doi: 10.3969/j.issn.0490-6756.2020.

热传导界面问题的一种神经网络算法

王铸础，谢小平
(四川大学数学学院，成都 610064)

摘 要：本文讨论了一种单隐层神经网络算法在数值求解热传导界面问题中的应用.该算法设定含有神经网络函数的近似解满足初边值条件和 Dirichlet 界面条件,并通过求解由原方程导出的关于神经网络权重的离散优化问题来训练近似解中的神经网络,以使得近似解逼近真解.此外,本文还给出了一种基于随机梯度法思想的类随机梯度法来求解相应的离散优化问题.数值算例验证了算法的有效性.

关键词：热传导方程;界面问题;神经网络;随机梯度下降法
中图分类号：O242.2 **文献标识码**：A **文章编号**：0490-6756(2020)06-1038-09

A neural network algorithm for the heat conduction interface problems

WANG Zhu-Chu, *XIE Xiao-Ping*
(School of Mathematics, Sichuan University, Chengdu 610064, China)

Abstract: This paper discusses a single hidden layer neural network algorithm for the heat conduction interface problems. The proposed algorithm sets the approximate solution to contain a neural network function and to meet the initial/boundary conditions and the Dirichlet interface condition, and trains the neural network by solving the discrete optimization problem derived from the original system so as to obtain the approximate solution. A new optimization algorithm based on the stochastic gradient method is also given to solve the corresponding discrete optimization problem. Numerical examples are provided to verify the effectiveness of the neural network algorithm.

Keywords: Heat conduction equation; Interface problem; Neural network; Stochastic gradient descent method
(2010 MSC 35Q68)

1 引 言

近年来,神经网络在机器学习、人工智能领域取得了极大进展,被广泛应用于计算机视觉,自然语言处理,强化学习等诸多方面. 其中应用神经网络算法求解偏微分方程已成为一个热门的研究领域[1-17].

1998 年,Lagaris 和 Likas[9] 提出了一种用于求解偏微分方程初边值问题的人工神经网络(ANN)算法.该算法将近似解表示为两项之和:第一项的函数满足初边值条件;第二项为一个在满足在初始及边界上恒等于 0 的函数与一个前馈神经网络函数的乘积,并通过使用前馈神经网络对连续函数的逼近性能,以使近似解逼近方程真解.为训练第二项中的神经网络, 该算法将其权重作为自变量,以极小化一个适当的损失函数.由于单隐层

收稿日期：2019-04-19
基金项目：国家自然科学基金(11771312)
作者简介：王铸础(1995－)，男，硕士研究生，主要研究方向为微分方程数值解. E-mail：wangzhuchu@foxmail.com
通讯作者：谢小平. E-mail：xpxie@scu.edu.cn

神经网络可以以任意精度逼近连续函数[2]，所以该类神经网络可被用于求解微分方程. 文献[10]讨论了边界不规则时的推广算法. 2010 年，Baymani，Kerayechian 及 Effati 使用文献[9]中的神经网络算法求解了 Stokes 方程[1]. 2017 年，Raissi，Perdikaris，Karniadakis 提出了一种基于高斯过程的神经网络算法[15]，并用其求解线性微分方程. 2017 年，Long，Lu 和 Ma 提出了求解偏微分方程的一种所谓“Pde-net”的神经网络[12]，其特点是使用卷积核学习微分算子. 2018 年，Han，Jentzen 和 E[4] 提出了一种通过使用多层神经网络计算随机游走项相关梯度，以求解高维向后随机微分方程的算法. 基于 Ritz 法与神经网络，E 和 Yu 提出了 Deep Ritz 方法[3]；He，Li，Xu 和 Zheng 从使用 RELU 激活函数的神经网络函数是分片线性多项式这一角度出发，证明了自由度为 N 的 d 维线形有限元函数可由至多 $[\log_n d]+1$ 层 ReLu 神经网络表出[5]. Sirignano 和 Spiliopoulos 提出了一种求解偏微分方程的深度学习算法，其损失函数为方程残差与初边值条件残差的 L^2 范数的平方和[16]. 2019 年，He 和 Xu 在文献[6]中讨论了卷积神经网络与多重网格之间的关系.

本文将研究 Lagaris 和 Likas 所提出的神经网络算法[9]在热传导界面问题中的推广应用. 我们在每个子区域构造形如 $A(x,t)+B(x,t)*N(x,t)$ 的近似解，其中 $A(x,t)$ 满足原方程的初边值条件，$B(x,t)$ 为一在初始时刻和边界上恒等于零的函数，$N(x,t)$ 为神经网络函数. 对于界面条件的处理，我们将 Dirichlet 界面条件视为边界条件强制满足，将 Neumann 界面条件的残差作为损失函数的一部分. 最后，通过极小化损失函数，我们得到优化的神经网络函数权重，从而最终获得问题的近似解. 针对一维问题，我们使用 Adam 算法求解离散优化问题. 针对高维情形，我们给出了一种基于随机梯度法思想的类随机梯度法来求解相应的离散优化问题. 本文的神经网络算法既保留了原算法近似解表达形式简洁、适用范围广等优点，又易于处理界面条件，便于推广应用于其它类型的界面问题.

2　预备知识

令 $\mathbf{N}^+$ 为正整数集. 给定 $n,m\in\mathbf{N}^+$，定义线性函数

$$\Theta^{n\to m}(\tilde{x})=W\tilde{x}+b \tag{1}$$

其中 $\tilde{x}\in\mathbf{R}^n$，$W=[w_{ij}]\in\mathbf{R}^{m\times n}$，$b\in\mathbf{R}^m$. 定义非线性激活函数

$$\hat{\sigma}(x)=\frac{1}{1+\exp(-x)},x\in\mathbf{R} \tag{2}$$

给定整数 $s,r\geqslant 1$ 和矩阵 $[Q_{ij}]\in\mathbf{R}^{s\times r}$，定义

$$\sigma([Q_{ij}])=[\hat{\sigma}(Q_{ij})]\in\mathbf{R}^{s\times r} \tag{3}$$

给定 $d,c,k\in\mathbf{N}^+$，另给定 $n_1,\cdots,n_k\in\mathbf{N}^+$，并令 $n_0=d$，$n_{k+1}=c$，定义具有 k 隐层的前馈神经网络函数 $f:\mathbf{R}^d\to\mathbf{R}^c$ 如下：$\forall\,\tilde{x}\in\mathbf{R}^d$，

$$\begin{aligned}f(\tilde{x})=&\Theta^{n_k\to n_{k+1}}\circ\sigma\circ\Theta^{n_{k-1}\to n_k}\circ\sigma\cdots\circ\\&\Theta^{n_1\to n_2}\circ\sigma\circ\Theta^{n_0\to n_1}(\tilde{x})\in\mathbf{R}^c\end{aligned} \tag{4}$$

本文考虑标量型（$c=1$）单隐层（$k=1$）且隐层神经元数为 $M(\geqslant 1)$ 的神经网络，对应的神经网络函数空间记为 ANN_1^M：

$$\begin{aligned}ANN_1^M(\widetilde{\Omega})=\{&f(\tilde{x}):f(\tilde{x})=\\&\Theta^{M\to 1}\circ\sigma\circ\Theta^{d\to M}(\tilde{x})=\\&\sum_{i=1}^{M}\alpha_i\sigma(W_i\tilde{x}+b_i)+\beta,\tilde{x}\in\widetilde{\Omega}\subset\mathbf{R}^d,\\&\alpha_i,b_i,\beta\in\mathbf{R},\mathrm{W}_i\in\mathbf{R}^{1\times d},i=1,2,\cdots,M\}\end{aligned} \tag{5}$$

其中 $\widetilde{\Omega}$ 为 $\mathbf{R}^d$ 中的有界子集.

定理 2.1　当 $M\to\infty$ 时，$ANN_1^M(\widetilde{\Omega})$ 在 $C^0(\widetilde{\Omega})$ 中稠密.

注 1　为方便起见，后文用 p 表示式(5)中的参数 α_i,b_i,W_i,β，并记

$$p=(\alpha_1,\cdots,\alpha_M,b_1,\cdots,b_M,W_1,\cdots,W_M,\beta).$$

我们称其为神经网络函数的权重.

3　算　法

设 Ω 为有界区域，$\Omega\subset\mathbf{R}^d$，$d=1,2,3$. 一个光滑的界面将区域 Γ 分成 Ω_1,Ω_2 两部分. 设 n_1,n_2 分别为 Ω_1,Ω_2 在界面 Γ 上的单位外法向量. 设时间长度 $T>0$. 考虑如下的热传导初边值界面问题：求解 $u(x,t),v(x,t)$，满足

$$u_t-\nabla\cdot(a_1\nabla u)=f_1,(x,t)\in\Omega_1\times(0,T] \tag{6}$$

$$u_t-\nabla\cdot(a_2\nabla u)=f_2,(x,t)\in\Omega_2\times(0,T] \tag{7}$$

$$u=g_1,(x,t)\in\partial\Omega_1\backslash\Gamma\times(0,T] \tag{8}$$

$$u=g_2,(x,t)\in\partial\Omega_2\backslash\Gamma\times(0,T] \tag{9}$$

$$u(x,0)=u_0,x\in\Omega_1 \tag{10}$$

$$v(x,0)=v_0,x\in\Omega_2 \tag{11}$$

$$u-v=\varphi,(x,t)\in\Gamma\times(0,T] \tag{12}$$

$$a_1\nabla u\cdot n_1+a_2\nabla v\cdot n_2=\psi,(x,t)\in\Gamma\times(0,T] \tag{13}$$

其中 $a_1,a_2>0$ 为热传导系数，f_1,f_2 为源项，f_1，$f_2,g_1.\ g_2.\ u_0.\ v_0,\varphi,\psi$ 均为各自定义域上的连续函数.

引入损失函数

$$Loss_u(u(x,t))=(u_t(x,t)-\nabla\cdot(a_1\nabla u(x,t))-f_1(x,t))^2,$$

$$Loss_v(v(x,t))=(v_t(x,t)-\nabla\cdot(a_2\nabla v(x,t))-f_2(x,t))^2,$$

$$Loss_J(u(x,t),v(x,t))=(a_1\nabla u(x,t)\cdot n_1+a_2\nabla v(x,t)\cdot n_2-\psi(x,t))^2.$$

给定参数 $\alpha>0$，定义如下优化问题：

$$\min_{u,v}\{\sum_{(x,t)\in D_1}Loss_v u(u(x,t))+\sum_{(x,t)\in D_2}Loss_v(v(x,t))+\alpha\sum_{(x,t)\in D_3}Loss_J(u(x,t),v(x,t))\} \tag{14}$$

其中

$$D_1=\{(x,t)\,|\,x\in\Omega_1,t\in[0,T]\},$$

$$D_2=\{(x,t)\,|\,x\in\Omega_2,t\in[0,T]\},$$

$$D_3=\{(x,t)\,|\,x\in\Gamma,t\in[0,T]\}.$$

易知，原问题(式 6～13)的真解为此优化问题的最优解. 本文的神经网络算法的基本思路是：首先利用神经网络函数构造满足初边值条件和 Dirichlet 界面条件的近似解 U,V，然后求解如下的关于神经网络权重的离散优化问题：

$$\min_{U,V}\{\sum_{(x,t)\in X_1}Loss_u(U(x,t))+\sum_{(x,t)\in X_2}Loss_v(V(x,t))+\sum_{(x,t)\in X_3}(\alpha Loss_J(U(x,t),V(x,t)))\} \tag{15}$$

其中 $X_i\subset D_i(i=1,2,3)$ 为给定的离散点集(训练集).

下面给出近似解 U,V 的构造步骤.

第一步，设

$$U(x,t,p_1)=A_1(x,t)+B_1(x,t)*N_1(x,t,p_1),(x,t)\in\Omega_1\times[0,T] \tag{16}$$

其中 A_1 和 B_1 满足

$$A_1(x,t)=g_1,(x,t)\in\partial\Omega_1\backslash\Gamma\times(0,T],$$

$$A_1(x,0)=u_0,x\in\Omega_1,$$

$$B_1(x,t)=0,(x,t)\in\partial\Omega_1\backslash\Gamma\times(0,T],$$

$$B_1(x,0)=0,x\in\Omega_1,$$

$N_1(x,t,p_1)$ 为以 $p_1=(\alpha_1^{(1)},\cdots,\alpha_{M_1}^{(1)},b_1^{(1)},\cdots,b_{M_1}^{(1)},W_1^{(1)},\cdots,W_{M_1}^{(1)},\beta^{(1)})$ 为权重的单隐层神经网络函数，

$$N_1(x,t,p_1)=\sum_{i=1}^{M}\alpha_i^{(1)}\sigma(W_i^{(1)}[x,t]+b_i^{(1)})+\beta^{(1)},x\in\Omega_1,t\in[0,T] \tag{17}$$

第二步，设

$$V(x,t,p_2)=A_2(x,t)+B_2(x,t)*N_2(x,t,p_2),(x,t)\in\Omega_2\times[0,T] \tag{18}$$

其中 A_2 和 B_2 满足

$$A_2(x,t)=\begin{cases}g_2,\ (x,t)\in\partial\Omega_2\backslash\Gamma\times(0,T],\\ \varphi-U,\ (x,t)\in\Gamma\times(0,T],\end{cases}$$

$$A_2(x,0)=v_0,\ x\in\Omega_2,$$

$$B_2(x,t)=0,\ (x,t)\in\partial\Omega_2\times(0,T],$$

$$B_2(x,0)=0,\ x\in\Omega_2,$$

$N_2(x,t,p_2)$ 为以 $p_2=(\alpha_1^{(2)},\cdots,\alpha_{M_2}^{(2)},b_1^{(2)},\cdots,b_{M_2}^{(2)},W_1^{(2)},\cdots,W_{M_2}^{(2)},\beta^{(2)})$ 为权重的神经网络函数，其形式类似式(17).

将上述 U,V 代入离散优化问题(15)，则离散优化问题转化为关于权重 p_1,p_2 的优化问题，并通过优化算法进行求解.

注 2 构造近似解 U 的关键在于构造满足关于 u 的初边值条件的函数 A_1，构造近似解 V 的关键在于构造满足关于 v 的初边值条件与 Dirichlet 界面条件的函数 A_2.

注 3 由于激活函数 $\hat{\sigma}(x)$ 是强非线性的，关于权重 p_1,p_2 的离散优化问题(15) 也是强非线性的，其优化算法的选择将直接影响问题求解的精度与效率.

为验证算法的有效性，下面定义三种形式的误差：

$$E_{\mathrm{R}}=\frac{\sum_{(x,t)\in X_1}Loss_u(u(x,t))+\sum_{(x,t)\in X_2}Loss_v(v(x,t))+\alpha\sum_{(x,t)\in X_3}Loss_J(u(x,t),v(x,t))}{|X_1|+|X_2|+|X_3|},$$

$$E_{\mathrm{MS}}=\frac{\sum_{(x,t)\in X_1}(u(x,t)-U(x,t))^2+\sum_{(x,t)\in X_2}(v(x,t)-V(x,t))^2}{|X_1|+|X_2|},$$

$$E_2=\frac{\left(\int_{\Omega_1\times T}\left|u-U\right|^2\mathrm{d}x\mathrm{d}t+\int_{\Omega_2\times T}\left|v-V\right|^2\mathrm{d}x\mathrm{d}t\right)^{\frac{1}{2}}}{\left(\int_{\Omega_1\times T}\left|u\right|^2\mathrm{d}x\mathrm{d}t+\int_{\Omega_2\times T}\left|v\right|^2\mathrm{d}x\mathrm{d}t\right)^{\frac{1}{2}}},$$

其中$|X_1|,|X_2|,|X_3|$分别表示优化问题(14)中离散点集 X_1,X_2,X_3 包含的点的个数.

4　数值实验

算例 4.1　考虑一维变系数问题(6)～(13)，其中 $\Omega_1=[0,\frac{2}{3}),\Omega_2=(\frac{2}{3},1],T=1$，热传导系数

$$a_1(x)=0.1-0.09x,x\in[0,\frac{2}{3}],$$

$$a_2(x)=0.01,x\in(\frac{2}{3},1],$$

源项

$$f_1(x,t)=0.09\pi\mathrm{e}^{(-0.1\pi^2t)}(\cos(\pi x)-\pi x\sin(\pi x)),(x,t)\in[0,\frac{2}{3}]\times[0,T],$$

$$f_2(x,t)=0.06\pi^2e^{(-0.1\pi^2t)}\sin(4\pi x),\quad(x,t)\in(\frac{2}{3},1]\times[0,T],$$

初值条件

$$u_0(x)=\sin(\pi x),x\in[0,\frac{2}{3}],$$

$$v_0(x)=\sin(4\pi x),x\in(\frac{2}{3},1],$$

边值条件

$$g_1(0,t)=g_2(1,t)=0,\quad t\in[0,t],$$

界面条件

$$\varphi(\frac{2}{3},t)=\psi(\frac{2}{3},t)=0,\ t\in[0,1].$$

方程的真解为

$$u(x,t)=\mathrm{e}^{(-0.1\pi^2t)}\sin(\pi x),\quad(x,t)\in[0,\frac{2}{3}]\times[0,T],$$

$$v(x,t)=\mathrm{e}^{(-0.1\pi^2t)}\sin(4\pi x),\quad(x,t)\in(\frac{2}{3},1]\times[0,T].$$

按照上一节给出的神经网络算法步骤，我们首先构造近似解

$$U(x,t)=(1-t)*\sin(\pi x)+x*t*N_1(x,t,p_1),$$

$$V(x,t)=(1-t)*\sin(4\pi x)+(3-3x)*\{U(\frac{2}{3},t)-(1-t)*\sin(\frac{2}{3}*\pi)\}+(1-x)*(3x-2)*t*N_2(x,t,p_2),$$

其中神经网络 N_1,N_2 的隐层神经元数 $M_1=M_2=10$. 离散优化问题(14)中的离散训练点集取为：

$$X_1=\{(0+\frac{1}{30}*i,0+\frac{1}{30}*j)\mid 0<i\leqslant 20,\ 0<j\leqslant 30,i,j\in\mathbf{N}^+\},$$

$$X_2=\{(\frac{2}{3}+\frac{1}{30}*i,0+\frac{1}{30}*j)\mid 0<i\leqslant 10,\ 0<j\leqslant 30,i,j\in\mathbf{N}^+\},$$

$$X_3=\{(\frac{2}{3},0+\frac{1}{30}*j)\mid 0<j\leqslant 30,j\in\mathbf{N}^+\}.$$

将 U,V 代入式(15)，取参数 $\alpha=1$，然后使用 Adam 算法[7]求解，步长取 0.01，其余可调参数均取默认值；神经网络权重初值随机生成.

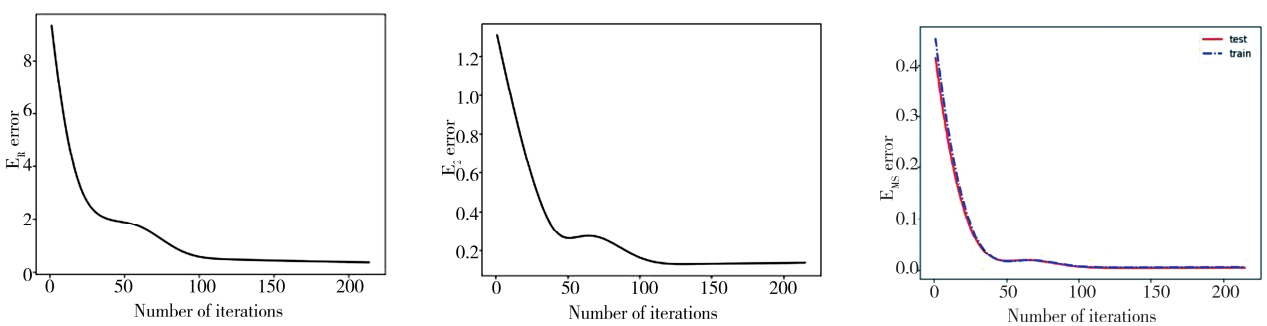

图 1　三种误差 E_R(左)，E_2(中) 和 E_{MS}(右) 随 Adam 算法迭代次数变化曲线：算例 4.1

Fig. 1　Variations of three errors E_R(left), E_2(middle) and E_{MS}(right) with iterations: Example 4.1

表 1 给出了误差 E_R，E_2 和 E_{MS} 随 Adam 算法迭代次数变化情况，图 1 给出了相应的变化曲线. 特别地，右图实线、虚线分别为测试集(见注 4)与训练集 上的 E_{MS} 误差随迭代次数变化的曲线. 可以

看出:随着迭代次数增加,近似解精度逐步提高;在迭代近 100 次后,误差趋于稳定;右图的两条误差变化曲线基本一致,表明我们的算法具有较好的泛化能力.

表 1 不同误差随迭代变化情况:算例 4.1

Tab. 1 Different errors with iterations: Example 4.1

迭代次数	初值	8	16	24	32	48	72	108	148
E_R	9.3089	5.9498	3.7843	2.6411	2.1498	1.8729	1.2820	0.4983	0.4250
E_{MS}	0.4537	0.2904	0.1708	0.0918	0.0455	0.0165	0.0173	0.0058	0.0051
$\dot{E}_{MS}$	0.4156	0.2646	0.1550	0.0833	0.0421	0.0170	0.0170	0.0050	0.0042
E_2	1.3076	1.0435	0.7986	0.5855	0.4161	0.2642	0.2642	0.1431	0.1316

图 2 至图 9 给出了近似解图像随迭代次数变化的对比,迭代次数的选取与表 1 对应,真解图像见图 11. 虽然初始误差较大,但随着迭代次数增加,近似解越来越接近真解,这表明本文的神经网络算法具有学习界面问题真解的能力.

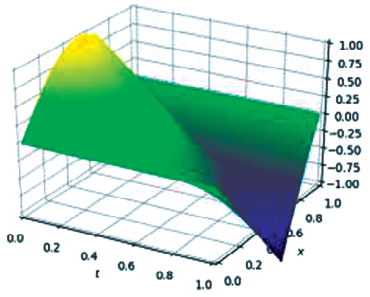

图 2 迭代初值

Fig. 2 The initial value

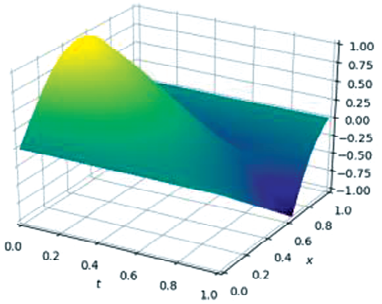

图 3 迭代 16 次

Fig. 3 16 times iteration

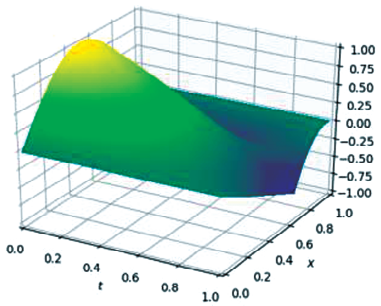

图 4 迭代 24 次

Fig. 4 24 times iteration

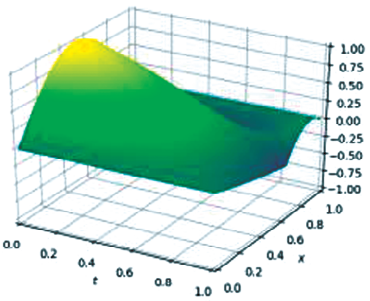

图 5 迭代 32 次

Fig. 5 32times iteration

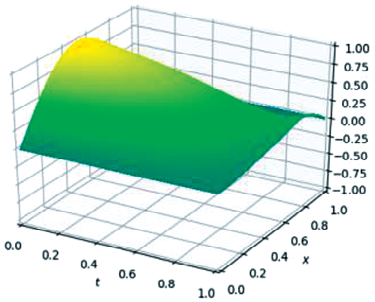

图 6 迭代 48 次

Fig. 6 48 times iteration

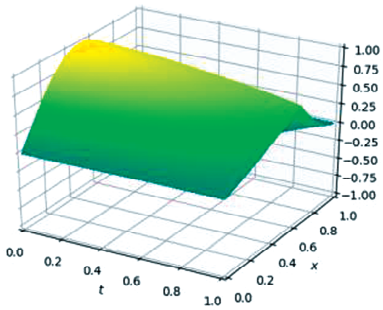

图 7 迭代 72 次

Fig. 7 48 timesiteration

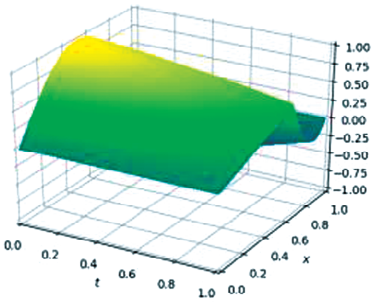

图 8 迭代 108 次

Fig. 8 108 times iteration

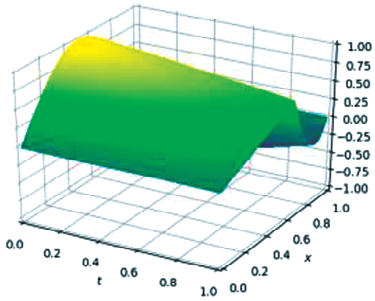

图 9 迭代 148 次

Fig. 9 148 times iteration

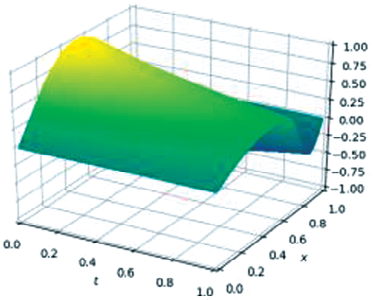

图 10 真解

Fig. 10 The exact solution

注 4 本算例中测试集选取为:

$$X_1=\{(0+\frac{1}{90}*i,0+\frac{1}{90}*j)\mid 0<i\leqslant 60,\ 0<j\leqslant 90,i,j\in \mathbf{N}^+\},$$

$$X_2=\{(\frac{2}{3}+\frac{1}{90}*i,0+\frac{1}{90}*j)\mid 0<i\leqslant 30,$$

$0<j\leqslant 90, i,j\in \mathbf{N}^+\}$.

记测试集上的 E_{MS} 误差为

$$E_{MS}=\frac{\sum\limits_{(x,t)\in X_1}(u(x,t)-U(x,t))^2+\sum\limits_{(x,t)\in X_2}(v(x,t)-V(x,t))^2}{|X_1|+|X_2|}.$$

为考察了离散优化问题(15)中参数 α 的选择对近似解精度的影响，我们分别取 $\alpha=0.1,1,10,100,1\ 000,10\ 000$. 图 11 给出了相应的 E_2 误差随迭代变化的曲线，其中因为 $\alpha=0.1,1,10$，对应的三条曲线基本重合，所以在图中只给出 $\alpha=0.1$ 的曲线作为这三种情形的代表.

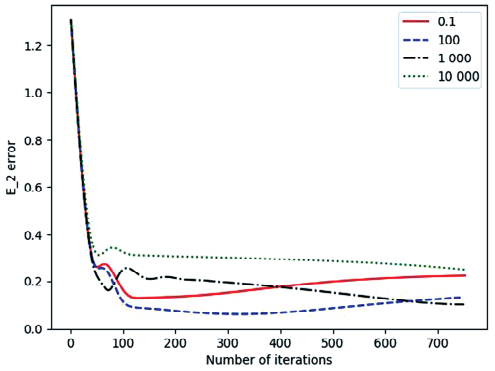

图 11　不同参数 α 对应的 E_2 误差随迭代次数的变化曲线：算例 4.1，$\alpha=0.1,100,1\ 000,10\ 000$

Fig. 11　The variation of the E_2 error corresponding to thenumber of iterations for different parameters α: Example 4.1, $\alpha=0.1,100,1\ 000,10\ 000$

从图 11 可以看出，对所有情况，随着迭代次数增加，近似解的 E_2 误差均逐步减小；到达一定迭代次数后，误差趋于稳定. 特别地，当 $\alpha=100$ 时，其对应的误差在迭代 100 次后小于其它参数选择对应的误差. 综合考虑精度与速度，$\alpha=100$ 是较优的选择.

算例 4.2　考虑二维常系数问题(6)～(13)，其中 $\Omega_1=[0,0.5)\times[0,1]$，$\Omega_2=(0.5,1]\times[0,1]$，$T=1$，热传导系数

$$a_1(x,y)=1,(x,y)\in[0,0.5]\times[0,1],$$

$$a_2(x,y)=0.5,(x,y)\in(0.5,1]\times[0,1],$$

源项

$$f_1(x,y,t)=e^{-t}((x-0.5)\sin(x+y)-2\cos(x+y)),(x,y,t)\in[0,0.5]\times[0,1]\times[0,1],$$

$$f_2(x,y,t)=-2e^{-t}\cos(x+y),(x,y,t)\in(0.5,1]\times[0,1]\times[0,1],$$

初值条件

$$u_0(x,y)=(x-0.5)\sin(x+y),(x,y)\in[0,0.5]\times[0,1],$$

$$v_0(x,y)=(2x-1)\sin(x+y),(x,y)\in(0.5,1]\times[0,1],$$

边界条件

$$u(0,y,t)=-0.5e^{-t}\sin y,y\in[0,1],\ t\in[0,1],$$

$$u(x,0,t)=e^{-t}(x-0.5)\sin(x),x\in[0,0.5],\ t\in[0,1],$$

$$u(x,1,t)=e^{-t}(x-0.5)\sin(x+1),\ x\in[0,0.5],t\in[0,1],$$

$$v(1,y,t)=e^{-t}\sin(y+1),y\in[0,1],\ t\in[0,1],$$

$$v(x,0,t)=e^{-t}(2x-1)\sin(x),x\in(0.5,1],\ t\in[0,1],$$

$$v(x,1,t)=e^{-t}(2x-1)\sin(x+1),\ x\in(0.5,1],t\in[0,1],$$

界面条件

$$\varphi(0.5,y,t)=\psi(0.5,y,t)=0,(y,t)\in[0,1]\times[0,1],$$

精确解为

$$u(x,y,t)=e^{-t}(x-0.5)\sin(x+y),\ (x,y,t)\in[0,0.5]\times[0,1]\times[0,1],$$

$$v(x,y,t)=e^{-t}(2x-1)\sin(x+y),\ (x,y,t)\in(0.5,1]\times[0,1]\times[0,1].$$

构造近似解

$$\begin{aligned}U(x,y,t)=&(1-t)((x-0.5)\sin(x+y))+\\&(1-x)(-0.5e^{-t}\sin(y)-\\&(1-t)(-0.5\sin(y)))+\\&(1-y)(e^{-t}(x-0.5)\sin(x)-\\&(1-t)((x-0.5)\sin(x)))+\\&y(e^{-t}(x-0.5)\sin(x+1)-\\&(1-t)((x-0.5)\sin(x+1))-\\&(1-x)(-0.5e^{-t}\sin(1)-\\&(1-t)(-0.5\sin(1))))+\\&xy(1-y)tN_1(x,y,t,p_1),\end{aligned}$$

$$\begin{aligned}V(x,y,t)=&(1-t)(2x-1)\sin(x+y)+\\&(2x-1)(e^{-t}\sin(1+y)-\\&(1-t)\sin(1+y))+(2-2x)U(\tfrac{1}{2},y,t)+\\&(1-y)(e^{-t}(2x-1)\sin(x)-\\&(1-t)(2x-1)\sin(x)-(2x-1)(e^{-t}\sin(1)-\\&(1-t)\sin(1))-(2-2x)U(\tfrac{1}{2},0,t))+\end{aligned}$$

$$y(e^{-t}(2x-1)\sin(x+1)-(1-t)(2x-1)\sin(x+1)-(2x-1)(e^{-t}\sin(2)-(1-t)\sin(2))-(2-2x)U(\frac{1}{2},1,t))+(2-2x)(2x-1)y(1-y)tN_2(x,y,t,p_2),$$

其中神经网络 N_1, N_2 的隐层神经元数 $M_1=M_2=10$.

由于本算例空间区域是 2 维的,若使用 Adam 算法求解优化问题(15),则需选取较多离散训练点,这将导致计算复杂度增加. 因此,我们基于随机梯度法思想提出一种针对界面问题的类随机梯度法(算法 1) . 此算法通过如下步骤更新神经网络 N_1, N_2 的权重 p_1, p_2:

1. 随机生成一点 $x=((x_0,y_0),t_0)\in\Omega\times[0,T]$,然后在该点计算损失函数关于权重 p_1, p_2 的梯度,选择适当步长,沿负梯度方向更新权重;

2. 重复步骤 1 若干次,得到新的权重;

3. 将新的权重代入近似解,计算近似解在测试集上的残差 E_R,判断是否满足停机准则:若满足,则输出优化权重;若不满足,则重复步骤 1,2,3.

在算法 1 中,每次重复步骤 1 迭代更新权重时,只代入一个随机训练点,对应的离散问题(15)中优化泛函只含一项表达式,其计算复杂度低于多训练点情形. 当然,算法 1 无法避免一般随机算法的共同缺点,即在每次迭代时可能不会按照最优的方向进行梯度下降. 但另一方面,由于随机性这类算法存在这样的可能性:跳出当前陷入的局部最优并最终达到全局最优.

算法 4.3 针对界面问题的一种类随机梯度法 (L-SGD)

输入: 随机生成的初始权重 p_1, p_2, 正实数 ε, 每轮迭代生成总点数 I, 校正比例 β, 最大迭代轮数 $N_{\max}$ 输出: 最终权重 p_1, p_2,

1: function L-SGD(p_1, p_2, ε)

2: $E_R=\varepsilon+1$, $n=0$

3: while $E_R>\varepsilon$ and $n<N_{\max}$ do

4: $i=0$

5: while $i<I$ do

6: 随机生成一点 $x=((x_0,y_0),t_0)\in\Omega\times[0,T]$

7: if $x\in\Omega_1\times[0,T]$ then

8: $g=\nabla_{p_1}Loss_U$, 确定步长 η, $p_1=p_1-\eta g$

9: else

10: if $x\in\Omega_2\times[0,T]$ then

11: $g=\nabla_{p_2}Loss_V$, 确定步长 η, $p_2=p_2-\eta g$

12: else

13: $g=\nabla_{p_1,p_2}(\alpha Loss_J)$, 确定步长 η, $p=(p_1,p_2)$, $p=p-\eta g$

14: end if

15: end if

16: $i=i+1$

17: end while

18: − − − 以下 j 循环是前面 i 循环的校正,目的是为解决 i 循环中随机生成的点落在界面上概率较低的问题− − −

19: $j=0$

20: while $j<I*\beta$ do

21: 随机生成界面上一点 $x=x=((x_0,y_0),t_0)\in\Gamma\times[0,T]$

22: $g=\nabla_{p_1,p_2}(\alpha Loss_J)$, 确定步长 η, $p=(p_1,p_2)$, $p=p-\eta g$

23: $j=j+1$

24: end while

25: − − − − − − − − − − − − − − − − − −

26: 在测试集上计算 E_R, $n=n+1$

27: end while

28: return 最终权重 p_1, p_2

29: end function.

在本算例中, 优化问题中的参数 $\alpha=1$,神经网络 N_1, N_2 的隐层神经元数 $M_1=M_2=10$,每轮迭代生成总点数 $I=500$,取定步长 $\eta=0.001$, 校正比例 $\beta=6\%$.

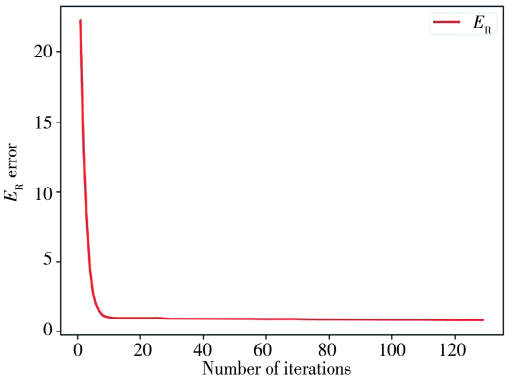

图 12 近似解在测试集上的 E_R 误差随迭代轮数变化曲线

Fig. 12 E_R error curve for the approximate solution computed on the test set

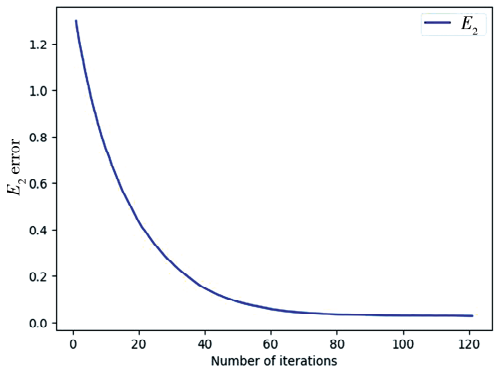

图 13　近似解的 E_2 误差随迭代轮数变化曲线

Fig. 13　E_2 error curve for the approximate solution

图 12 和图 13 分别示出了近似解在测试集上的 E_R 误差和其 E_2 误差随迭代轮数变化的曲线；表 2 给出不同迭代轮次的近似解在不同时刻的 E_2 误差. 图 12 显示，前 10 轮迭代中，E_R 误差下降较快；迭代 20 轮后，其趋于稳定. 从图 13 和表 2 可以看出，在前 60 轮迭代中，近似解的 E_2 误差下降较快；迭代 120 轮后，其趋于稳定. 上述结果表明，类随机梯度算法(算法 1)是可行的，且能给出较好的数值结果.

表 2　近似解不同迭代轮次不同时刻的 E_2 误差

Tab. 2　E_2 error of the approximate solution at different times of the iteration

t	轮　次							
	0	20	40	60	80	100	120	700
0.1	0.165 3	0.051 4	0.019 9	0.011 2	0.008 9	0.008 0	0.007 4	0.002 0
0.2	0.368 4	0.113 7	0.042 5	0.022 3	0.017 0	0.015 2	0.013 9	0.003 7
0.3	0.614 7	0.188 6	0.068 0	0.032 8	0.023 5	0.020 6	0.018 7	0.005 5
0.4	0.910 5	0.277 9	0.097 0	0.042 5	0.027 6	0.023 5	0.021 2	0.007 9
0.5	1.262 9	0.384 1	0.130 6	0.051 9	0.029 6	0.024 0	0.021 4	0.011 8
0.6	1.679 8	0.509 9	0.170 7	0.063 4	0.032 3	0.025 0	0.022 6	0.017 8
0.7	2.170 3	0.658 7	0.219 6	0.081 2	0.042 9	0.034 6	0.032 4	0.026 3
0.8	2.744 6	0.834 7	0.280 3	0.109 4	0.066 4	0.057 0	0.054 1	0.037 3
0.9	3.414 3	1.041 8	0.355 2	0.150 2	0.102 7	0.091 0	0.086 3	0.051 1
1.0	4.192 5	1.284 8	0.446 3	0.203 9	0.150 2	0.135 1	0.127 5	0.067 5

注 5　从算例 4.1 和 4.2 的数值结果可以看出，当算法迭代到一定次数后，误差趋于稳定. 我们分析其原因，主要有以下三个方面：

• 算例中神经网络隐层神经元数目较少，导致近似解与真解始终存在一定的误差(参见定理 2.1)；

• 代入离散优化问题(15)的训练点数有限，不可避免会产生一定的离散误差；

• 关于神经网络权重的离散优化问题(15) 是强非线性的，优化算法一般很难得到全局最优解.

5　结　论

本文讨论了求解热传导界面问题的一种单隐层神经网络算法. 该算法使用神经网络函数构造满足初边值条件和 Dirichlet 界面条件的近似解，通过求解由原方程导出的关于神经网络权重的离散优化问题来训练近似解中的单隐层神经网络，使近似解逼近真解. 本文也给出了一种基于随机梯度法思想的类随机梯度法来求解相应的离散优化问题. 数值算例验证了神经网络算法的有效性. 这一算法既具有近似解表达形式简洁、适用范围广等优点，又易于处理界面条件，便于推广应用于其它类型的界面问题.

参考文献：

[1] Baymani M, Kerayechian A, Effati S. Artificial neural networks approach for solving Stokes problem [J]. Appl Math, 2010, 1: 288.

[2] Cybenko G. Approximation by superpositions of a sigmoidal function [J]. Math Control Signal, 1989, 2: 303.

[3] E W, Yu B. The Deep Ritz method: A deep learning-based numerical algorithm for solving variationalproblems [J]. Commun Math Stat, 2018, 6: 1.

[4] Han J, Jentzen A, E W. Solving high-dimensional partial differential equations using deep learning [J]. P Natl Acad Sci, 2018, 115: 8505.

[5] He J, Li L, Xu J, Zheng C. Relu deep neural networks and linear finite elements [EB/OL]. arXiv: 1807.03973, 2018.

[6] He J, Xu J. MgNet: A unified framework of multi-

grid and convolutional neural network [EB/OL]. arXiv:1901.10415, 2019.

[7] Kingma D P, Ba J. Adam: A method for stochastic optimization [EB/OL]. arXiv:1412.6980, 2014.

[8] Kiefer J, Wolfowitz J. Stochastic estimation of the maximum of a regression function [J]. Ann Math Stat, 1952, 23: 462.

[9] Lagaris I E, Likas A, Fotiadis D I. Artificial neural networks for solving ordinary and partial differential equations [J]. IEEE Tr Neur Net, 1998, 9: 987.

[10] Lagaris I E, Likas A C, Papageorgiou D G. Neural-network methods for boundary value problems with irregular boundaries [J]. IEEE T Neur Net, 2000, 11: 1041.

[11] Lee H, Kang I S. Neural algorithm for solving differential equations [J]. J Comput Phys, 1990, 91: 110.

[12] Long Z, Lu Y, Ma X, *et al*. Pde-net: Learning pdes from data [EB/OL]. arXiv: 1710.09668, 2017.

[13] Meade Jr A J, Fernandez A A. The numerical solution of linear ordinary differential equations by feed-forward neural networks [J]. Math Comput Model, 1994, 19: 1.

[14] Meade J A J, Fernandez A A. Solution of nonlinear ordinary differential equations by feedforward neural networks[J]. Math Comput Mode, 1994, 20:1.

[15] Raissi M, Perdikaris P, Karniadakis G E. Machine learning of linear differential equations using Gaussian processes [J]. J Comput Phys, 2017, 348: 683.

[16] Sirignano J, Spiliopoulos K. DGM: A deep learning algorithm for solving partial differential equations [J]. J Comput Phys, 2018, 375: 1339.

[17] Yentis R, Zaghloul M E. VLSI implementation of locally connected neural network for solving partial differential equations [J]. IEEE T Circ S, 1996, 43: 687.

引用本文格式:

中 文: 王铸础, 谢小平. 热传导界面问题的一种神经网络算法[J]. 四川大学学报: 自然科学版, 2020, 57: 1038.

英 文: Wang Z C, Xie X P. A neural network algorithm for the heat conduction interface problems [J]. J Sichuan Univ: Nat Sci Ed, 2020, 57: 1038.