ذات التغذية التقدمية ، مسائل القيم الحدودية والابتدائية، التوصيل الحراري.