DOI: 10.13656/j.cnki.gxkx.2008.03.022
广西科学 Guangxi Sciences 2008, 15(3): 278~281

基于二重积分定义的神经网络求解数值积分模型及学习算法*

The Model and Algorithm for Solving Numerical Integration Based on Double Integral Definition Neural Network

蒋群华,周永权
JIANG Qun-hua, ZHOU Yong-quan

(广西民族大学数学与计算机科学学院,广西南宁 530006)
(College of Mathematics and Computer Science, Guangxi University for Nationalities, Nanning, Guangxi, 530006, China)

摘要:根据二重积分的定义和一般网络的拓扑结构特点,提出求解二重积分的神经网络模型和学习算法,并进行算法收敛性分析和算例验证.该算法收敛并且比传统的数值积分方法的计算精度高、收敛速度快.

关键词:神经网络 二重积分 数值积分 收敛性
中图法分类号:O241.4, TP183 文献标识码:A 文章编号:1005-9164(2008)03-0278-04

Abstract According to the double integral definition and the characteristic of common network topology, a network model and algorithm is proposed for the double integration in this paper, and the convergence is analyzed. Simulated result of the example indicates that the method has higher accuracy and faster convergence rate than the traditional methods for the double integration.

Key words neural network, double integral, numerical integration, convergence

二重数值积分在科学计算中起着重要作用,有关三角形单元上的数值积分有几个不同代数精度的积分公式[1].关于矩形单元上的数值积分公式,多数文献采用化重积分为累次积分,然后借助于定积分已有的数值积分计算公式(如 Simpson公式, Gauss公式等)来推导,而直接在矩形上利用二元插值函数代替被积函数推导的计算公式的不多[2,3].当然有些低次插值,如矩形单元上的双线性插值导出的数值积分公式与累次积分两次用梯形公式得到的算法是一致的.

人工神经网络的主要特点是其非线性映射能力,这种能力使得神经网络能够逼近任何给定的连续函数.从应用的角度来看,一个人工神经网络是根据某种"目标"而构造的,这样的目标一般情形下是一个多元或一元函数,称之为"目标函数",可以通过对权值的学习,使得目标函数尽可能逼近于人们理想中目标函数.

本文基于二重积分的定义和一般神经网络的拓扑结构,提出求解二重积分的神经网络模型和学习算法.首先给出二重积分定义,并说明 $\iint_D f(x,y)\,\mathrm{d}\sigma = \lim_{n,m\to\infty}\sum_{i,k=1}^{n,m} f(\xi_i,\eta_k)\Delta x\Delta y$,其中 $D=[a,b]\times[c,d]$,然后详细讨论基于二重积分定义的神经网络算法及算法的收敛性,最后给出计算实例来验证算法的有效性和正确性.该算法与传统的一些方法相比,具有计算精度高、收敛速度快等特点.

1 预备知识[4~6]

定义 1 设 $f(x,y)$ 是定义在有界可求面积的闭区域 D 上的函数,对区域 D 任作一分划

$\Delta: D_1, D_2, \cdots, D_l$,

其面积记为 $\Delta\sigma_j$, $j=1,2,\cdots,l$.令 $\lambda(\Delta)=\max_{1\le j\le l}\{D_j$ 的直径$\}$,在每一个小区域 D_j 内任取一点 (ξ_j,η_j),作和

收稿日期:2007-11-14
修回日期:2007-12-24
作者简介:蒋群华(1979-),女,硕士研究生,主要从事神经网络方面的研究工作.
* 国家自然科学基金项目(60461001),广西自然科学基金项目(0542048)和广西民族大学重大项目资助.

式 $W(\Delta,\xi_j,Z_j)=\sum_{j=1}^{l} f(\xi_j,Z_j)\Delta\sigma_j$. 如果当 $\lambda(\Delta)\to 0$ 时,和式的极限存在,且不依赖于区域 D 的分划和 D_j 中点 (ξ_j,Z_j) 的取法,那么我们称这个极限为函数 $f(x,y)$ 在区域 D 上的黎曼意义下的积分,简称 R,记

$$(R)\iint_D f(x,y)\,d\sigma=\lim_{\lambda(\Delta)\to 0} W(\Delta,\xi_j,Z_j)=\lim_{\lambda(\Delta)\to 0}\sum_{j=1}^{l} f(\xi_j,Z_j)\Delta\sigma_j.$$

在这种情形下,我们说函数 $f(x,y)$ 在区域 D 上可积,简称 R 可积.

定义 **2** 设 $f(x,y)$ 是定义在有界可求面积的闭区域 D 上的函数.作一矩形 $[a,b,c,d]$,使得 $D\subset[a,b,c,d]$.把区间 $[a,b]$ n 等分,$\Delta x_1=\Delta x_2=\cdots=\Delta x_n$.把区间 $[c,d]$ m 等分,$\Delta y_1=\Delta y_2=\cdots=\Delta y_m$,记它们的共同长度为 $\Delta x=\frac{b-a}{n}$,$\Delta y=\frac{d-c}{m}$.相应地把矩形 $[a,b,c,d]$ 分成了 mn 个小矩形,记作 R,其面积为 $\Delta x\Delta y=\frac{(b-a)(d-c)}{mn}$.在每一个 $D\cap R_{ik}$ 上任取一点 (ξ_i,Z_k)(如果 $D\cap R_{ik}=\varnothing$,取 $f(\xi_i,Z_k)=0$)作和式

$$\sum_{i,k=1}^{n,m} f(\xi_i,Z_k)\Delta x\Delta y.$$

若极限 $\lim_{n,m\to\infty}\sum_{i,k=1}^{n,m} f(\xi_i,Z_k)\Delta x\Delta y$ 存在,且不依赖于点 (ξ_i,Z_k) 在 $D\cap R_{ik}$ 上的取法,则称函数 $f(x,y)$ 在 D 上等分意义下可积,简称 R_z,记作

$$(R_z)\iint_D f(x,y)\,dx\,dy=\lim_{n,m\to\infty}\sum_{i,k=1}^{n,m} f(\xi_i,Z_k)\Delta x\Delta y.$$

引理 **1**[7,8] 函数 $f(x,y)$ 在区域 D 上 R 可积与 R_z 可积等价,并且有

$$(R)\iint_D f(x,y)\,d\sigma=(R_z)\iint_D f(x,y)\,dx\,dy.$$

2 求解数值积分的模型与学习算法

2.1 模型

确定学习训练样本数据,首先对训练样本数据作预处理,假设被积函数 $f(x,y)$ 在区域 $D=[a,b]\times[c,d]$ 上有 n 种分法 $T_1,T_2,\cdots,T_n$,其中分划 T_k 为 $X_k=\{x_0(k),x_1(k),x_2(k),\cdots,x_{m_1}(k)\}$,$Y_k=\{y_0(k),y_1(k),y_2(k),\cdots,y_{m_2}(k)\}$,$k=1,2,\cdots,n$. 令 $\Delta X=(\Delta X_1,\Delta X_2,\cdots,\Delta X_n)$,$\Delta Y=(\Delta Y_1,\Delta Y_2,\cdots,\Delta Y_n)$,其中 $\Delta X_k=(\Delta x_1(k),\Delta x_2(k),\cdots,\Delta x_{m_1}(k))$,$\Delta Y_k=(\Delta y_1(k),\Delta y_2(k),\cdots,\Delta y_{m_2}(k))$,$k=1,2,\cdots,n$,$\Delta x_i(k)=x_i(k)-x_{(i-1)}(k)$,$\Delta y_j(k)=y_j(k)-y_{(j-1)}(k)$,$i=1,2,\cdots,m_1$,$j=1,2,\cdots,m_2$,$k=1,2,\cdots,n$.则学习训练样本数据集为 $\{((\Delta X_k,\Delta Y_k),d)\mid k=1,2,\cdots,n\}$,其中 d 为积分 $\iint_D f(x,y)\,dx\,dy$ 的准确值.基于二重积分定义的神经网络求解数值积分的模型如图 1所示.

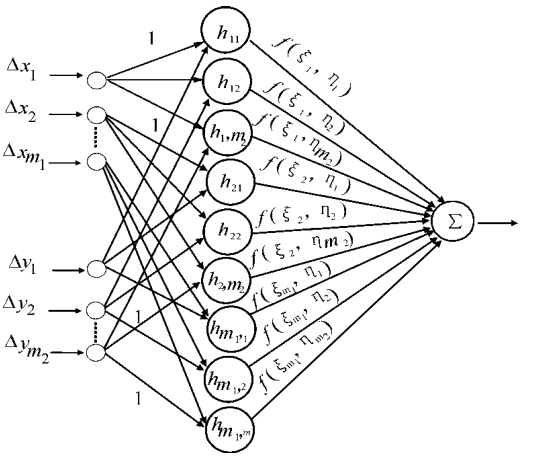

图 1 基于二重积分定义神经网络求积分模型

Fig. 1 The model for solving numerical integration based on double integral definition neural network

模型中输入层到神经元 h_{ij} 的连接权均为 1,因为连线太多,故在图 1中标出几个权值 1,其余省略.神经元 h_{ij} 取双曲函数 $h_{ij}(u,v)=uv$,$i=1,2,\cdots,m_1$;$j=1,2,\cdots,m_2$;$f(\xi_i,Z_j)$ 为中间层到输出层的权值.设该神经网络的权值矩阵为

$$W=[f(X_1,Z_1),f(X_1,Z_2),\cdots,f(X_1,Z_{m_2}),f(X_2,Z_1),f(X_2,Z_2),\cdots,f(X_2,Z_{m_2}),\cdots,f(X_{m_1},Z_1),f(X_{m_1},Z_2),\cdots,f(X_{m_1},Z_{m_2})]^{\mathrm{T}}. \quad (1)$$

输出层的激励函数为恒等函数,阀值取 0,则该网络输出

$$y(k)=\sum_{i=1}^{m_1}\sum_{j=1}^{m_2} w_{ij}\Delta x_i(k)\Delta y_j(k)=\sum_{i=1}^{m_1}\sum_{j=1}^{m_2} f(X_i,Z_j)\Delta x_i(k)\Delta y_j(k). \quad (2)$$

定义误差代价函数:

$$e(k)=d-y(k),k=1,2,\cdots,n, \quad (3)$$

其中 n 为训练样本点数,$f(x,y)$ 为被积函数.

设误差矩阵为 $E=[e(1),e(2),\cdots,e(n)]^{\mathrm{T}}$,则有性能指标:

$$J(k)=\frac{1}{2}\sum_{k=0}^{n-1} e^2(k)=\frac{1}{2}\|E\|_2^2. \quad (4)$$

在(3)式中,$\|\cdot\|_2^2$ 为 Euclidean范数的平方.根据梯度下降法学习规则,则权值调整公式为:

$$\Delta a_i=-u\frac{\partial J(k)}{\partial e(k)}\times\frac{\partial e(k)}{\partial y(k)}\times\frac{\partial y(k)}{\partial a_i}=ue(k)\Delta x_i(k)\sum_{j=1}^{m_2} f'(a,Z_j)\Delta y_j(k), \quad (5)$$

$$a(k+1)=a(k)+\Delta a(k)=a(k)+ue(k)\Delta X_k\sum_{j=1}^{m_2} f'(a,Z_j)\Delta y_j(k). \quad (6)$$

同理有

$$Z(k+1)=Z(k)+\Delta Z(k)=Z(k)+ue(k)\Delta Y_k\sum_{i=1}^{m_1}f'(a_i,Z)\Delta x_i(k), \quad (7)$$

其中 u 为学习率,且 $0<u<1$.

2.2 学习算法

神经网络学习的目标就是对权值的学习,参照一般神经网络的学习过程,神经网络求积分的学习算法的步骤如下.

步骤 1: 获取神经网络训练样本集 $\{((\Delta X_k,\Delta Y_k),d)\mid k=1,2,\cdots,n\}$; 随机产生 a,Z, 其中, $a=[a_1,a_2,\cdots,a_{m_1}]^T$, $Z=[Z_1,Z_2,\cdots,Z_{m_2}]^T$. 由 (1) 式可以得出网络权值矩阵 W;给定误差精度 X,令性能指标 $J=0$.

步骤 2 计算网络输出 $y(k)=\sum_{i=1}^{m_1}\sum_{j=1}^{m_2}w_{ij}\Delta x_i(k)\Delta y_j(k)=\sum_{i=1}^{m_1}\sum_{j=1}^{m_2}f(X,Z_j)\Delta x_i(k)\Delta y_j(k)$.

步骤 3 计算误差函数 $e(k)=d-y(k)$.

步骤 4 计算性能指标 $J=\frac{1}{2}\sum_{k=0}^{n-1}e^2(k)$.

步骤 5 计算学习率

$$u=1/[\sum_{i=1}^{m_1}|\Delta x_i(k)\sum_{j=1}^{m_2}f'(a_i,Z_j)\Delta y_j(k)|^2+\sum_{j=1}^{m_2}|\Delta y_j(k)\sum_{i=1}^{m_1}f'(a_i,Z_j)\Delta x_i(k)|^2].$$

步骤 6 权值调整 $a(k+1)=a(k)+ue(k)\Delta X_k\sum_{j=1}^{m_2}f'(a,Z_j)\Delta y_j(k)$, $Z(k+1)=Z(k)+ue(k)\Delta Y_k\sum_{i=1}^{m_1}f'(a_i,Z)\Delta x_i(k)$.

步骤 7 如果样本集未训练完,返回步骤 2,再重复上述步骤;否则,判断性能指标是否 $J>X$,如果 $J>X$,令 $J=0$,返回步骤 2,再重复上述步骤;如果 $J\leqslant X$,结束网络训练,输出网络权值 W.

3 算法的收敛性分析

给出神经网络算法的收敛性,可以为神经网络训练过程中学习率的选取提供理论依据.

定理 1 设 u为学习率,则当

$$0<u<2/(\sum_{i=1}^{m_1}|\Delta x_i\sum_{j=1}^{m_2}f'(a_i,Z_j)\Delta y_j|^2+\sum_{j=1}^{m_2}|\Delta y_j\sum_{i=1}^{m_1}f'(a_i,Z_j)\Delta x_i|^2)$$

时,神经网络算法是收敛的,其中 m_1,m_2 是分别为 $\Delta x,\Delta y$的神经元个数.

证明 取 Lyapunov函数为 $V(k)=\frac{1}{2}e^2(k)$,则

$$\Delta V(k)=\frac{1}{2}e^2(k+1)-\frac{1}{2}e^2(k), \quad (8)$$

$$e(k+1)=e(k)+\Delta e(k)=e(k)+[(\frac{\partial e(k)}{\partial a})^T\Delta a+(\frac{\partial e(k)}{\partial Z})^T\Delta Z].$$

而 $\Delta a=-u_1\frac{\partial J(k)}{\partial a}=-u_1\frac{\partial J(k)}{\partial e(k)}\frac{\partial e(k)}{\partial a}=-u_1e(k)\frac{\partial e(k)}{\partial a}$,

$$\Delta Z=-u_2\frac{\partial J(k)}{\partial Z}=-u_2\frac{\partial J(k)}{\partial e(k)}\frac{\partial e(k)}{\partial Z}=-u_2e(k)\frac{\partial e(k)}{\partial Z}.$$

所以 $\Delta e(k)=(\frac{\partial e(k)}{\partial a})^T\Delta a+(\frac{\partial e(k)}{\partial Z})^T\Delta Z=-u_1e(k)(\frac{\partial e(k)}{\partial a})^T\frac{\partial e(k)}{\partial a}-u_2e(k)(\frac{\partial e(k)}{\partial Z})^T\frac{\partial e(k)}{\partial Z}=-e(k)[u_1\|\frac{\partial e(k)}{\partial a}\|_2^2+u_2\|\frac{\partial e(k)}{\partial Z}\|_2^2]=-ue(k)[\|\frac{\partial e(k)}{\partial a}\|_2^2+\|\frac{\partial e(k)}{\partial Z}\|_2^2]$,

其中 $\|x\|_2^2=\sum_{i=1}^{n}|x_i|^2$称为 Euclidean范数的平方, $x=[x_1,x_2,\cdots,x_n]^T$. (8) 式改写为

$$\Delta V(k)=\Delta e(k)[e(k)+\frac{1}{2}\Delta e(k)]=-ue(k)[\|\frac{\partial e(k)}{\partial a}\|_2^2+\|\frac{\partial e(k)}{\partial Z}\|_2^2][e(k)-\frac{1}{2}ue(k)[\|\frac{\partial e(k)}{\partial a}\|_2^2+\|\frac{\partial e(k)}{\partial Z}\|_2^2]]=[\|\frac{\partial e(k)}{\partial a}\|_2^2+\|\frac{\partial e(k)}{\partial Z}\|_2^2]e^2(k)[-u+\frac{1}{2}u^2[\|\frac{\partial e(k)}{\partial a}\|_2^2+\|\frac{\partial e(k)}{\partial Z}\|_2^2]]. \quad (9)$$

由 (9) 式可知,要使神经网络算法收敛,必有

$$-u+\frac{1}{2}u^2[\|\frac{\partial e(k)}{\partial a}\|_2^2+\|\frac{\partial e(k)}{\partial Z}\|_2^2]<0,$$

$u>0$,

则 $0<u<\frac{2}{\|\frac{\partial e(k)}{\partial a}\|_2^2+\|\frac{\partial e(k)}{\partial Z}\|_2^2}$.

由 (2) 式和 (3) 式有

$$\frac{\partial e(k)}{\partial a}=\frac{\partial e(k)}{\partial y(x_k)}\frac{\partial y(x_k)}{\partial a}=-\Delta X_k\sum_{j=1}^{m_2}f'(a,Z_j)\Delta y_j(k),$$

$$\frac{\partial e(k)}{\partial Z}=\frac{\partial e(k)}{\partial y(x_k)}\frac{\partial y(x_k)}{\partial Z}=-\Delta Y_k\sum_{i=1}^{m_1}f'(a_i,Z)\Delta x_i(k),$$

$$\|\frac{\partial e(k)}{\partial a}\|_2^2+\|\frac{\partial e(k)}{\partial Z}\|_2^2=\|-\Delta X_k\sum_{j=1}^{m_2}f'(a,Z_j)\Delta y_j(k)\|_2^2+\|-\Delta Y_k\sum_{i=1}^{m_1}f'(a_i,Z)\Delta x_i(k)\|_2^2=\|\Delta X_k\sum_{j=1}^{m_2}f'(a,Z_j)\Delta y_j(k)\|_2^2+\|\Delta Y_k\sum_{i=1}^{m_1}f'(a_i,$$

$$\|\Delta x_i(k)\|_2^2 = \sum_{i=1}^{m_1} |\Delta x_i(k) \sum_{j=1}^{m_2} f'(a_i, Z_j) \Delta y_j(k)|^2 + \sum_{j=1}^{m_2} |\Delta y_j(k) \sum_{i=1}^{m_1} f'(a_i, Z_j) \Delta x_i(k)|^2.$$

即学习率 u 为

$$0 < u < 2 / [\sum_{i=1}^{m_1} |\Delta x_i(k) \sum_{j=1}^{m_2} f'(a_i, Z_j) \Delta y_j(k)|^2 + \sum_{j=1}^{m_2} |\Delta y_j(k) \sum_{i=1}^{m_1} f'(a_i, Z_j) \Delta x_i(k)|^2]$$

时,有 $\Delta V(k) < 0$,从而算法是收敛的.证明完毕.

4 数值积分算例

例 求积分 $\int_{1.4}^{2.0} \int_{1.0}^{1.5} \ln(x + 2y)\, dy\, dx$.

在 MATLAB语言环境下,取神经网络结构为 (40+ 50)× 2000× 1(即 $m_1 = 40, m_2 = 50$),性能指标 $J = 10^{-16}$,在区域 [1.4, 2.0]× [1.0, 1.5]上有 50 个分法,训练样本集为 $\{((\Delta X_k, \Delta Y_k), d) | k = 1, 2, \cdots, 50\}$,所得到结果为 0.429554526002843,迭代 38 次,而积分准确值为 0.42955452600,此结果与精确值相比精确到 2.843×10^{-12}.性能指标 J 与迭代次数的变化曲线见图 2.

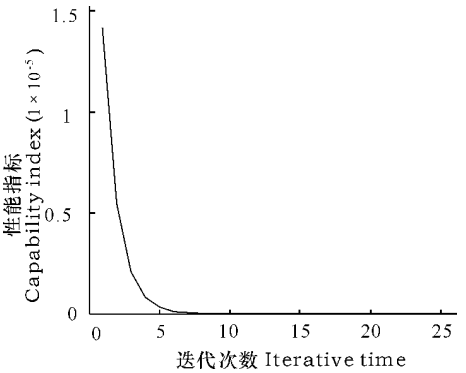

图 2 性能指标与迭代次数的变化曲线

Fig. 2 Curve of capability and iterative time

文献 [9]用复化辛卜生方法求解,所得结果为 0.4295524387,此计算值与精确值相比精确到 2.1×10^{-6}; 文献 [10]用高斯型求积公式计算,所得结果为 0.4295545313,此结果与精确值相比精确到 4.8×10^{-9}.

与文献 [9]复化辛卜生公式和文献 [10]的高斯型求积公式相比,本文提出的算法具有计算精度高、收敛速度快的特点.

参考文献:

[1] Carlet P G. The finite element method for elliptic problems [M]. North-Holland: Amsterdam, 1978.

[2] 徐利治,周蕴时.高维数值积分 [M].北京: 科学出版社, 1980.

[3] Davis P J, Rabinow P J. Method of numerical integration blaaisdell [M]. Waltham: Mass, 1975.

[4] 王能超.数值分析简明教程 [M].北京: 高等教育出版社, 1997: 66-296.

[5] 李岳生,黄友谦.数值逼近 [M].北京: 人民教育出版社, 1978: 76-204.

[6] 刘玉琏,傅沛仁.数学分析讲义: 下册 [M].北京: 高等教育出版社, 2001: 309-342.

[7] 翁显炎.也谈黎曼积分的定义 [J].丽水师范专科学校学报, 1989, 1: 25-29.

[8] 张永锋.关于重积分定义的注记 [J].安康师专学报, 1995(1): 62-66.

[9] 《现代应用数学手册》编委会.计算与数值分析卷 [M].北京: 清华大学出版社, 2001: 232-239.

[10] Curtis F Gerald (美), Patrick O Wheatley.应用数值分析 [M].吕淑娟,译.北京: 机械工业出版社, 2003: 249-253.

(责任编辑: 尹 闯)

(上接第 27页 Continue from page 277)

[6] Sumru G Altug. Complete and incomplete market models [J]. Koc University and CEPR, 2006(3): 213-303.

[7] Merton R C. Option pricing when underlying stock returns are discontinuous [J]. Journal of Financial Economics, 1976(3): 125-144.

[8] Naik V, Lee M. General equailibrium pricing of options on the market portfolio with discontinuous returns [J]. The Review of Financial Studies, 1990(3): 493-521.

[9] Kallsen Jan, Kuhn Christoph. Pricing derivatives of American and game type in incomplete markets [J]. Finance and Stochastics, 2004(8): 261-284.

[10] Mel′nikov A V. Financial markets stochastic analysis and the pricing of derivative securities [M]. New York: American Mathematical Society, 1999: 21-53.

(责任编辑: 尹 闯)