# 基函数可递归的泛函神经元网络学习算法

肖 倩[1] 周永权[2] 陈 振[1]
(广西大学计算机与电子信息学院 南宁 530004)[1] (广西民族大学信息科学与工程学院 南宁 530006)[2]

**摘 要** 将泛函神经元结构做了一个变形,给出了一种基函数可递归的泛函神经元网络学习算法,该算法借助于矩阵伪逆递归求解方法,完成对泛函神经元网络基函数的自适应调整,最终实现泛函网络结构和参数共同的最优求解。数值仿真实验结果表明,该算法具有自适应性、鲁棒性和较高的收敛精度,将在实时在线辨识中有着广泛的应用。

**关键词** 泛函神经元,基函数,矩阵伪逆,学习算法
**中图法分类号** TP183 **文献标识码** A

## Functional Network Learning Algorithm with Recursively Base Functions

XIAO Qian[1] ZHOU Yong-quan[2] CHEN Zhen[1]
(School of Computer and Electronics Information, Guangxi University, Nanning 530004, China)[1]
(College of Information Science and Engineering, Guangxi University for Nationalities, Nanning 530006, China)[2]

**Abstract** By transforming the functional neuron, we proposed a functional network learning algorithm with recursively base functions. The algorithm uses a recursive method for solving matrix's pseudo-inverse to achieve adaptive adjustment of base functions in functional neural network, finally realizes the functional network structure and parameters of the optimal solution together. The experimental results show that the learning algorithm has adaptive, robustness and high accuracy of convergence, and will have broad application in real-time online identification.

**Keywords** Functional neuron, Base functions, Matrix's pseudo inverse, Learning algorithm

## 1 引言

1998年,泛函网络(Functional Network, FNt)被西班牙学者 Enrique Castillo 提出[1,2],它是对人工神经网络(Artificial Neural Network, ANN)模型的一种全新的发展。它与ANN结构不同,泛函网络各个神经元之间的连接没有权值;并且神经元函数不固定,而是可学习的,通常是一些给定的基函数簇的线性组合。在实际应用中可据问题的特性来选择不同的基函数簇(如多项式、三角函数、Fourier 函数等);隐含层通常是由计算神经元层、中间存储单元层组成,其涵义明确;神经元之间的有向连线表示数据流动的方向,连线的多少及连接方式,对于某种目标存在着最优连接,实现能用较小的网络规模获得更满意的泛化特性。文献[5,6]通过一些典型应用实例研究表明,泛函网络的性能优于ANN,泛函网络不仅表现在ANN可以解决的问题泛函网络同样可以解决,而且对于某些ANN不能解决的问题泛函网络也可解决。因此,泛函网络已成为人工神经网络研究领域的一个新的研究方向。

目前,国内外学者关于泛函网络的研究已有10多年,其应用范围已扩展到许多领域,且表现出了卓越的性能。如时间序列预测[3]、海洋捕鱼作业[4]、微分差分方程求解[6]、混凝土强度预测[7]、故障诊断[8]、曲面重构[9,13]、非线性回归[10]、结构工程优化设计[11]、水资源预测[12]、多维空间分类[14]、仪器分析[15]、软件可靠性分析[16,30]、非线性系统辨识[17,18]、混沌通信解码[19]、多维函数逼近[20]、互联网点对点的延时动力学[21]、智能计算[22-27]、植物生长动态建模[29]等许多应用领域。对于以上应用,神经网络虽能解决,但存在着建模困难、逼近精度不高等缺陷[31]。随着泛函网络研究的不断深入,近年来其应用上取得了很大成绩,但理论与新模型及算法方面却进展不大。以下问题急待解决:一是泛函网络神经元函数选取问题。由于泛函网络神经元函数不固定,而是可学习的,是一些给定的基函数簇的组合,因此,选取什么样的基函数簇,在所选择基函数簇中哪一项是最重要的,这些确定着泛函网络的整体泛化能力。遗憾的是目前泛函网络基函数的选取没有理论指导,只能凭人为经验选取。如果网络基函数选取不当,网络的泛化能力就会受到很大影响。二是泛函网络结构学习问题。泛函网络学习目的是求出神经元函数的精确或近似表达式。学习方法可分为精确学习和近似学习两种。其中精确学习就是确定泛函网络所表示的泛函方程的解函数;近似学习是依据给定样本数据评价神经元函数。其基本方法

到稿日期:2012-03-21 返修日期:2012-06-22 本文受国家自然科学基金(61165015),广西自然科学基金(2012GXNSFDA053028),广西高等学校重大科研项目(201201ZD008)资助。

肖 倩(1989-),女,硕士生,主要研究方向为计算智能及其应用;周永权(1962-),男,博士,教授,主要研究方向为计算智能、神经网络及应用,E-mail:yongquanzhou@126.com;陈 振(1990-),男,硕士生,主要研究方向为计算智能及其应用。

是线性组合基函数簇的函数，并优化线性组合中的系数。目前采用线性和非线性两种方法，特别是非线性方法，存在着有多个最佳参数，用梯度下降法可迭代出最佳参数，但通常需要较长的迭代时间，且易陷入局部极小点，学习精度不高。

本文将泛函神经元结构做了一个变形，针对泛函网络基函数选取问题，利用矩阵伪逆递归求解法的思想，给出了一种基函数可递归的泛函神经元网络学习算法，用以完成对泛函神经元网络基函数的自适应调整，最终实现泛函网络结构和参数共同的最优求解。数值实验结果表明该算法具有较高的收敛精度。

## 2 泛函网络

一般泛函网络由以下元素组成：(1)输入单元层：这是输入数据的一层单元，输入单元以带有相应的名字的实心圆表示。(2)输出单元层：这是最后一层单元，它输出网络的结果数据。输出单元也是以带有相应的名字的实心圆表示。(3)一层或多层神经元或计算单元：每一个神经元是一个计算单元，它计算一组来自前一层神经元或输入单元的输入值，并给下一层神经元或输出单元提供一组输入数据。计算单元相互连接，每一个神经元的输出可以作为另一个神经元或输出单元数据的一部分，一旦给定输入值，输出便由神经元的类型确定，它由一函数定义，神经元由带有相应的 $f_i$ 函数名称的圆圈来表示。(4)有向连接线：它们将输入层、第一层神经元、第二层神经元、最后一层神经元及输出单元连接起来，箭头表示信息流动的方向，所有这些一起形成网络结构，它确定了整个网络的泛化能力。网络的结构是指神经元的组织及包含的连接。

一般地，$n$ 层泛函网络的模型为：

$$Y=f_n\cdot f_{n-1}\cdot\cdots\cdot f_1(X) \tag{1}$$

对应的网络结构如图 1 所示。

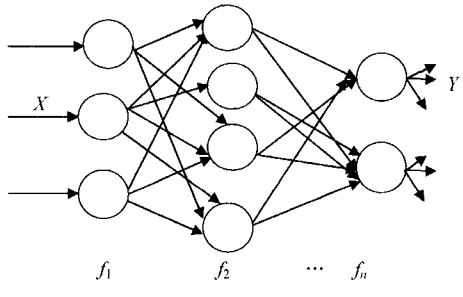


图 1 一般泛函网络的拓扑结构示意图

其中，$Y\in R^m$ 表示系统的输出，$X\in R^n$ 表示系统的输入，$f_1$，$f_2$，…，$f_n$ 分别表示不同网络层的泛函神经元函数，其接受上层函数的输入(可能是多维输入)，负责向下层函数输出数据(可能是多维输出)。

## 3 泛函神经元网络

泛函网络拓扑结构由若干个泛函神经元组成，图 2 给出了其对应的图 1 中每个泛函神经元拓扑结构图。

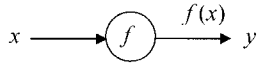


图 2 泛函神经元结构图

一般地，泛函神经元函数 $f$ 是由一些给定的基函数的线性组合，即

$$y=f(x)=\sum_{j=1}^{m}a_j\varphi_j(x) \tag{2}$$

式中，$m$ 对应基函数的个数；$a_j$ 对应泛函神经元的泛函参数，且 $a=[a_1a_2\cdots a_m]^{\mathrm{T}}$；$\varphi_j(x)$ 是一些给定的基函数，且 $\phi=[\varphi_1(x)\varphi_2(x)\cdots\varphi_m(x)]^{\mathrm{T}}$。因此，可将泛函神经元结构做一变形，变形后结构如图 3 所示。

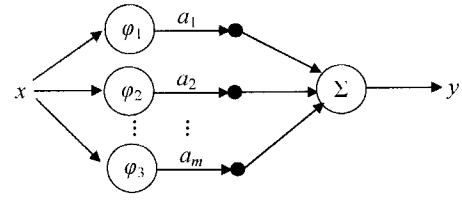


图 3 变形后的泛函神经元网络结构图

## 4 泛函神经元网络学习算法

对泛函神经元网络的学习实际上就是对该网络中神经元函数的学习，而神经元函数是一些给定基函数的线性组合，因此，对泛函网络的学习实际上就是对该组合中参数的学习，即对泛函网络参数的学习。对于有教师学习算法，事先给定学习样本数据。例如有 $p$ 个样本组：$\{(x^{(k)},y^{(k)})\mid k=1,2,\cdots,p\}$，将其代入式(2)，得到以下线性方程：

$$\begin{cases}a_1\varphi_1(x^{(1)})+a_2\varphi_2(x^{(1)})+\cdots+a_m\varphi_m(x^{(1)})=y^{(1)}\\ a_1\varphi_1(x^{(2)})+a_2\varphi_2(x^{(2)})+\cdots+a_m\varphi_m(x^{(2)})=y^{(2)}\\ \quad\vdots\\ a_1\varphi_1(x^{(p)})+a_2\varphi_2(x^{(p)})+\cdots+a_m\varphi_m(x^{(p)})=y^{(p)}\end{cases}$$

写成矩阵向量表达式形式如下：

$$[a_1a_2\cdots a_m]\begin{bmatrix}\varphi_1(x^{(1)}) & \varphi_1(x^{(2)}) & \cdots & \varphi_1(x^{(p)})\\ \varphi_2(x^{(1)}) & \varphi_2(x^{(2)}) & \cdots & \varphi_2(x^{(p)})\\ \vdots & \vdots & & \vdots\\ \varphi_m(x^{(1)}) & \varphi_m(x^{(2)}) & \cdots & \varphi_m(x^{(p)})\end{bmatrix}=\begin{bmatrix}y^{(1)}\\ y^{(2)}\\ \vdots\\ y^{(p)}\end{bmatrix}^{\mathrm{T}}$$

定义误差代价函数：

$$e_k=y^{(k)}-\sum_{j=1}^{m}a_j\varphi_j(x^{(k)}) \tag{3}$$

其误差平方和表示为：

$$E=\frac{1}{2}\sum_{k=1}^{p}e_k^{\,2} \tag{4}$$

将式(3)代入式(4)并对其求 $a_j$ 的导，可得到式(5)：

$$\frac{\partial E}{\partial a_j}=\sum_{k=1}^{p}[\varphi_j(x^{(k)})*(\sum_{j=1}^{m}a_j\varphi_j(x^{(k)})-y^{(k)})] \tag{5}$$

非线性方法是指在参数学习过程中，通过利用标准的梯度下降法来求出泛函网络中的最优参数。因此依据梯度下降法有：

$$a_j(k+1)=a_j(k)-\frac{\partial E}{\partial a_j} \tag{6}$$

由式(5)代入式(6)，并将其写成如下矩阵向量形式，即为：

$$a(k+1)=a(k)-\eta\phi^{\mathrm{T}}[\phi a(k)-\gamma] \tag{7}$$

式中，$a=[a_1 a_2\cdots a_m]^{\mathrm{T}}$；$\eta$ 代表学习效率；$k$ 代表迭代次数；$\gamma=[y^{(1)}\ y^{(2)}\cdots y^{(p)}]^{\mathrm{T}}$；

$$\phi=\begin{bmatrix}\varphi_1(x^{(1)}) & \varphi_1(x^{(2)}) & \cdots & \varphi_1(x^{(p)})\\ \varphi_2(x^{(1)}) & \varphi_2(x^{(2)}) & \cdots & \varphi_2(x^{(p)})\\ \vdots & \vdots & & \vdots\\ \varphi_m(x^{(1)}) & \varphi_m(x^{(2)}) & \cdots & \varphi_m(x^{(p)})\end{bmatrix}^{\mathrm{T}} \tag{8}$$

当 $a(k+1)=a(k)$ 时，即参数不再调整时便达到了网络的稳定状态，则有 $\phi^{\mathrm{T}}\phi a-\phi^{\mathrm{T}}\gamma=0$。由此可以得到式(9)[31]。

$$a=[\phi^{\mathrm{T}}\phi]^{-1}\phi^{\mathrm{T}}\gamma \tag{9}$$

通过以上推导过程可看出，泛函参数可由基函数的伪逆求得。当某个基函数选取不当时，需要多次调整基函数，这样将会造成内存的使用效率降低。为解决这个问题，以下给出一种递归地选取和调整基函数的方法。该方法当某个基函数选取不当时，在式(8)中，可以递归地新增一个基函数行(列)，删除一个对结果精度影响较大的基函数行(列)，这样，直到选取适合问题的解的基函数。图 4 给出其实施过程，其中 $\varphi_{m+1}$ 是新增加的基函数，$\varphi_1$ 是待删除的基函数。

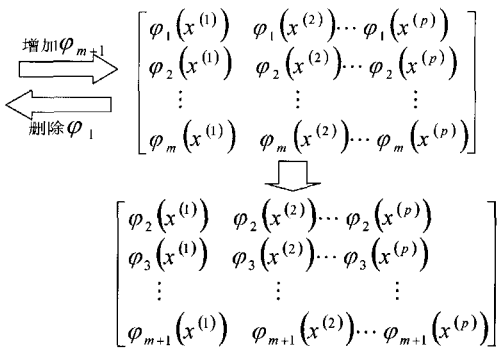


图 4　基函数递归选取过程示例

文献[28]中提出了一种矩阵伪逆递归计算算法，要求不使用求逆子程序，而使用 $W_1$ 和 $G_2$ 来计算 $G_2$ 的伪逆 $W_2$。其中，$W_1$ 表示 $G_1$ 的伪逆，$G_2$ 与 $G_1$ 包含 $(n-1)$ 个相同的行向量。该文献给出了详细的理论推导过程，将 $G_2G_2^{\mathrm{T}}$、$[G_2G_2^{\mathrm{T}}]^{-1}$ 和 $[G_1G_1^{\mathrm{T}}]^{-1}$ 表示成式(10)－式(12)[28]。

$$G_2G_2^{\mathrm{T}}=\begin{bmatrix}R^{-1} & y\\ y^{\mathrm{T}} & S^2\end{bmatrix} \tag{10}$$

$$[G_2G_2^{\mathrm{T}}]^{-1}=\begin{bmatrix}Q_1 & q_2\\ q_2^{\mathrm{T}} & q_n\end{bmatrix} \tag{11}$$

$$[G_1G_1^{\mathrm{T}}]^{-1}=\begin{bmatrix}P_1 & P_2^{\mathrm{T}}\\ P_2 & P_3\end{bmatrix} \tag{12}$$

并将矩阵伪逆递归计算算法归纳为式(13)[28]：

$$\begin{cases}R=P_3-\dfrac{P_2P_2^{\mathrm{T}}}{P_1}\\ y=D^{\mathrm{T}}b\\ S^2=b^{\mathrm{T}}b\\ q_n=\dfrac{1}{S^2-y^{\mathrm{T}}Ry}\\ q_2=-q_nRy\\ Q_1=R+\dfrac{q_2q_2^{\mathrm{T}}}{q_n}\\ W_2=[G_2G_2]^{-1}G_2\end{cases} \tag{13}$$

式中，$b$ 代表新增的一个行向量；$D$ 代表矩阵 $G_1$ 与 $G_2$ 相同的 $(n-1)$ 个行向量所组成的矩阵。该矩阵伪逆递归计算方法可以很好地实现该筛选基函数过程。

基于基函数递归泛函神经元网络学习算法的实施步骤：

Step 1　采集数据样本组 $[X,Y]$；给定基函数簇、基函数个数 $numW$ 及判定误差 $e_0$，并初始化学习次数 $k=1$；

Step 2　开始第 $k$ 次自适应学习：在基函数簇中选取 $numW$ 个基函数随机初始化基函数矩阵：$xiM$，调用矩阵伪逆递归求解方法，利用式(9)获得泛函参数 $A$，计算初始误差 $errall$；

Step 3　基函数递归增删操作：调用矩阵伪逆递归求解方法，实现选择性删除基函数(删除泛函参数的绝对值最小值所对应的基函数)、随机性增加基函数(在基函数簇中剩余的基函数中随机挑选一个增加)，并求得新的误差 $e_1$ 和泛函参数 $a_1$，比较 $e_1$ 和 $errall$ 的值，并保存较小的误差到 $errall$ 及其相对应的泛函参数 $A$；比较 $errall$ 与判定误差 $e_0$ 的大小，若大于或者等于 $e_0$，则反复执行该操作直至达到迭代次数阈值 $L_{阈}$；否则终止该操作；

Step 4　保存该次自适应学习中最小的误差值、相应的基函数矩阵及泛函参数；$k=k+1$；重新开始 Step 2，直到达到学习次数阈值 $T_{阈}$；

Step 5　输出结果。

图 5 给出了程序流程图。

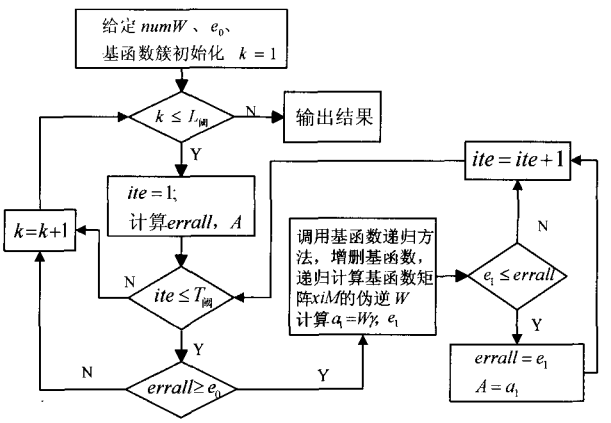


图 5　基函数递归泛函神经元网络学习算法流程图

## 5　数值实验仿真结果

为了验证本文学习算法的性能，通过 Matlab 编程，采用以下 2 个目标函数分别对泛函神经元网络进行仿真，基函数簇都选取 $\{x^0,x^1,\cdots,x^9\}$，在区间 $[-2,2]$ 内以 0.01 的采样间隔得到 401 个训练样本集。基函数个数 $n$ 的确定方法采用逐次“加 1”的方法；随机初始化基函数矩阵，通过选择性删除基函数、随机增加基函数操作来实现自适应学习 50 次，下面给出算法实验仿真结果。

在结果出现了误差相同或者相近的情况下，借助基函数复杂度来进行评判，复杂度最小的组合为最优解。所谓基函数复杂度是指基函数有效项(泛函参数中某个行向量为 0 所对应的基函数为无效项，见表 1 中的 $n=8$)的幂指数之和，用 $S$ 表示。

例 1　以 $y=x^2+\cos x$ 为例，该泛函神经元学习算法学习的结果如表 1 所列。

当 $n=5$、6、7、8、9、10 时，总误差 $errall$ 都接近 2.055370e－011。

在这种情况下，综合考虑基函数复杂度 S 的大小，确定选取 5 个基函数泛函神经元网络就能达到较高的精度。图 6 给出了 $n=5$ 时的逼近效果图。

表 1　$y=x^2+\cos x$ 实验结果

| n | 泛函参数 $a_i$ | 基函数 | S | 总误差 errall |
|---|---|---|---|---|
| 2 | 0.95031449359931<br>0.62847677652329 | $\{1,x^2\}$ | 2 | 0.36463898647933 |
| 3 | 0.99826204709741<br>0.50920079373704<br>0.03461646715329 | $\{1,x^2,x^4\}$ | 6 | 4.597114203323665e−004 |
| 4 | 0.99996797559312<br>0.50028856184423<br>0.04126782770840<br>−0.00121341978567 | $\{1,x^2,$<br>$x^4,x^6\}$ | 12 | 1.581041839511759e−007 |
| 5 | 0.99999963606829<br>0.50000499640243<br>0.04165584024909<br>−0.00138074216031<br>0.00002230025410 | $\{1,x^2,x^4,$<br>$x^6,x^8\}$ | 20 | 2.055374796712058e−011 |
| 6 | 0.99999963606811<br>0.50000499640414<br>0.04165584024716<br>−0.00138074215937<br>0.00002230025397<br>0.00000000000000 | $\{1,x^2,x^4,$<br>$x^6,x^8,x^9\}$ | 30 | 2.055374796644235e−011 |
| 7 | 0.99999963606809<br>0.00000000000000<br>0.50000499640380<br>0.04165584024746<br>−0.00138074215933<br>0.00000000000000<br>0.00002230025397 | $\{1,x,x^2,$<br>$x^4,x^6,$<br>$x^7,x^8\}$ | 28 | 2.055374796625120e−011 |
| 8 | 0.99999963606805<br>−0.00000000000000<br>0.50000499640386<br>0.04165584024727<br>0<br>−0.00138074215941<br>0.00002230025396<br>0.00000000000000 | $\{1,x,x^2,$<br>$x^4,x^5,$<br>$x^6,x^8,$<br>$x^9\}$ | 30 | 2.055374796611133e−011 |

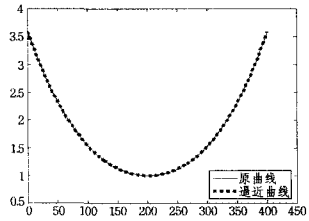


图 6　$n=5$ 时的逼近效果图

例 2　以 $y=\sin x$ 为例，该泛函神经元网络学习算法结果如表 2 所列。

当 $n=5$、6、7、8、9、10 时，总误差 *errall* 都接近 1.73369e−013。在这种情况下，综合考虑基函数复杂度 S 的大小，确定选取 5 个基函数泛函神经元网络就能达到较高的精度。图 7 给出了 $n=5$ 时的逼近效果图。

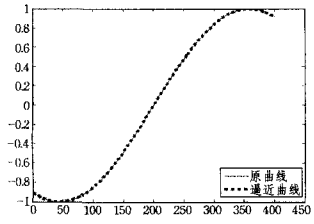


图 7　$n=5$ 时的逼近效果图

以上两例，也可以选用其它基函数，如指数基函数$\{1,e^x,e^{2x},\cdots,e^{ix},\cdots\}$、三角基函数$\{1,\sin x,\cos x,\sin 2x,\cos 2x,\cdots,\sin ix,\cos ix,\cdots\}$以及傅立叶基函数等都可实现其逼近，且可达到预定的精度。

表 2　$y=\sin x$ 实验结果

| n | 泛函参数 $a_i$ | 基函数 | S | 总误差<br>errall |
|---|---|---|---|---|
| 2 | 0.97162531780000<br>−0.13267679707253 | $\{x,x^3\}$ | 4 | 0.01568264861671 |
| 3 | 0.99904946415567<br>−0.16451390817154<br>0.00712797740939 | $\{x,x^3,x^5\}$ | 9 | 9.782908095437936e−006 |
| 4 | 0.99998287681190<br>−0.16660384596057<br>0.00827182879236<br>−0.00017615850600 | $\{x,x^3,$<br>$x^5,x^7\}$ | 16 | 2.007261585142145e−009 |
| 5 | 0.99999980807986<br>−0.16666562892703<br>0.00833177718820<br>−0.00019746539851<br>0.00000250324588 | $\{x,x^3,x^5,$<br>$x^7,x^9\}$ | 25 | 1.733687610152518e−013 |
| 6 | 0.99999980807097<br>−0.16666562889963<br>−0.00000000000000<br>0.00833177716440<br>−0.00019746539061<br>0.00000250324501 | $\{x,x^3,x^4,$<br>$x^5,x^7,x^9\}$ | 29 | 1.733687604211746e−013 |
| 7 | 0.00000000000000<br>0.99999980807096<br>−0.00000000000000<br>−0.16666562889958<br>0.00833177716440<br>−0.00019746539064<br>0.00000250324502 | $\{1,x,x^2,$<br>$x^3,x^5,$<br>$x^7,x^9\}$ | 27 | 1.733687604646505e−013 |
| 8 | 0.00000000000000<br>0.99999980807127<br>−0.16666562890058<br>0.00000000000000<br>0.00833177716473<br>−0.00019746539081<br>−0.00000000000000<br>0.00000250324504 | $\{1,x,x^3,$<br>$x^4,x^5,x^7,$<br>$x^8,x^9\}$ | 37 | 1.733687606485848e−013 |
| 9 | 0.00000000000000<br>0.99999980807102<br>−0.00000000000001<br>−0.16666562889970<br>0.00000000000001<br>0.00833177716462<br>−0.00000000000000<br>−0.00019746539074<br>0.00000250324502 | $\{1,x,x^2,$<br>$x^3,x^4,x^5,$<br>$x^6,x^7,x^9\}$ | 37 | 1.733687606328264e−013 |
| 10 | 0.00000000000011<br>0.99999980807119<br>−0.00000000000077<br>−0.16666562890023<br>0.00000000000091<br>0.00833177716500<br>−0.00000000000036<br>−0.00019746539089<br>0.00000000000004<br>0.00000250324505 | $\{1,x,x^2,$<br>$x^3,x^4,$<br>$x^5,x^6,$<br>$x^7,x^8,x^9\}$ | 45 | 1.733696437834321e−013 |

**结束语**　文中提出一种基函数可递归的泛函神经元网络学习算法，通过数值实验可看出，它在规模较小时就能达到较高的精度，逼近效果也很好。因此，基于基函数递归的泛函神经元网络学习算法可在任意给定的基函数簇得到最优解。而

传统的逼近方法先确定函数的结构形式，然后用最小二乘法确定参数，事实上这种假设在实际应用中是不合理的；而基于基函数递归的泛函神经元网络学习算法，可实现函数结构和参数共同自适应调整，且具有自适应性和鲁棒性，将在实时在线辨识中有着广泛的应用。

## 参 考 文 献


[1] Castillo E. Functional Networks [J]. Neural Processing Letters, 1998,7:151-159

[2] Castillo E, Gutierrez J M. Nonlinear Time Series Modeling and Prediction Using Functional Networks Extraction Information Masked by Chaos [J]. Physics Letters A, 1998, 244:71-84

[3] Castillo E, Cobo A, Gutierrez J M. Functional Networks with Applications [M]. Kluwer Academic Publishers, 1999

[4] Castillo E, Cobo A, Castillo C. A Minimax Method for Learning Functional Networks [J]. Neural Processing Letters, 2000, 11 (1):39-49

[5] Iglesias A, Arcay B, Cotos J M, et al. A Comparison between Functional Networks and Artificial Neural Networks for the Prediction of Fishing Catches [J]. Neural Comput & Applie, 2004,13:24-31

[6] Castillo E, Cobo A, Gutierrez J M. Working with Differential Functional and Difference Equations Using Functional Networks [J]. Appl. Math. Model, 1999, 23:89-107

[7] Rajasekaran S, Lee S-C. Prediction of Concrete Strength Using Serial Functional Network Mode [J]. Structural Engineering and Mechanics, 2003, 16(1):83-89

[8] Fontenla-Romero O, Castillo E, Alonso-Betanzos A. A Measure of Fault Tolerance for Functional Networks [J]. Neurocomputing, 2004, 62:327-347

[9] Iglesias A, Echevarr'la G, G'alvez A. Functional networks for B-spline surface reconstruction [J]. Future Generation Computer Systems, 2004, 20:1337-1353

[10] Castillo E, Hadi A S, Locruz B. Semi-Parametric Nonlinear Regression and Transformating Using Functional Networks [J]. Computational Statistics & Data Analysis, 2008, 52:2129-2157

[11] Rajasekaran S. Functional Networks in Structural Enineering. [J]. Journal of Computing in CivilEngineering, 2004, 18(2): 172-181

[12] Bruen M, Yang Jian-qing. Functional Networks in Real-Time flood Forecasting-A Novel Application [J]. Advances in Water Resources, 2005, 28(9):899-909

[13] Dossevi A, Cosmelli D, Garnero L, et al. Multivariate Reconstruction of Functional Networks From Cortical Souces Dynamics in MRI[J]. IEEE Transactions on Biomedical Engineering, 2008, 55(8):2074-2086

[14] El-Sebakhy E A, Hadi A S, Faisal K A. Iterative Least Squares Functional Networks Classifier [J]. IEEE Transactions on Neural Network, 2007, 18(3):844-850

[15] Castillo E, Sánchez-Maroño N, Alonso-Betanzos A, et al. Functional Network Topology Learning and Sensitivity Analysis Based on Anova Decomposition [J]. Neural Computation, 2007, 19(1):231-257

[16] El-Sebakhy E A. Software reliability identification using functional networks. A comparative study [J]. Expert Syst. Appl., 2009, 36(2):4013-4020

[17] 李春光，寥晓峰，何松柏，等. 非线性系统辨识的一种泛函网络方法 [J]. 系统工程与电子技术，2001，23(11)：50-53

[18] Li Chun-guang, Liao Xiao-feng, Wu Zhong-fu, et al. Complex functional networks [J]. Mathematics and Computers in Simulation, 2001, 57:355-365

[19] Li Wei-bin, Jiao Li-cheng. Functional Network and Its Application to Extract Information from Chaotic Communication[J]. Journal of Systems Engineering and Electronics, 2004, 15(1): 46-49

[20] Li Wei-bin, Liu Fang, Jiao Li-cheng, et al. Approximation Multidimension Function with Functional Network[J]. Journal of Electronics 2006, 23(1):81-84

[21] Zhu Chang-hua, Pei Chang-xing, Li Jian-dong, et al. Internet End-to-End Delay Dynamics[J]. Journal of Systems Engineering and Electronics, 2006, 17(3):685-691

[22] Zhou Yong-quan, Jiao Li-cheng. Interpolation Mechanism of Functional Networks [J]. Lecture Notes in Computer Science, 2005, 3697:45-51

[23] Zhou Yong-quan, Jiao Li-cheng. Approximate Factorization Learning Algorithm of Multivariate Polynomials Based on Functional Networks[J]. Journal of Information and Computational Science, 2005, 2(1):205-210

[24] Zhou Yong-quan, Jiao Li-cheng. One-variable interpolation Function Based on Functional Networks [J]. International Journal of Information Technology, 2006, 12(2):120-129

[25] 周永权，焦李成. 层次泛函网络整体学习算法[J]. 计算机学报，2005，28(8)：1277-1286

[26] 周永权，赵斌，焦李成. 序列泛函网络模型及其学习算法与应用[J]. 计算机学报，2008，31(7)：1073-1081

[27] 周永权，赵斌，焦李成. 一种递归泛函网络模型及学习算法[J]. 系统工程与电子技术，2007，29(10)：1727-1731

[28] 高以成. 矩阵伪逆的递归计算和程序[J]. 自动化仪表，1984，12：9-12

[29] Qu Han-bing, Hu Bao-gang. Variational Learning for Generalized Associative Functional Networks in Modeling Dynamic Process of Plant Growth[J]. Ecological Informatics, 2009, 4: 163-176

[30] El-Sebakhy E A. Functional Networks as a Novel Data Mining Paradigm in Forecasting Software Development Efforts[J]. Expert Systems with Applications, 2011, 38(3):2187-2194

[31] 周永权，赵斌. 泛函网络模型及应用研究综述[J]. 电子科技大学报：自然科学版，2010，39(6)：803-809