求用 matlab 编 BP 神经网络预测程序

求一用 matlab 编的程序

```
P1=[3140 3767 4801 5288 5501 6157 6495 7061]
P2=[127 133 130 125 136 140 142 142]
T=[455 544 642 668 731 792 862 918]
%   创建一个新的前向神经网络
net_1=newff(minmax(P1),[10,1],{'tansig','purelin'},'traingdm')

%   创建一个新的前向神经网络
net_1=newff(minmax(P2),[10,1],{'tansig','purelin'},'traingdm')

%   当前输入层权值和阈值
inputWeights=net_1.IW{1,1}
inputbias=net_1.b{1}
%   当前网络层权值和阈值
layerWeights=net_1.LW{2,1}
layerbias=net_1.b{2}

%   设置训练参数
net_1.trainParam.show = 50;
net_1.trainParam.lr = 0.05;
net_1.trainParam.mc = 0.9;
net_1.trainParam.epochs = 10000;
net_1.trainParam.goal = 1e-3;

%   调用 TRAINGDM 算法训练 BP 网络
[net_1,tr]=train(net_1,P,T);

%   对 BP 网络进行仿真
A = sim(net_1,P);
%   计算仿真误差
E = T - A;
MSE=mse(E)

X1=[7505 7972
]';%测试
sim(net_1,X1)
X2=[145 146
]';%测试
sim(net_1,X2)

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

不可能啊 我 2009

对初学神经网络者的小提示

第二步：掌握如下算法:
2.最小均方误差,这个原理是下面提到的神经网络学习算法的理论核心,入门者要先看《高等数学》（高等教育出版社，同济大学版）第 8 章的第十节："最小二乘法"。
3.在第 2 步的基础上看 Hebb 学习算法、SOM 和 K-近邻算法，上述算法都是在最小均方误差基础上的改进算法，参考书籍是《神经网络原理》（机械工业出版社，Simon Haykin 著，中英文都有）、《人工神经网络与模拟进化计算》（清华大学出版社，阎平凡，张长水著）、《模式分类》（机械工业出版社，　　Richard O. Duda 等著，中英文都有）、《神经网络设计》（机械工业出版社，Martin T. Hargan 等著，中英文都有）。
4.ART(自适应谐振理论),该算法的最通俗易懂的读物就是《神经网络设计》（机械工业出版社，Martin T. Hargan 等著，中英文都有）的第 15 和 16 章。若看理论分析较费劲可直接编程实现一下 16.2.7 节的 ART1 算法小节中的算法.
4.BP 算法,初学者若对误差反传的分析过程理解吃力可先跳过理论分析和证明的内容,直接利用最后的学习规则编个小程序并测试,建议看《机器学习》(机械工业出版社，Tom M. Mitchell 著，中英文都有）的第 4 章和《神经网络设计》（机械工业出版社，Martin T. Hargan 等著，中英文都有）的第 11 章。

BP 神经网络 Matlab 实例（1）
分类：Matlab 实例

采用 Matlab 工具箱函数建立神经网络，对一些基本的神经网络参数进行了说明，深入了解参考 Matlab 帮助文档。

```
% 例 1 采用动量梯度下降算法训练 BP 网络。
% 训练样本定义如下：
% 输入矢量为
% p =[-1 -2 3 1
%      -1 1 5 -3]
% 目标矢量为    t = [-1 -1 1 1]
close all
```

```
clear
clc
% ------------------------------------------------------------
% NEWFF——生成一个新的前向神经网络,函数格式：
% net = newff(PR,[S1 S2...SNl],{TF1 TF2...TFNl},BTF,BLF,PF)
takes,
% PR -- R x 2 matrix of min and max values for R input
elements
% （对于 R 维输入，PR 是一个 R x 2 的矩阵，每一行是相应输入的
边界值）
% Si -- 第 i 层的维数
% TFi -- 第 i 层的传递函数, default = 'tansig'
% BTF -- 反向传播网络的训练函数, default = 'traingdx'
% BLF -- 反向传播网络的权值/阈值学习函数, default =
'learngdm'
% PF -- 性能函数, default = 'mse'
% ------------------------------------------------------------
% TRAIN——对 BP 神经网络进行训练，函数格式：
% train(NET,P,T,Pi,Ai,VV,TV)，输入参数：
% net -- 所建立的网络
% P -- 网络的输入
% T -- 网络的目标值, default = zeros
% Pi -- 初始输入延迟, default = zeros
% Ai -- 初始网络层延迟, default = zeros
% VV -- 验证向量的结构, default = []
% TV -- 测试向量的结构, default = []
% 返回值：
% net -- 训练之后的网络
% TR -- 训练记录(训练次数及每次训练的误差)
% Y -- 网络输出
% E -- 网络误差
% Pf -- 最终输入延迟
% Af -- 最终网络层延迟
% ------------------------------------------------------------
% SIM——对 BP 神经网络进行仿真，函数格式：
% [Y,Pf,Af,E,perf] = sim(net,P,PiAi,T)
% 参数与前同。
% ------------------------------------------------------------
%
% 定义训练样本
% P 为输入矢量
echo on
P=[-1, -2,   3,   1;
  -1,   1,   5, -3];
% T 为目标矢量
T=[-1, -1, 1, 1];
```

```
% 创建一个新的前向神经网络
net=newff(minmax(P),[3,1],{'tansig','purelin'},'traingdm')
% ------------------------------------------------------------
% 训练函数：traingdm，功能：以动量 BP 算法修正神经网络的权值和阈值。
% 它的相关特性包括：
% epochs：训练的次数，默认：100
% goal：误差性能目标值，默认：0
% lr：学习率，默认：0.01
% max_fail：确认样本进行仿真时，最大的失败次数，默认：5
% mc：动量因子，默认：0.9
% min_grad：最小梯度值，默认：1e-10
% show：显示的间隔次数，默认：25
% time：训练的最长时间，默认：inf
% ------------------------------------------------------------
% 当前输入层权值和阈值
inputWeights=net.IW{1,1}
inputbias=net.b{1}
% 当前网络层权值和阈值
layerWeights=net.LW{2,1}
layerbias=net.b{2}
% 设置网络的训练参数
net.trainParam.show = 50;
net.trainParam.lr = 0.05;
net.trainParam.mc = 0.9;
net.trainParam.epochs = 1000;
net.trainParam.goal = 1e-3;
% 调用 TRAINGDM 算法训练 BP 网络
[net,tr]=train(net,P,T);
% 对 BP 网络进行仿真
A = sim(net,P)
% 计算仿真误差
E = T - A
MSE=mse(E)
echo off
figure;
plot((1:4),T,'-*',(1:4),A,'-o')
```

1 BP神经网络的原理及算法的基本步骤

理论上已证明,一个 3层的 BP网络能够实现任意的连续映射,可以任意精度逼近任何给定的连续函数。

1.1 BP神经网络的原理

BP(Back Propagation)神经网络通常由具有多个节点的输入层(input layer)、隐含层(hidden layer)和多个或一个输出节点的输出层(output layer)组成,其学习过程分为信息的正向传播过程和

误差的反向传播过程两个阶段 。外部输入 的信号经输入层 隐含层为止 。的神经元逐层处理 ,向前传播到输 出层 ,给出结果 。如果在输出层得不到期望输出 ,则转入逆 向传播过程 ,将实际值与网络输出之间的误差沿原连接通路 返回 ,通过修改各层神经元的连接权重 ,减少误差 ,然后再转 入正向传播过程 ,反复迭代 ,直到误差小于给定的值

表 1 1981～2005 年全国总人口（单位：万人）

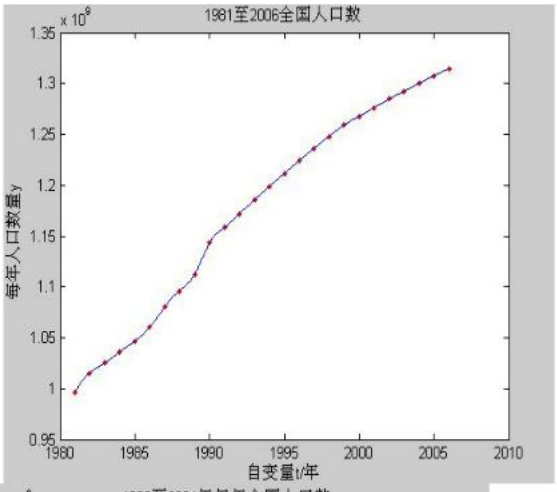
年份	1981	1982	1983	1984	1985	1986	1987	1988	1989
人口	99622	101541	102495	103604	104639	106008	108000	109614	111191
年份	1990	1991	1992	1993	1994	1995	1996	1997	1998
人口	114333	115823	117171	118517	119850	121121	122389	123626	124810
年份	1999	2000	2001	2002	2003	2004	2005		
人口	125909	缺省	127627	128453	129227	129988	130756		

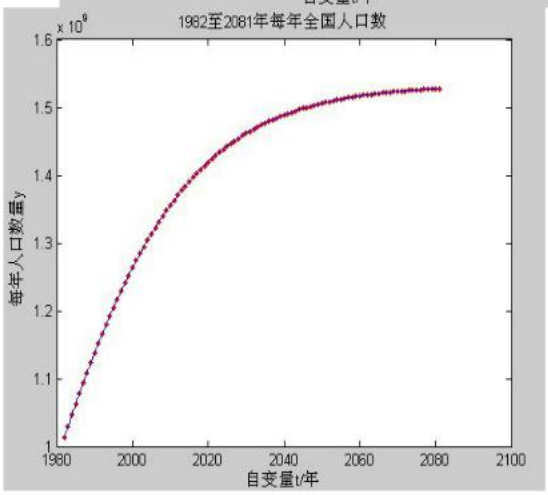
模型二预测 2001 年～2010 年全国总人口（单位:万人）

年份	2001	2002	2003	2004	2005
人口	127699	128457	129220	129987	130758
年份	2006	2007	2008	2009	2010
人口	131534	132315	133100	133890	134685

模型Ⅲ 预测的全国总人口(1981 年至 2016 年)(单位：万人)

年份	1981	1982	1983	1984	1985	1991
人口	99266	101308	102967	104600	106203	115172
年份	1992	1993	1994	1995	2001	2002
人口	116550	117895	119205	120480	127403	128436
年份	2003	2004	2005	2006	2007	2008
人口	129437	130405	131340	132244	133116	133958
年份	2009	2010	2015	2020	2025	2030
人口	134770	135552	139049	141921	144257	146144
年份	2035	2040	2045	2050	2060	2070
人口	147659	148869	149832	150596	151678	152352
年份	2080	2090	2100	2105	2110	2120
人口	152769	153026	153185	153240	153283	153344





模型四预测 2011 至 2020 年人口(单位：万人)

年份	2011	2012	2013	2014	2015
全国总人口	134668	135478	136325	137185	138036
年份	2016	2017	2018	2019	2020
全国总人口	138862	139652	140402	141106	141760

表 1　1999年 1—12月份外国人入境旅游人数

1999年	实际人数	BP神经网络预测人数
1月	529 323	528 500
2月	494 216	486 010
3月	690 393	661 960
4月	716 292	715 160
5月	724 188	722 710
6月	693 599	710 690
7月	718 341	713 530
8月	769 209	764 460
9月	769 967	770 810
10月	887 492	805 620
11月	776 649	775 250
12月	662 627	613 980

2.1　利用 Matlab Script 节点实现

在此以对一个非线性函数的逼近作为例子来说明实现流程,其中输入矢量 p=[-1：0.05：1];目标矢量 t=sin(2*pi*p)+0.1randn(size(p))。利用 Matlab Script 节点实现 BP 算法的过程如下:

(1)新建一个 LabVIEW vi,在框图程序中添加 Matlab Script 节点。

(2)在节点内添加 Matlab 的动量 BP 算法实现代码,并分别在节点左右边框分别添加对应的输入/输出参数,如图 1 所示。

(3)在 vi 的前面板添加相应的控件,设置输入参数,连接输出控件。执行程序,结果如图 2、图 3 所示。

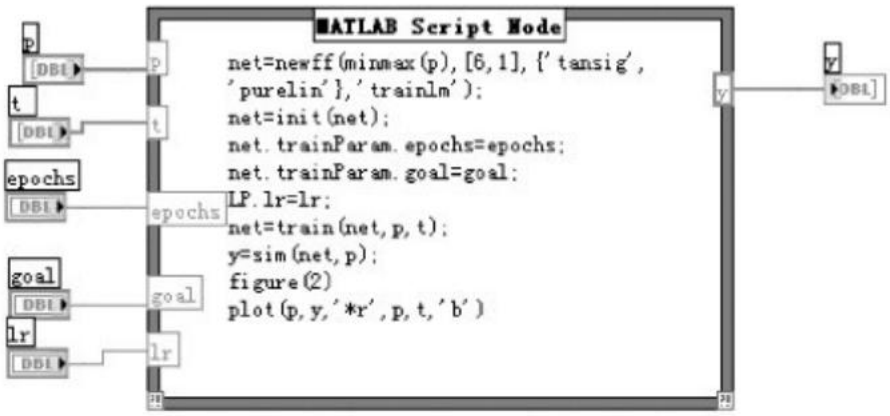


图 1　Matlab Script 实现改进的 BP 算法

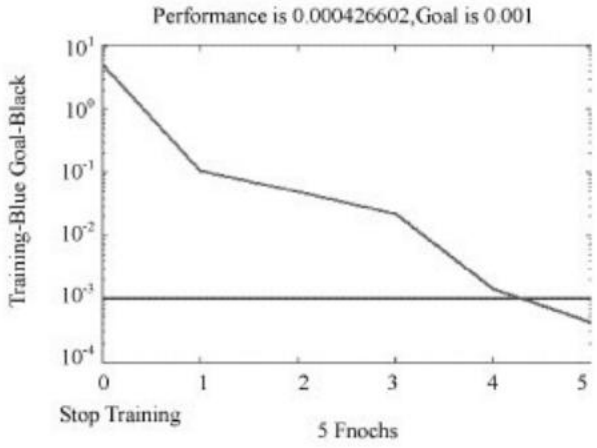


图 2　误差曲线图

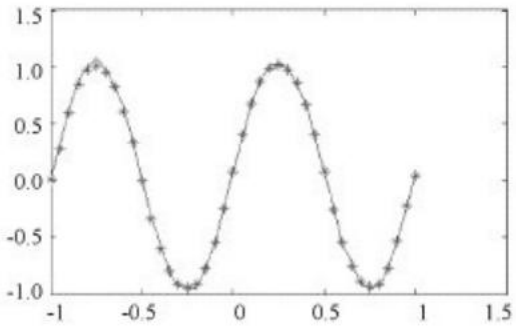


图 3　逼近曲线图

下面的代码将重建我们以前的网络，然后用批处理最速下降法训练网络。(注意用批处 理方式训练的话所有的输入要设置为矩阵方式)

```
net=newff([-1                2;                0
5],[3,1],{'tansig','purelin'},'traingd');
```

```
net.trainParam.show = 50;
net.trainParam.lr = 0.05;
net.trainParam.epochs = 300;
net.trainParam.goal = 1e-5;
p = [-1 -1 2 2;0 5 0 5]; t =
[-1 -1 1 1];
net=train(net,p,t);
TRAINGD, Epoch 0/300, MSE 1.59423/1e-05, Gradient 2.76799/
1e-10
TRAINGD, Epoch 50/300, MSE 0.00236382/1e-05, Gradient
0.0495292/1e-10
TRAINGD, Epoch 100/300, MSE 0.000435947/1e-05, Gradient
0.0161202/1e-10
TRAINGD, Epoch 150/300, MSE 8.68462e-05/1e-05, Gradient
0.00769588/1e-10
TRAINGD, Epoch 200/300, MSE 1.45042e-05/1e-05, Gradient
0.00325667/1e-10
TRAINGD, Epoch 211/300, MSE 9.64816e-06/1e-05, Gradient
0.00266775/1e-10
TRAINGD, Performance goal met.
a = sim(net,p)
a =
-1.0010 -0.9989 1.0018 0.9985
```

用nnd12sd1 来演示批处理最速下降法的性能。 带动量的批处理梯度下降法（TRAINGDM）

```
net=newff([-1 2; 0 5],[3,1],{'tansig','purelin'},'traingdm');
net.trainParam.show = 50;
```

```
net.trainParam.lr = 0.05;
net.trainParam.mc = 0.9;
net.trainParam.epochs = 300;
net.trainParam.goal = 1e-5;
p = [-1 -1 2 2;0 5 0 5]; t =
[-1 -1 1 1];
net=train(net,p,t);
TRAINGDM, Epoch 0/300, MSE 3.6913/1e-05, Gradient 4.54729/
1e-10
TRAINGDM, Epoch 50/300, MSE 0.00532188/1e-05, Gradient
0.213222/1e-10
TRAINGDM, Epoch 100/300, MSE 6.34868e-05/1e-05, Gradient
0.0409749/1e-10
TRAINGDM, Epoch 114/300, MSE 9.06235e-06/1e-05, Gradient
0.00908756/1e-10
TRAINGDM, Performance goal met.
a = sim(net,p)
a =
-1.0026 -1.0044 0.9969 0.9992
```

3.1 MATLAB神经网络工具箱的GUI（图形用户界面）工具

. 神经网络工具箱的GUI 工具主要包括：1）神经网络GUI 工具主窗口；2）神经网络的建立 窗口；3）网络训练对话框；4）自适应参数设置对话框；5）权值显示窗口. 通过神经网络工具箱的 GUI 工具按钮就能很方便地打开所建立的神经网络的结构图进行察看，也可以看到一个训练过程的 偏差曲线变化图.

3.2 神经网络工具箱解决问题的一般步骤

1）对待解决的问题进行分析，根据各种网络的特点选用合适的网络模型；2）建立网络；3） 对网络初始化；4）对网络进行训练；5）

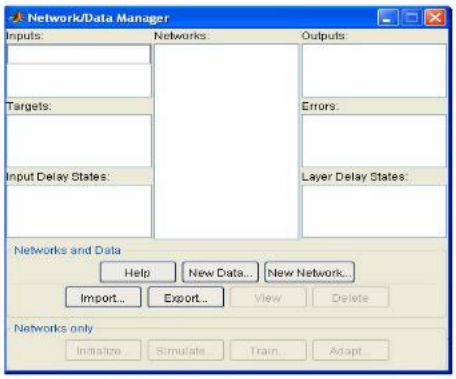
对网络进行仿真检验；6）应用网络解决问题.

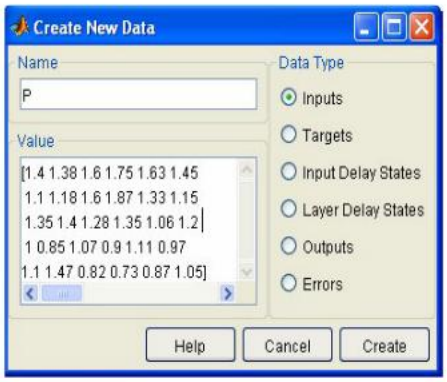
4 系统的预测仿真

4.1 使用神经网络GUI工具建立神经网络的输入样本和目标样本

1）在MATLAB 命令窗口输入nntool，打开图形用户界面工具主窗口，如图1 所示.

2）单击New Data 按钮，打开数据生成对话框. 建立输入样本P，数据的输入和设置如图2 所示，单击Create 按钮关闭对话框.





3）依照上一步，输入目标样本T，数据类型选Targets. 回到GUI 工具的主窗口，单击Export 按钮弹出导出对话框，选中变量P 和T，然后单击Export 按钮，把变量P 和T 导出到工作区. 这为仿真文件从工作空间调用导入数据做好了准备.

4.2 建立仿真模型文件进行预测

运行MATLAB 软件中的Simulink 仿真环境，在神经网络模块库中调用神经网络预测控制模块NN Predictive Controller，用模块封装技术建立河流水质数学方程模块，连接信号源模块和示波器模块后就建立了河流水质预测仿真文件predwq.mdl，如图3，其中From Workspace 模块中是目标样本T 的数据，可以直接从工作区导入.

双击神经网络预测控制模块NN Predictive Controller，弹出如图4 的窗口，此窗口用于设计模型预测控制器,输入控制器变量空间N_2 和N_u 、权值参数ρ 和控制最优化参数α 的值. 然后点击Plant Identification 按钮，打开系统辨识窗口，从工作区导入输入样本P，设置好其它参数后训练网络，单击OK 按钮，将训练好的神经网络模型导入到神经网络预测控制模块中，在NN Predictive Controller 窗口中单击OK 按钮，将控制器参数导入到NN Predictive Controller 模块中.

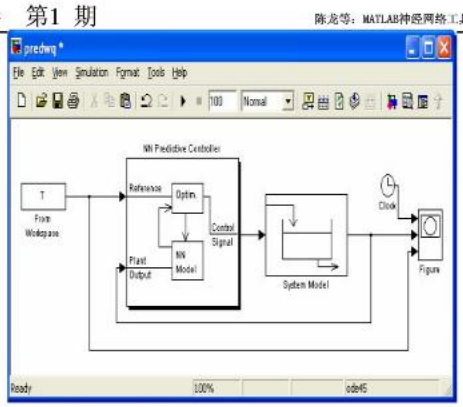


图3 predwq.mdl仿真文件

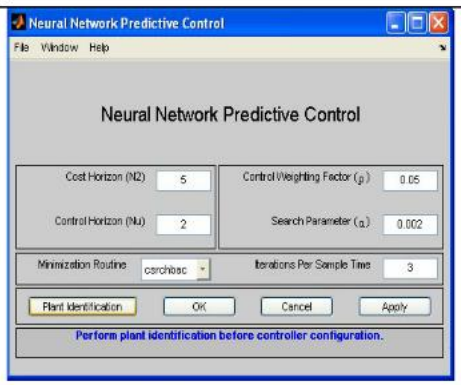


图4 神经网络预测控制模块窗口

系统模块System Model 是用模块封装技术封装的河流水质微分方程式（4）的仿真模块，如图5 所示.

在predwq Simulink 主窗口，仿真时间输入100，再单击Start simulation 命令按钮开始仿真. 仿真结束后，双击示波器模块就可以查看仿真结果，如图6 所示. 对仿真结果和目标样本的值进行比较，根据它们的差值绘出预测误差变化曲线，如图7 所示.

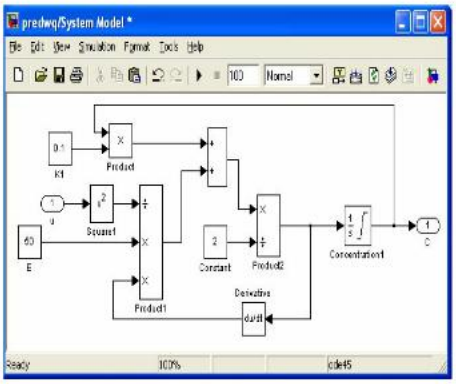


图5 河流水质微分方程仿真模块

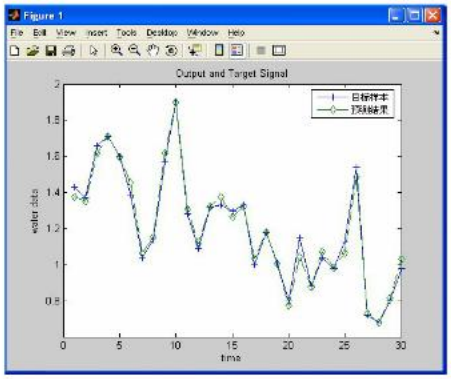


图 6 目标和预测结果图

5 结果与讨论

比较图6、图7 知，本文建立的网络系统对水质参数耗氧量的预测图像和目标样本基本一致. 把仿真预测结果数据和实际目标样本数据进行比较，正、负最大误差分别为0.11 和-0.06，最小误差为0，虽然还有偏差，但误差是在满意的范围之内. 因此，把河流水质的数学模型用MATLAB 神经网络进行仿真预测，具有较高的精度，为河流水质预测提供了方便的方法.

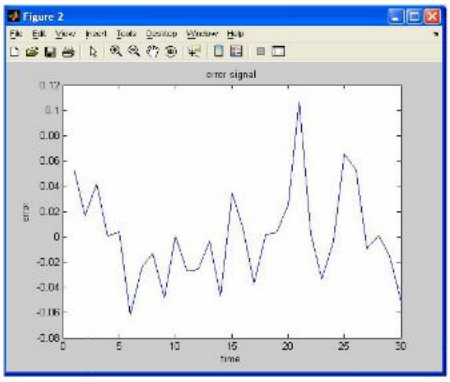


图7　预测误差的曲线变化

利用 NNToolbox 4.0.2 架构神经网络主要采用以下两种方法:

(1)使用网络数据管理器(Network Data Manager)。具体操作为:在 Launch Pad 窗体中点击 Neural Network Toolbox 目录下的 nntool 文件,即在计算机屏幕中央出现 Network Data Manager 窗体;点击窗体中 NewNetwork 按钮,根据提示设定网络类型、网络结构、网络算法和网络激活函数即可生成用户定义的神经网络;点击 View 按钮可显示该神经网络的结构图;点击 initialize、simulate、train 和 adapt 按钮并设定参数可对神经网络进行初始化、模拟、训练和仿真;最后点击 Export 按钮可将网络模拟、训练和仿真的结果以文件的形式导出。该方法操作简易,无需编写程序代码,即可完成神经网络的构建、初始化、训练和仿真等主要工作。但是该方法不能和 MATLAB 其他程序动态链接,网络仿真结果只能以数据文件的形式导出,不能可视化显示。

(2)编写 MATLAB 应用程序,即运用 MATLAB 语言引用神经网络工具箱函数编写程序代码并保存为 M 文件,然后运行该文件。该种方法可以根据研究人员的需要,调用 MATLAB 丰富的内部函数,并能和各类数据库及其他应用程序(包括 C、FORTRAN 程序)动态链接,使神经网络功能更为强大。本研究即采用该种方法,其中引用的重要算法、函数及参数将在第 2、3 节详述。

中国未来几年人口总量的预测

本文对所使用的序列数据进行GM(1,1)模型维数优化时 得到最佳的维数是5~8 维不等均属于短序列预测只适合短 期的人口总量的预测所以对

2008-2012 年共5 个时间序列点 的中国人口总量进行预测结果如表2 所示。

表2 灰色人工神经网络GANN模型对2008-2012 年 中国人口总量预测的结果 万人

年份	2008	2009	2010	2011	2012
总人口	132 663	132 963	133 706	134 373	135 025

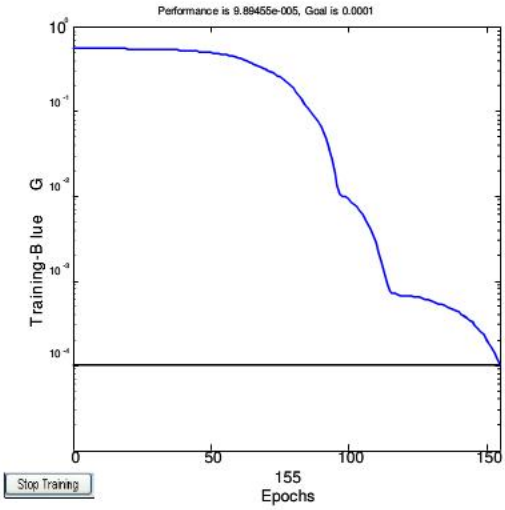
建立 BP 神经网络预测模型
在进行 BP 网络预测模型设计时,主要考虑网络的层数和每层中神经元的个数。

神经网络通过计算机程序实现对非线性映射逼近,在众多语言中,MATLAB 语言允许数学形式的语言编写程序,比其他语言更接近我们书写计算公式的思维方式。因此编程效率高,易学易懂。

年份	人口 总 数（万人）
1985	105851
1986	107507
1987	109300
1988	111026
1989	112704
1990	114333

1991	115823
1992	117171
1993	118517
1994	119850
1995	121121
1996	122389
1997	123626
1998	124761
1999	125786
2000	126743
2001	127627
2002	128453
2003	129227
2004	129988
2005	130756

采用单隐层的BP 网络进行预测[3]。建立两层神经网络，由于输入样本为 3 维的输入向量，因此，输入层一共有3 个神经元，根据Kolmogorov[2]定理，网络应该为3×7×3 的结构。 其中，threshold 设定了网络输入向量的取值范围[0,1]， 第一层神经元数为 3,传函类型为‘LOGSIG’,第二层神经元数为 7,传函类型为‘PURELIN’,采用动量梯度下降反向传播算法对 网络进行训练。中间层的神经元个数是很难确定的，而这又在很大程度上影响着网络的预测 性能。当网络的预测误差最小时，网络中间层的神经元数目就是最佳值。进行对比后，可见 中间层神经元个数为 8 时，网络的预测性能最好。训练结果如图 1 所示，网络的预报误差如



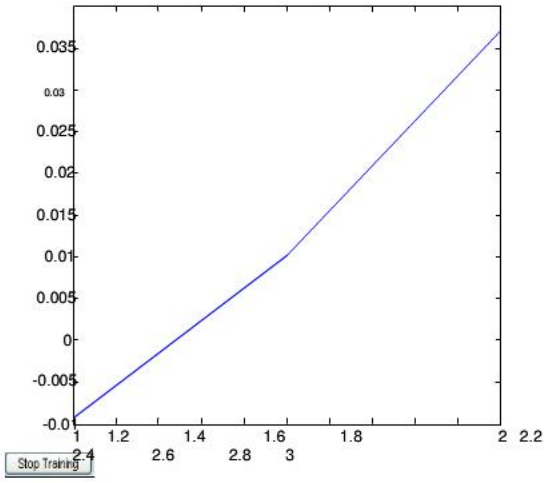


图1　训练结果（中间层神经元数目：8）

年份	总人口数（万人）
2006	140800
2007	142500
2008	146600
2009	151000

图2　网络的预报误差（中间层神经元数目：8）

2010	155800
2011	159600
2012	162100
2013	163300
2014	163800
2015	164000
2016	164100
2017	164100
2018	164100
2019	164100
2020	164100

，每行语句前都有一个符号“>>”，此即命令提示符。在此符号后(也只能 在此符号后)输入各种语句并按Enter 键，方可被MATLAB 接收和执行。执行的结果通常 就直接显示在语句下方，如图1.2 所示。

不同类型语句用不同颜色区分。在默认情况下，输入的命令、函数、表达式以及计算 结果等采用黑色字体，字符串采用赭红色，if、for 等关键词采用蓝色，注释语句用绿色。

因为MATLAB 是一个多功能集成软件，不同的功能需要使用不同的文件格式去表现，所以MATLAB 的文件也有多种格式。最基本的是M 文件、数据文件和图形文件，除此之 外，还有MEX 文件、模型文件和仿真文件等。下面分别予以说明。

(1) M 文件，以.m 为扩展名，所以称为M 文件。M 文件是由一系列MATLAB 语句组 成的文件，包括命令文件和函数文件两类，命令文件类似于其他高级语言中的主程序或主 函数，而函数文件则类似于子程序或被调函数。

MATLAB 众多工具箱中的(函数)文件基本上是M 函数文件因为它们是由ASCII 码表示的文件，所以可由任一文字处理软件编辑后以文本格式存放。

(2) 数据文件，以.mat 为扩展名，所以又称MAT 文件。在讨论工作空间窗口时已经涉及到MAT 文件。显然，数据文件保存了MATLAB 工作空间窗口中变量的数据。

(3) 图形文件，以.fig 为扩展名。主要由MATLAB 的绘图命令产生，当然也可用File 菜单中的New 命令建立。

(4) MEX 文件，以.mex 或.dll 为扩展名，所以称MEX 文件。MEX 实际是由MATLAB Executable 缩写而成的，由此可见，MEX 文件是MATLAB 的可执行文件。

(5) 模型和仿真文件，模型文件以.mdl 为扩展名，由 Simulink 仿真工具箱在建立各种 仿真模型时产生。仿真文件以.s 为扩展名。

键盘输入语句(input)

其调用格式有

(1) x = input('prompt')：显示提示字符串'prompt'，要求用户键盘输入x 的值。(2) x = input('prompt','s')：显示提示字符串'prompt'，要求用户键盘输入字符型变量 x 的 值，不至于将输入的数字看成是数值型数据。

6.3.2 屏幕输出语句(disp)

屏幕输出最简单的方法是直接写出欲输出的变量或数组名，后面不加分号。此外，可 以采用disp 语句，其调用格式为disp(x)。

6.3.3 M 数据文件的存储/加载(save / load)

1. save 语句

其调用格式有

(1) save：将所有工作空间变量存储在名为MATLAB.mat 的文件中。

(2) save filename：将所有工作空间变量存储在名为filename 的文件中。

(3) save filename X Y Z:将工作空间的指定变量X、Y、Z 存于名为filename 的文件中。

2. load 语句

其调用格式有

(1) load：如果MATLAB.mat 文件存在，则加载MATLAB.mat 文件中存储的所有变量到工作空间；否则返回一错误信息。

(2) load filename：如果filename 文件存在，则加载filename 文件中存储的所有变量到工作空间；否则返回一错误信息。

(3) load filename X Y Z如果filename 文件及存储的变量XYZ 存在则加载filename 文件中存储的变量X、Y、Z 到工作空间；否则返回一错误信息。

在MATLAB 中，主要的二维绘图函数如下：

(1) plot：x 轴和y 轴均为线性刻度。

(2) loglog：x 轴和y 轴均为对数刻度。

(3) semilogx：x 轴为对数刻度，y 轴为线性刻度。

(4) semilogy：x 轴为线性刻度，y 轴为对数刻度。

(5) plotyy: 绘制双纵坐标图形。

其中plot 是最基本的二维绘图函数，其调用格式有

① plot(Y)：若Y 为实向量，则以该向量元素的下标为横坐标，以Y 的各元素值为纵 坐标，绘制二维曲线；若Y 为复数向量，则等效于plot(real(Y),imag(Y))；若Y 为实矩阵， 则按列绘制每列元素值相对其下标的二维曲线，曲线的条数等于Y 的列数；若Y 为复数矩 阵，则按列分别以元素实部和虚部为横、纵坐标绘制多条二维曲线。

② plot(X,Y)：若X、Y 为长度相等的向量，则绘制以X 和Y 为横、纵坐标的二维曲线；若X 为向量，Y 是有一维与Y 同维的矩阵，则以X 为横坐标绘制出多条不同色彩的曲线，曲线的条数与Y 的另一维相同；若X、Y 为同维矩阵，则绘制以X 和Y 对应的列元素 为横、纵坐标的多条二维曲线，曲线的条数与矩阵的列数相同。

③ plot(X1,Y1,X2,Y2,…Xn,Yn)其中的每一对参数 Xi 和 Yi(i=1,2,...,n)的取值和所绘图

形与②中相同。

④ plot(X1,Y1,LineSpec,...)：以LineSpec 指定的属性，绘制所有Xn、Yn 对应的曲线。

⑤ plot(...,'PropertyName',PropertyValue,...)：对于由plot 绘制的所有曲线，按照设置的属性值进行绘制，PropertyName 为属性名，PropertyValue 为对应的属性值。

⑥ h = plot(...)：调用函数plot 时，同时返回每条曲线的图形句柄h(列向量)。

① 在工作空间查看各个变量，或在命令窗口用who, whos(注意大小写)查看各个 变量。

② 在工作空间双击变量，弹出Array Editor 窗口(数组编辑器窗口)，即可修改变量。

③ 使用save 命令把工作空间的全部变量保存为my_var.mat 文件。

```
>>save my_var.mat
```

④ 输入下列命令：

```
>>clear all %清除工作空间的所有变量
```

观察工作空间的变量是否被清空。使用load 命令把刚才保存的变量载入工作空间。

```
>>load my_var.mat
```

⑤ 清除命令窗口命令：

```
>>clc
```