A NOVEL NUMERICAL METHOD FOR HEAT EQUATION

by

Yun HE and Haibin LI *

College of Science, Inner Mongolia University of Technology, Hohhot, China

Short paper
DOI: 10.2298/TSCI1603018H

A neural network computation is proposed to solve a heat equation, and an example is given to elucidate its simulation efficiency. The algorithm developed in this paper can be used as a paradigm for many other numerical applications.

Key words: *neural network computation, multiple neural networks, training, heat equation, rational neural network*

Introduction

How to obtain approximate solutions reasonably and effectively to heat equations has attracted wide attention [1-4]. There are also many methods to solve a non-linear problem through a neural network and its learning algorithm [5-8]. In this paper, we will convert a heat equation to a joint training problem of a multiple neural network.

Consider a heat equation without a heat source:

$$\frac{\partial u}{\partial t} \quad k \quad \frac{\partial^2 u}{\partial x_1^2} \quad \frac{\partial^2 u}{\partial x_2^2} \quad 0 \tag{1}$$

where $u(x_1, x_2, t)$ is the temperature and k is the thermal diffusivity.

Now we consider a heat equation with heat fluxes:

$$\frac{\partial^2 u}{\partial x_1^2} \quad \frac{\partial^2 u}{\partial x_2^2} \quad (\lambda^2 \quad \mu^2)\exp(\lambda x_1 \quad \mu x_2) \tag{2}$$

with the following boundary conditions:

$$u \quad \exp(\lambda x_1 \quad \mu x_2), \quad x_2 \quad 0 \quad \text{and} \quad x_2 \quad 1 \tag{3}$$

$$\frac{\partial u}{\partial x_1} \quad \lambda \exp(\lambda x_1 \quad \mu x_2), \quad x_1 \quad 0 \quad \text{and} \quad x_1 \quad 1 \tag{4}$$

where $0 \quad x_1 \quad 1$ and $0 \quad x_2 \quad 1, \lambda$ and μ are constants.

Analysis of the method

Assume that the solution of eq. (2) can be expressed in the form:

$$u \quad \vec{\mathrm{w}}^T f(\vec{\mathrm{K}}x_1 \quad \vec{\mathrm{K}}_1 x_2 \quad \vec{\mathrm{b}}) \tag{5}$$

* Corresponding author; e-mail: lhbnm2003@126.com

where the vectors $\vec{w}^T$, $\vec{b}$, $\vec{K}$, and $\vec{K}_1$ are unknown to be further determined, and f is a continuous function, by a simple calculation, we have:

$$\frac{du}{dx_1} = \vec{w}^T \vec{K}^T f(\vec{K}x_1 + \vec{K}_1 x_2 + \vec{b}) \tag{6}$$

$$\frac{d^2u}{dx_1^2} = \vec{w}^T \vec{K}^T \vec{K}^T f(\vec{K}x_1 + \vec{K}_1 x_2 + \vec{b}) \tag{7}$$

$$\frac{d^2u}{dx_2^2} = \vec{w}^T \vec{K}_1^T \vec{K}_1^T f(\vec{K}x_1 + \vec{K}_1 x_2 + \vec{b}) \tag{8}$$

Let $f(x) = \exp(x)$ represent the network activation function. According to eqs. (7) and (8), we have:

$$\frac{d^2u}{dx_1^2} + \frac{d^2u}{dx_2^2} = \vec{w}^T (\vec{K}^T \vec{K}^T + \vec{K}_1^T \vec{K}_1^T) f(\vec{K}x_1 + \vec{K}_1 x_2 + \vec{b}) \tag{9}$$

Based on eqs. (5), (6), and (9), three neural networks were constructed. They were recorded as network I, network II, and network III, eq. (3), (4), and (2), respectively. Weights from the input layer (x_1 and x_2) to the hidden layer of these two networks are $\vec{K}$ and $\vec{K}_1$, respectively and the threshold vector is $\vec{b}$. The weight vector from the input layer to the hidden layer in the network I is $\vec{w}^T$, that of the network II is $\vec{w}^T \vec{K}^T$ and that of the network III is $\vec{w}^T (\vec{K}^T \vec{K}^T + \vec{K}_1^T \vec{K}_1^T)$. For the network I, x_1 within [0, 1] is divided into 11 parts uniformly, $x_2 = 0$ and $x_2 = 1$. For the network II, x_2 within [0, 1] is divided into 11 parts uniformly, $x_1 = 0$ and $x_1 = 1$. For the network III, x_1 and x_2 within [0, 1] are divided into 11 parts uniformly, intersections between any two parts compose the input samples. They serve as the input samples. There are s neurons in the hidden layer of neural networks. In this paper, we take $s = 50$ as an example.

The performance functions of the i^{th} sample point of the network I, the network II and the network III are, respectively:

$$E1_i = \vec{w}^T f(\vec{K}x_1 + \vec{K}_1 x_2 + \vec{b}) - u_{1i} \tag{10}$$

$$E2_i = \vec{w}^T \vec{K}^T f(\vec{K}x_1 + \vec{K}_1 x_2 + \vec{b}) - u_{2i} \tag{11}$$

$$E3_i = \vec{w}^T (\vec{K}^T \vec{K}^T + \vec{K}_1^T \vec{K}_1^T) f(\vec{K}x_1 + \vec{K}_1 x_2 + \vec{b}) - u_{3i} \tag{12}$$

Performance function of the network system is:

$$E(\vec{w}, \vec{K}, \vec{K}_1, \vec{b}) = \frac{1}{2}\left(\sum_{i=1}^{q1} E1_i^2 + \sum_{i=1}^{q2} E2_i^2 + \sum_{i=1}^{q3} E3_i^2\right) \tag{13}$$

In this case, $q1 = 22$, $q2 = 22$, and $q3 = 121$.

Results and discussion

After 100 steps of joint training of the networks I, II, and III, E tends to be stable. $\vec{w}, \vec{K}, \vec{K}_1$, and $\vec{b}$ are derived and then substituted into eq. (5) to simulate the following testing sample points, thus getting the numerical solution of the proposed algorithm. Numerical simulation results of the proposed algorithm are compared with theoretical results as given in tabs. 1 and 2. Calculation error is shown in fig. 1.

Table 1. Numerical prediction of the proposed algorithm

x_1 \ x_2	0	0.2	0.3	0.4	0.6	0.7	0.8	1
0	1.0001	1.4919	1.8221	2.2255	3.3200	4.0550	4.9528	7.3886
0.1	1.3500	2.0138	2.4596	3.0041	4.4815	5.4737	6.6856	9.9737
0.2	1.8222	2.7183	3.3201	4.0551	6.0495	7.3888	9.0247	13.4633
0.3	2.4597	3.6693	4.4817	5.4739	8.1660	9.9739	12.1822	18.1737
0.4	3.3202	4.9530	6.0496	7.3890	11.0230	13.4635	16.4443	24.5321
0.5	4.4817	6.6859	8.1661	9.9741	14.8795	18.1739	22.1977	33.1150
0.6	6.0497	9.0250	11.0231	13.4636	20.0854	24.5323	29.9638	44.7008
0.7	8.1662	12.1825	14.8797	18.1741	27.1125	33.1152	40.4471	60.3400
0.8	11.0232	16.4446	20.0855	24.5324	36.5981	44.7010	54.5979	81.4506
0.9	14.8798	22.1979	27.1126	33.1154	49.4023	60.3401	73.6996	109.947
1	20.0856	29.9641	36.5982	44.7011	66.6862	81.4507	99.4841	148.413

Table 2. Exact values

x_1 \ x_2	0	0.2	0.3	0.4	0.6	0.7	0.8	1
0	1.0000	1.4918	1.8221	2.2255	3.3201	4.0552	4.9530	7.3891
0.1	1.3499	2.0138	2.4596	3.0042	4.4817	5.4739	6.6859	9.9742
0.2	1.8221	2.7183	3.3201	4.0552	6.0496	7.3891	9.0250	13.4637
0.3	2.4596	3.6693	4.4817	5.4739	8.1662	9.9742	12.1825	18.1741
0.4	3.3201	4.9530	6.0496	7.3891	11.0232	13.4637	16.4446	24.5325
0.5	4.4817	6.6859	8.1662	9.9742	14.8797	18.1741	22.1980	33.1155
0.6	6.0496	9.0250	11.0232	13.4637	20.0855	24.5325	29.9641	44.7012
0.7	8.1662	12.1825	14.8797	18.1741	27.1126	33.1155	40.4473	60.3403
0.8	11.0232	16.4446	20.0855	24.5325	36.5982	44.7012	54.5982	81.4509
0.9	14.8797	22.1980	27.1126	33.1155	49.4024	60.3403	73.6998	109.947
1	20.0855	29.9641	36.5982	44.7012	66.6863	81.4509	99.4843	148.413

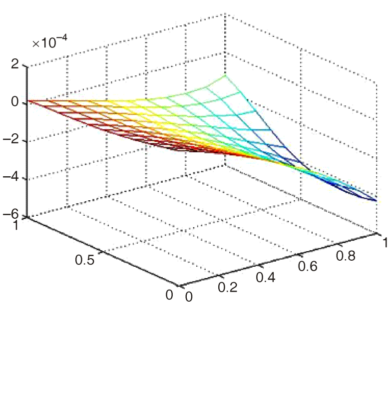

Figure 1. Absolute error value between numerical results and exact ones on testing sample points

The mean absolute error of the proposed algorithm is as low as $1.9\ 10^{-4}$, while that in [8] is $1.66\ 10^{-2}$, showing our algorithm is robust and reliable.

Conclusions

This paper puts forward a neural network construction method for heat equations. It constructs neural networks and network training samples according to heat equations and its boundary conditions. For the purpose of synchronous training of multiple neural networks, this paper developed a joint training algorithm of a multiple neural network, which is feasible and effective through the case study. This paper has made some great contributions: (1) a problem-oriented neural network construction method is put forward to solving the heat equation; (2) It realizes joint training of multiple neural networks with related parameters; (3) The proposed algorithm is applied to solve the heat equation successfully.

When the proposed algorithm is used in neural network construction, it does not view neural network as a black boxy system any more, it combines known knowledge (equation) of problems to make neural network construction more rational. It provides a new idea for theoretical development and application of neural networks.

Acknowledgment

This study was supported by the National Natural Science Foundation of China under Grant No.11262014.

References

[1] Yan, S. P., Local Fractional Laplace Series Expansion Method for Diffusion Equation Arising in Fractal Heat Transfer, *Thermal Science, 19* (2015), 1, pp. 131-135
[2] Liu, J. F., Modified Variational Iteration Method for Variant Boussinesq Equation, *Thermal Science, 19* (2015), 4, pp. 1195-1199
[3] Qian, Y. H., Guo, Q. W., A New Derivative-Free Iterative Method for Solving Nonlinear Equations, *Nonlinear Science Letters A, 7* (2016), 2, pp. 32-40
[4] Zhang, Y. Z., *et al*., Initial Boundary Value Problem for Fractal Heat Equation in the Semi-Infinite Region by Yang-Laplace Transform, *Thermal Science, 18* (2014), 2, pp. 677-681
[5] Leung, H. F. F., *et al*., Tuning of the Structure and Parameters of a Neural Network Using an Improved Genetic Algorithm, *IEEE Transaction on Neural Networks, 14* (2003), 1, pp. 79-88
[6] Zhang, J. R., *et al*., A Hybrid Particle Swarm Optimization-Back-Propagation Algorithm for Feedforward Neural Network Training, *Applied Mathematics and Computation, 185* (2007), 2, pp. 1026-1037
[7] Zhao, Z., *et al.*, Soft Sensing of Coal Quality*, Thermal Science, 19* (2015), 1, pp. 231-242
[8] Li, J. Y, *et al.*, Numerical Solution of Elliptic Partial Differential Equation Using Radial Basis Function Neural Networks, *Neural Networks, 16* (2003), 5-6, pp. 729-734

Paper submitted: December 10, 2015
Paper revised: June 1, 2015
Paper accepted: June 6, 2016