一种基于泛函网络的多项式 Euclidean 算法*)

周永权[1,2]　焦李成[1]　李陶深[3]

(西安电子科技大学智能信息处理研究所　西安 710071)[1]

(广西民族学院计算机与信息科学学院　南宁 530006)[2]　(广西大学计算机与电子信息学院　南宁 530003)[3]

摘　要　提出一种基于泛函网络的多项式 Euclidean 计算新模型，给出一种基于泛函网络的多项式 Euclidean 新算法。网络的泛函参数利用解线性方程组方法来完成。相对于传统方法，该方法不但能够快速地获得所求多项式问题的精确解，而且可获得所求多项式问题的近似解。计算机仿真结果表明，该算法十分有效、可行，可以看作是对传统的 Euclidean 算法的一种推广。该算法将在计算机数学、代数密码学等方面有着广泛的应用。

关键词　泛函网络，Euclidean 算法，多项式，学习算法，计算机数学

A Learning Algorithm of Euclidean Based on Functional Networks

ZHOU Yong-Quan[1,2]　JIAO Li-Cheng[1]　LI Tao-Shen[3]

(Institute of intelligence information Processing, Xidian University, Xi'an 710071)[1]

(College of Computer and Information Science, Guangxi University for Nationalities, Nanning 530006)[2]

(School of Computer and Electrical Information, Guangxi University, Nanning 530004)[3]

Abstract　In this paper, a novel polynomial functional network based on Euclidean of computation model is designed, and a learning algorithm based on Euclidean algorithm is proposed, the learning of parameters of the functional networks is carried out by the solving linear equations. Not only we obtained the exact roots of polynomial equaiton, but also the approximate roots of polynomial equation. Finally, the simulation results demonstrate that the identification method presented in the paper is more efficient and feasible in finding the roots of arbitrary polynomials. The algorithm is an extension of traditional Euclidean algorithm, to be applied to computer and algebra code fields.

Keywords　Functional network, Polynomial Euclidean algorithm, Polynomials, Learning algorithm, Compute mathematics

1　引言

在计算机数学中，寻找算法可以说是我们面对的核心问题之一。经典的多项式 Euclidean 算法，又称多项式辗转相除法，是求多项式最大公因式、求解多项式丢番图方程、编码与解码等问题常用的算法之一。Euclidean 算法几乎是多项式理论中的核心算法，同时在代数密码学领域有着广泛的应用。但传统的多项式 Euclidean 算法的计算原理是基于 Von Neumann 串行算法所设计的[1]，特别当多项式的次数很高时，辗转相除法显得效率低。且当多项式系数是浮点数时，在实施辗转相除的过程中，每一步都存在着误差累积问题，导致难以得到所求问题的精确解。

特别是近年来，由计算数学而导致的神经计算学科正得到迅猛的发展[2]，如主分量分析(PCA)、独立分量分析(ICA)、多项式求根等问题。文[5]指出，使用神经网络(ANN)来实现这些求解问题，相对于传统的方法，"神经计算"方法具有以下优点：

1)使用具有并行结构的 ANN(artificial neural network)，能够使得问题的解"一次训练，并行获取"。目前人们是在传统的 von Neumann 计算机上进行模拟的，在网络一次训练收敛到给定的满意精度后，可以同时获得问题的解，即网络权矩阵。而传统的方法一般都是一次迭代只能获得一个解，即两者计算原理不同。

2)传统的求解方法主要通过"串行-收缩"的方法。显然，如果前一步求解不太准确，必将引起下一步误差累积，必将导致求解精度下降。而 ANN 方法可并行求解，所以精度必然比传统方法高。

3)传统的数值方法必须选择合适的初值，否则算法不收敛. 而 ANN 方法与基于问题的"先验知识"结合，可以随机地选择初值，来寻找问题的最优解，所以算法不依赖于初值。

4)ANN 具有自适应、自学习的特点。如果来自问题的数据稍微发生变化，我们只须调整权值，无须从头开始，而传统的方法从头选取初值，所以"神经计算"方法具有随环境变化自适应调整权值的特点。

尽管 ANN 算法在许多情况下性能较好，但由于实际中神经网络存在结构过于复杂、结点过多，并且在学习过程中容易陷入局部极小点而导致逼近能力有限等问题，因此在一些复杂系统建模时，仍存在较大的建模误差。神经计算也不例外。

泛函网络[6]是 1998 年由 E. Castillo 提出的。与神经网络不同，它在各个神经元之间的连接没有权值；并且神经元函数不是不固定的，而是可学习的，是一个给定的基函数族的组合。我们可根据特定问题来选择不同的基函数族(如多项式、三角函数、Fourier 展开级数等)来满足不同系统问题的建模和逼近。文[7,8]中，作者通过具体实例，把泛函网络与神经网络的性能进行了比较。从仿真的结果可看出，泛函网络

*)国家自然科学基金[60461001]和广西自然科学基金[0542048]资助项目。**周永权**　博士，教授，主要研究方向为神经网络、计算智能；**焦李成**　教授，博士导师，主要研究方向为智能信息处理、进化计算等；**李陶深**　博士，教授，主要研究方向为计算机网络、CAD 等。

的性能优于神经网络。目前泛函网络已被成功地应用于许多领域，归结起来，包括非线性系统辨识、混沌时间序列预测、微分、差分泛函方程求解、计算机辅助设计、线性回归等领域[9]。相比神经网络，泛函网络在以上应用领域都表现出了良好的性能。

本文的出发点是利用泛函网络结构的特点，提出一种基于泛函网络的多项式 Euclidean 计算新模型，给出一种基于泛函网络的多项式 Euclidean 新算法，而网络的参数利用解线性方程组方法来完成。相对传统方法，该方法不但能够快速地获得所求多项式问题的精确解，而且可获得所求多项式问题的近似解。计算机仿真结果表明，该算法十分有效、可行，可以看作是对传统的 Euclidean 算法的一种推广，在计算机数学、代数密码学等方面有着广泛的应用。

2 泛函网络

2.1 泛函网络结构

一般的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名字的实心圆来表示；(2)输出单元层。这是最后一层单元，它输出网络的结果是数据。输出单元也以带有相应的名字来表示；(3)一层或多层神经元。每一个神经元是一个计算单元，它计算的是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数据。计算单元相互连接，每一个神经元的输出可作为另一个神经元或输出单元输入数据的一部分，一旦给定输入值，输出便由神经元的类型来确定，它由一函数定义。例如，假如有一个神经元具有 s 个输入 $(x_1, x_2, \cdots, x_s)$ 及 k 个函数 $f_j, j=1,2,\cdots,k$，使得 $y_j = f_j(x_1, x_2, \cdots, x_s); j=1,2,\cdots,k$。函数 f_j 由网络的结构来确定，神经元由带有相应 f_j 函数的名称用圆圈来表示；(4)有向连接线。它们将输入层、中间层、输出单元层连接起来，箭头表示信息流向，所有这些元素一起形成了泛函网络的结构，它确定了网络的泛函能力。

图 1 和图 2 分别给出了一神经网络和泛函网络结构图。

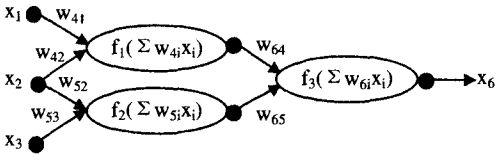

图 1 神经网络结构

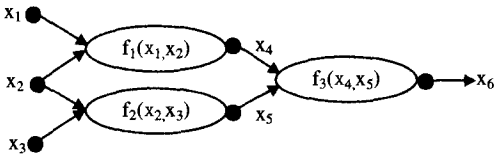

图 2 与图 1 对应的泛函网络结构

在泛函网络情形，人们通常的作法是根据问题的需要，将每一神经元函数 f_i 表示成一些已知函数簇的线性组合的形式，如多项式、三角函数等。

2.2 神经网络与泛函网络的主要区别

从上面的结构可看出，神经网络与泛函网络之间最明显的区别是[8~11]：①在神经网络中，神经元的激活函数是固定的，常用的激活函数有 Sigmoid 函数、三角波状函数、梯形函数和 RBF 等。激活函数一经选定，就不再改变。事实上，这种假定不尽合理。我们知道，在日常生活中，有的人在学习时对问题能很快理解，有的人则不然。这说明同一类型的神经元细胞对外来信号的处理功能因人而异，有的人善于适应，有的人难于适应。而泛函网络的神经元函数是用一些已知基函数簇的线性组合来表示，基函数是可训练的函数，在学习过程中能自适应调整或改变假设，是合理的。② 在神经网络中，对激活函数的连接权值进行训练，而初始权值的选定直接影响着神经网络的收敛速度。泛函网络没有连接权值，也即对激活函数进行训练。③在神经网络中，输出的单元是单个神经元的输出，而在泛函网络中，输出单元可输出一个或多个神经元，等等。

当前对神经网络的研究大都把它看作一个“黑箱”，只靠某种算法调整其权值来获得所需结果，没有充分利用已有的先验知识，盲目性较大。实际上，在解决某些问题时，要有一些先验知识作为导向。如果充分利用先验知识，在训练之前，选用和待解问题相“适配”的激活函数，则求解问题会比较容易；而泛函网络不像神经网络，它是利用问题的物理特性和先验知识来产生网络的初始结构，网络的参数的学习通过求解线性方程组而获得，因此求解问题变得容易。

3 多项式 Euclidean 算法的泛函网络计算模型

既然泛函网络作为神经网络的一种有效推广，与神经网络一样，泛函网络也有各种各样的结构。我们不可能用一个统一的通用结构来描述所有泛函网络，也不可能能用一个统一的泛函方程来表示所有的泛函网络。根据这个特性，以结合实际问题的“先验知识”作为指导，我们可设计出所求问题的泛函网络结构。在多项式计算系统中，Euclidean 算法是核心。由于 Euclidean 算法的基础是多项式带余除法，因此首先考察以下两个引理，这两个引理我们都可在一般的代数教科书多项式理论一节找到，本文不再加以证明，只加以引用。

3.1 多项式带余除法的泛函网络模型

引理 1 对任何两个多项式 $f(x)$ 与 $g(x)$，只要 $g(x)\neq 0$，总有唯一的多项式 $q(x)$ 与 $r(x)$，其中或 $r(x)=0$，或 $r(x)$ 的次数小于 $g(x)$ 的次数，使得

$$f(x)=g(x)q(x)+r(x) \tag{1}$$

这里 $q(x)$ 叫做 $f(x)$ 除以 $g(x)$ 之商，$r(x)$ 叫余式。

特别，当余式 $r(x)=0$ 时，我们称多项式 $g(x)$ 整除 $f(x)$，或者称 $g(x)$ 是 $f(x)$ 的一个倍式。结合引理 1 和泛函网络的结构特点，我们可设计出一个具有 3 个输入神经单元、1 个输出神经单元的 5 层泛函网络模型来实现多项式带余除法(图 3)。

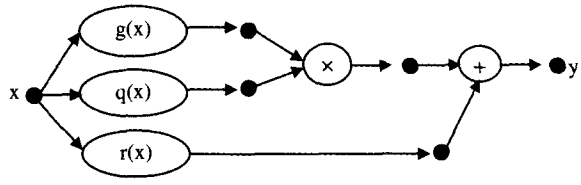

图 3 一元多项式带余除法泛函网络模型

根据图 3 网络的输出，我们可得到

$$y=g(x)q(x)+r(x) \tag{2}$$

令 $y=f(x)$，则(3)式等价于：

$$y=f(x)=g(x)q(x)+r(x) \tag{3}$$

下面，在一元多项式带余除法的泛函网络模型基础上，给出 Euclidean 算法的泛函网络计算模型。实质上，传统 Eu-

clidean 算法本身是一种精确的算法，设计出图 3 网络的目的在于给出一种 Euclidean 近似算法，使它适应于任意数域或环上运算，进一步扩展 Euclidean 算法的应用范围。

3.2 Euclidean 算法的泛函网络模型

Euclidean 算法的计算原理是基于 von Neumann 串行算法所设计的。特别当多项式的次数很高时，辗转相除法显得效率低。且当多项式系数是浮点数时，实施辗转相除的过程中，存在误差累积问题，导致难以得到精确解。因此，考虑它的泛函网络求解模型(图 4)如下：

引理 2 设 $f(x)$与 $g(x)$的次数分别为 $n\geqslant$及 $m\geqslant 1$，$d(x)=(f(x),g(x))$，则有多项式 $u(x)$及 $v()$，使 $d(x)=u(x)f(x)+v(x)g(x)$。这里 $u(x)$与 $v(x)$不是唯一的，但是可以选取 $u(x)$与 $v(x)$，使得它们的次数分别小于 m 与 n。

根据引理 2 可知，$u(x)$与 $v(x)$的选取不是唯一的。如何选取？传统的方法没有给出一个可行的算法。本文利用泛函网络作为计算模型，重点讨论这个问题。图 4 给出一种 Euclidean 算法的泛函网络计算模型，它是一个具有 4 个输入神经单元、1 个输出神经单元的 5 层泛函网络结构。

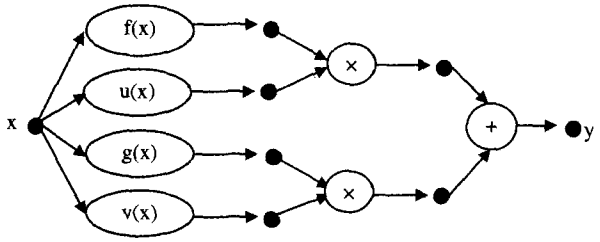

图 4 Euclidean 算法的泛函网络模型

根据图 4 的网络输出，我们有如下表达式：

$$y=f(x)u(x)+g(x)v(x) \tag{4}$$

令 $y=d(x)$，则(5)式等价于

$$y=d(x)=f(x)u(x)+g(x)v(x) \tag{5}$$

如何求出多项式 $u(x)$，$v(x)$和 $d(x)$即是所谓的多项式 Euclidean 算法。

4 基于泛函网络的多项式 Euclidean 学习算法

如何对图 4 的网络训练，由于多项式 $f(x)$，$g(x)$是事先给定的，只要任意给定样本数据，满足 $f(\cdot)\neq 0$，$g(\cdot)\neq 0$即可。因此，我们可将图 4 的泛函网络输入样本由五元组降为 3 元组的形式。于是，给定一组训练样本$\{(x_{1j},x_{2j},x_{3j})\mid j=1,2,\cdots,p\}$，其中 p 是训练模式。根据(5)式，考虑下列等式：

$$\hat{d}(x)=f(x)\hat{u}(x)+g(x)\hat{v}(x) \tag{6}$$

将样本数据代入式(6)，有

$$\hat{d}(x_{3j})=f(x_{1j})\hat{u}(x_{1j})+g(x_{2j})\hat{v}(x_{2j}) \tag{7}$$

则误差代价函数可表示为：

$$\begin{aligned} e_j &= d(x_{3j})-\hat{d}(x_{3j}) \\ &= d(x_{3j})-[f(x_{1j})]\hat{u}(x_{1j})+g(x_{2j})\hat{v}(x_{2j})] \end{aligned} \tag{8}$$

由于多项式 f,g 是给定的任意两个多项式，$d(x)=(f(x),g(x))$，$\hat{d}(\cdot)$是该网络的输出。我们通过训练可近似地获得多项式 $\hat{u}(x)$和 $\hat{v}(x)$以及 $\hat{d}(x)$。根据泛函网络的特点，对多项式来讲，我们可将(8)式中的所求多项式 $\hat{u}(x)$，$\hat{v}(x)$和 $\hat{d}(x)$分别表示一些已知多项式基函数簇线性组合的形式，即

$$\hat{u}(x)=\sum_{i=0}^{N}a_i\phi_i(x),\hat{v}(x)=\sum_{j=0}^{M}b_j\varphi_j(x),\hat{d}(x)=\sum_{k=0}^{l}c_k\gamma_k(x) \tag{9}$$

在式(10)中，a_i，b_j 和 c_l 是泛函网络的参数。根据引理 2 知，$N\leqslant m$，$M\leqslant n$，$l<\min\{n,m\}$，这样，为了找到最优化网络的参数，我们需要最小化误差平方和：

$$Q=\sum_{j=1}^{P}e_j^2=\sum_{j=1}^{P}[\sum_{k=0}^{l}c_k\gamma_k(x_{3j})-[f(x_{1j})\sum_{i=0}^{N}a_i\phi_i(x_{1j})+g(x_{2j})\sum_{j=0}^{M}b_j\lambda\varphi_j(x_{2j})]]^2 \tag{10}$$

为了保证泛函网络表达式的唯一性，需要给出一些网络的初始值。在这里，设网络的初始值为

$$\hat{u}(x_0)=\sum_{i=0}^{N}a_i\phi_i(x_0)=\alpha_0,\hat{v}(x_0)=\sum_{j=0}^{M}b_j\varphi_j(x_0)=\alpha_1 \tag{11}$$

$$\hat{d}(x_0)=\sum_{k=0}^{l}c_k\gamma_k(x_0)=\alpha_2 \tag{12}$$

其中：$\alpha_i(i=0,1,2)$是给定的一些常数。为了找到最优化网络的参数，应用 Lagrange 乘数法，考虑下列目标函数：

$$\begin{aligned} Q=\sum_{j=1}^{P}e_j^2=\sum_{j=1}^{P}[\sum_{k=0}^{l}c_k\gamma_k(x_{3j})-[f(x_{1j})\sum_{i=0}^{N}a_i\phi_i(x_{1j})+g(x_{2j})\sum_{j=0}^{M}b_j\lambda\varphi_j(x_{2j})]]^2+\mu_0(\sum_{k=0}^{l}c_k\gamma_k(x_0)-\alpha_0)+\mu_1(\sum_{i=0}^{N}a_i\phi_i(x_0)-\alpha_1)+\mu_2(\sum_{j=0}^{M}b_j\varphi_j(x_0)-\alpha_2) \end{aligned} \tag{13}$$

(13)式分别对 a_r，b_s，c_l，μ_0，μ_1，μ_2 求偏导数，得

$$\begin{cases} \dfrac{\partial Q}{\partial a_r}=-2\sum_{j=1}^{P}[\sum_{k=0}^{l}c_k\gamma_k(x_{3j}-(f(x_{1j})\sum_{i=0}^{N}a_i\phi_i(x_{1j})+g(x_{2j})\sum_{j=0}^{M}b_j\varphi_j(x_{2j}))]f(x_{1j})\phi_r(x_{1j})\phi_r(x_{rj})+\mu_0\varphi_r(x_0)=0 \\ \dfrac{\partial Q}{\partial b_s}=-2\sum_{j=1}^{P}[\sum_{k=0}^{l}c_k\gamma_k(x_{3j}-(f(x_{1j})\sum_{i=0}^{N}a_i\phi_i(x_{1j})+g(x_{2j})\sum_{j=0}^{M}b_j\varphi_j(x_{1j}))]g(x_{2j})\varphi_s(x_{sj})+\mu_1\varphi_s(x_0)=0 \\ \dfrac{\partial Q}{\partial c_t}=-2\sum_{j=1}^{P}[\sum_{k=0}^{l}c_k\gamma_k(x_{3j}-(f(x_{1j})\sum_{i=0}^{N}a_i\phi_i(x_{1j})+g(x_{2j})\sum_{j=0}^{M}b_j\varphi_j(x_{2j}))]\gamma_t(x_{tj})+\mu_2\gamma_t(x_0)=0_0) \\ \dfrac{\partial Q}{\partial \mu_0}=\sum_{i=0}^{N}a_i\phi_i(x_0)-\mu_0=0 \\ \dfrac{\partial Q}{\partial \mu_1}=\sum_{j=0}^{M}b_j\varphi_j(x_0)-\mu_1=0 \\ \dfrac{\partial Q}{\partial \mu_2}=\sum_{k=0}^{l}c_k\gamma_k(x_0)-\mu_2=0 \end{cases} \tag{14}$$

其中，$r=0,1,\cdots,N$；$s=0,1,\cdots,M$；$t=0,1,2,\cdots,l$。实际上，(14)式是一个关于参数 $a_r,b_s,c_r,\mu_0,\mu_1,\mu_2$ 的线性方程组，我们通过求解此线性方程组可获得最优化网络的参数。

5 算例与分析

为了验证本文所提出基于泛函网络的多项式 Euclidean 学习算法的有效性，考虑网络的实际输出 y_p 与期望值 $\hat{y}_p$ 之间的误差的大小，定义均方差(Root Square Error)如下：

$$RMSE=\sqrt{\frac{\sum_{p=1}^{r}\|y_p-\hat{y}_p\|^2}{r}} \tag{15}$$

其中 r 为学习样本数。

以下考虑两个具体例子，并对结果进行分析。

例 1 设 $f(x)=5x^3+5x-1$；$g(x)=x^3+2x^2+2x+3$；求多项式方程 $f(x)\hat{u}(x)+g(x)\hat{v}(x)=\hat{d}(x)$中 $\hat{u}(x)$，$\hat{v}(x)$和 $\hat{d}(x)$

为简单起见，在(14)式中，不妨设 $\mu_i=0$，$i=0,1,2,3$。在区间[0,1]上等距离产生 25 个样本数，即 r=25。我们利用图 4 的求解模型，假定该泛函网络输入都相同，即图 4 模型转化

成单输入、单输出的五层泛函网络，我们用 Maple 7.0 编程仿真，通过对神经元函数 $u(x)$，$v(x)$和 $d(x)$学习，就可得到近似解 $\hat{u}(x)$，$\hat{v}(x)$和 $\hat{d}(x)$。数值仿真部分结果如下。

(1)若置所求多项式基为：

$u\_base=\{1,x,x^2\}$，$v\_base=\{1,x,x^2,x^3\}$，$d\_base=\{1,x,x^2\}$。我们可得到 $\hat{u}(x)=-\frac{541}{661}-\frac{511}{661}x-\frac{163}{661}x^2$，$\hat{v}(x)=\frac{40}{661}+\frac{925}{661}x+\frac{815}{661}x^2$，$\hat{d}(x)=1+x+3x^2$。此时，均方和误差 RMSE=0。这样，我们就得到所求多项式的精确解。这种情况下，与传统多项式 Euclidean 算法所得结果一致。

(2)若置所求多项式基为：

$u\_base=\{1,x,x^2,x^3\}$，$v\_base=\{1,x,x^2,x^3\}$，$d\_dase=\{1,x,x^2\}$。我们可得到 $\hat{u}(x)=3b_1-1+(2b_1-\frac{591}{661})x+(2b_1-\frac{243}{661})x^2+(b_1+\frac{40}{661})x^3$，$\hat{v}(x)=b_1+(-5b_1+\frac{1125}{661})x^2+(-5b_1+\frac{200}{661})x^3$，$\hat{d}(x)=1+x+3x^2$，且均方和误差 $RMSE=0$。其中 b_1 是任意参数。这样，我们又得到所求多项式的精确解的参数形式。这种情况下，传统多项式 Euclidean 算法难以得到此精确解。

(3)若置所求多项式基为：

$u\_base=\{x,x^2,x^3\}$，$v\_base=\{1,x,x^2,x^3\}$，$d\_base=\{1,x,x^2\}$。同样我们可得到 $\hat{u}(x)=-\frac{451}{1983}x+\frac{593}{1983}x^2+\frac{541}{1983}x^3$，$\hat{v}(x)=\frac{1}{3}+\frac{70}{1983}x+\frac{815}{661}x^2-\frac{2705}{1983}x^3$，$\hat{d}(x)=1+x+3x^2$。且均方和误差 $RMSE=0$。这样，我们又得到所求多项式的另一精确解。这种情况下，传统多项式 Euclidean 算法难以得到此精确解的表达式。

(4)置所求多项式基为：$u\_base=\{1,x\}$，$v\_base=\{1,x,x^2\}$，$d_dase=\{1,x,x^2\}$我们可得到：$\hat{u}(x)=-1.181969457+0.7066423518x$，$\hat{v}(x)=-0.04743350+2.473402702x-1.259259013x^2$，$\hat{d}(x)=1+x+3x^2$。且均方和误差 $RMSE=0.00000125069264 2$。这样，得到所求多项式的一个近似解。传统多项式 *Euclidean* 算法难以得到近似解的表达式。

例 2　设

$f(x)=x^4-11x^3+39x^2-50x-24$，$g(x)=x^3-11x^2+38x-48$。求多项式方程 $f(x)\hat{u}(x)+g(x)\hat{v}(x)=\hat{d}(x)$中 $\hat{u}(x)$，$\hat{v}(x)$和 $\hat{d}(x)$。

(1)若用传统的多项式 Euclidean 算法，可求得多项式 $u(x)=-\frac{1}{44}x+\frac{9}{44}$，$v(x)=\frac{1}{44}x^2-\frac{9}{44}x+\frac{1}{44}$，$\hat{d}(x)=x-6$。

(2)若用本文提出的算法求解，为简单起见，不妨设 $\mu_i=0,i=0,1,2,3$。在区间[0,1]上等距离产生 8 个样本数，即 $r=8$。我们利用图 4 的求解模型，假定该泛函网络输入都相同，即图 4 模型转化成单输入、单输出的五层泛函网络。我们用 Maple 7.0 编程仿真，通过对神经元函数 $u(x)$，$v(x)$和 $d(x)$学习，就可得到近似解 $\hat{u}(x)$，$\hat{v}(x)$和 $\hat{d}(x)$。讨论如下：若置所求多项式基为：$u\_base=\{1,x\}$，$v\_base=\{1,x,x^2\}$，$d\_base=\{1,x\}$，我们就可得到 $\hat{u}(x)=\frac{9}{44}-\frac{1}{44}x$，$\hat{v}(x)=\frac{1}{44}-\frac{9}{44}x+\frac{1}{44}x^2$，$\hat{d}(x)=x-6$。此时，均方和误差 $RMSE=0$。我们就得到所求多项式方程的精确解。可看出与传统多项式 Euclidean 算法求得多项式 $\hat{u}(x)$，$\hat{v}(x)$和 $\hat{d}(x)$是一致的。至于其它的近似求法，可仿照例 1 的方法完成。

从上面的算例可看出，根据泛函网络的结构特点，通过给定不同的基来学习，不但可获得所求多项式方程的精确解，而且得到所求方程的近似解。传统的多项式 Euclidean 算法与本文提出的算法比较如下：

① 从计算原理来看，传统的多项式 Euclidean 算法是一种“串行”算法，而基于泛函网络的多项式 Euclidean 算法是一种“并行”算法，两者计算机制不同。

② 从上面的算例可看出，传统的多项式 Euclidean 算法只能求得一个精确解，而基于泛函网络的多项式 Euclidean 算法，不但能求得问题的精确解，而且能求得问题的近似解。

③ 基于泛函网络的多项式 Euclidean 算法，具有容错性、并行性，而传统的多项式 Euclidean 算法很难做到这一点。

④ 从算法的本质来分析可看出，传统的多项式 Euclidean 算法是一精确算法，即符号算法，而基于泛函网络的多项式 Euclidean 算法是一近似算法，即数值算法。

⑤本文提出的算法可看作是对传统的多项式 Euclidean 算法的一种推广，进一步扩充了 Euclidean 算法的应用范围。

结论　泛函网络与神经网络不同，它在各个神经元之间的连接没有权值；并且神经元函数不是不固定的，而是可学习的，是一个给定的基函数族的集合。我们可根据特定问题来选择不同的基函数族(如多项式、三角函数、Fourier 展开级数等)来满足不同系统问题的建模和逼近。本文提出了一种基于泛函网络多项式 Euclidean 计算新模型，给出了一种基于泛函网络的多项式 Euclidean 新算法，而网络的参数利用解线性方程组方法来完成。相对传统方法，本方法不但能够快速地获得所求多项式问题的精确解，而且可获得所求多项式问题的近似解。计算机仿真结果表明，本算法十分有效、可行，可以看作是对传统的 Euclidean 算法的一种推广，在计算机数学、代数密码学等方面有着广泛的应用。

参 考 文 献

1　裘仁保．计算机代数．长沙：国防科技大学出版社，1989

2　Karras D A, Palmer R G. An efficient constrained training algorithm for feedforward networks. IEEE Trans Neural Networks, 1995,6:1420～1434

3　Diamantaras K I, Kung S Y. Principal Component Neural Network: Theory and Applications. New York: Wiley, 1996

4　Huang De-Shuang. A Constructive Approach for Finding Arbitrary Roots of Polynomials by Neural Networks. IEEE Trans on Neural Networks, 2004, 15(2): 477～491

5　黄德双，池哲儒. 基于神经网络的递推分块方法求任意高阶多项式根. 中国科学(E辑)，2003,33(12):1115～1124

6　Castillo E. Functional Networks. Neural Processing, Letters, 1998,7: 151～159

7　Castillo E, Gutierrez J M. Nonlinear time series modeling and prediction using functional networks. Extracting information masked by chaos . Physics Letters, A, 1998,244:71～84

8　Castillo E, Cobo A, Gutièrrez J M. Working with differential, functional and difference equations using functional networks. Applied Mathematical Modeling, 1999, 23:89～107

9　Castillo E, Cobo A, Gutièrrez J M, et al. Functional Networks with Applications. Kluwer Academic Publishers, 1999

10　Sanchez-Marono N, Fontela-Romero O, Alonso-Betanzos A, et al. Self-Organizing Maps and Functional Networks for Local Dynamic Modeling. In: European Symposium on Artificial Networks, Bruges(Belgium), d-side public, Apral 2003. 39～44

11　Iglesias A, Arcy B, Cotos J M, et al. A comparison between functional networks and artificial neural networks for the prediction of fishing catches. Neural Comput & Applie, 2004, 13:24～31