有时候钻进去就是峰回路转的，偶尔也出来透透气吧，今天玩这个真心很累了，let's have a rest~

    随着计算机的普及，现代算法的发展已经不再是那么简单的循环套弄下的了。最近研究了很多回根据现实生活中的一些现象得出的算法，如神经网络算法。人脑大约有1000多亿个神经元，到现在人们对他的了解也是很有限的。神经网络算法便是从结构复杂的人脑抽象出来的，具有信息并行处理能力、自学能力和推理能力，很神奇吧？好吧，扯扯，具体的算法是可以自己设计的，不过比较麻烦，这里就不写了，直接给出matlab实现的方法。

    目前应用的最为广泛的两种网络是前向反馈（BP）和径向基（RBP）网络，这两个网络的研究都比较成熟了，其中BP神经网络的应用频率比较高，这里就先介绍这个了，其他的以后有时间再说，最近比较忙。。。

    神经网络的理论比较复杂，有兴趣自己去搜搜，一堆一堆的数学表达式，推了我半天。直接说正题吧，matlab实现的方法，MathWorks公司开发了神经网络工具箱，一般不需要了解太深的可以直接拿来用。

    matlab神经网络工具箱主要用于BP网络分析与设计的函数，这面这些函数最好都弄懂：

    1、前向网络创建函数：newcf、newff和newfftd

    2、激励函数：logsig、dlogsig、(S型对数式)tansig、dtansig、(正切函数式)purelin、dpurelin(线性函数)

    3、学习函数：learngd(梯度下降权值/阀值学习函数)、learndm(梯度下降动量函数)

    4、训练函数：trainbfg(BFGS准牛顿BP算法)、traingd(梯度下降BP算法训练函数)、traingdm(梯度下降动量BP算法)等等

    5、性能函数：mse(均方误差性能函数)、msereg

     函数学习这一块在学习BP神经算法的时候除了参考帮助文档(很有用)，也可以参考下面的代码自己改改试试，一般运行时间长短看你的参数设置决定，有时几秒钟，有时，可能就几小时了。

    BP网络求解过程，具体代码里也都注释了，理清思绪：

    1、原始数据的输入

    2、数据的归一化处理

    3、网络训练

    4、对原始数据进行仿真

    5、将原始数据仿真的结果与已知的样本进行对比

    6、对新的数据进行仿真

    代码具体是参考的《matlab在数学建模中的应用》(p117)里的一道题目编写的，由于时间有限，可以看看运行结果大概想想题目是什么，这里就不写了。

    代码：

%%原始数据的输入

clc,clear

sqrs = [20.55 22.44 25.37 27.13 29.45 30.1 30.96 34.06 36.42 38.09 39.13 39.99 ...

    41.93 44.59 47.30 52.89 55.73 56.76 59.17 60.63];

%人数 单位：万人

sqjdcs = [0.6 0.75 0.85 0.90 1.05 1.35 1.45 1.60 1.70 1.85 2.15 2.20...

    2.25 2.35 2.50 2.60 2.70 2.85 2.95 3.10];

%机动车数 单位：万辆

sqglmj = [0.09 0.11 0.11 0.14 0.20 0.23 0.23 0.32 0.32 0.34 0.36 0.36...

    0.38 0.49 0.56 0.59 0.59 0.67 0.69 0.79];

%公路面积 单位：万平方千米

glkyl = [5126 6217 7730 9145 10460 11387 12353 15750 18304 19836 21024....

    19490 20433 22598 25107 33442 36836 40548 42927 43462];

%公路客运量 单位：万人

glhyl = [1237 1379 1385 1399 1663 1714 1834 4322 8132 8936 11099 11203 10524....

    11115 13320 16762 18673 20724 20803 21804];

%公路货运量 单位：万吨

p = [sqrs;sqjdcs;sqglmj];   %输入数据矩阵

t = [glkyl;glhyl];          %目标矩阵

%%归一化处理

[pn,minp,maxp,tn,mint,maxt] = premnmx(p,t);  %对pt矩阵进行归一化处理

dx = [-1,1;-1 1;-1,1];                       %

%BP网络训练

net = newff(dx,[3,7,2],{'tansig','tansig','purelin'},'traingdx');

%%建立模型，并用梯度下降法训练

net.trainParam.show = 1000;        % 1000轮回显示一次

net.trainParam.Lr = 0.05;            %学习速率

net.trainParam.epochs = 50000;       %最大训练轮回

net.trainParam.goal = 0.65\*10^(-3);  %均方误差

net = train(net,pn,tn);              %开始训练

%%利用原始数据对BP网络仿真

an = sim(net,pn);                    %用训练好的模型进行仿真

a = postmnmx(an,mint,maxt);          %把仿真到的数据还原到原始的数量级

%%对比测试

x = 1990:2009;

newk = a(1,:);

newh = a(2,:);

figure(2);

subplot(2,1,1);

plot(x,newk,'r-o',x,glkyl,'b--+');

legend('网络输出客运量','实际客运量');

xlabel('年份');

ylabel('客运量/万人');

title('运用数据箱客运量学习和测试对比图');

subplot(2,1,2);

plot(x,newh,'r-o',x,glhyl,'b--+');

legend('网络输出货运量','实际货运量');

xlabel('年份');

ylabel('货运量/万吨');

title('运用数据箱货运量学习和测试对比图');

%利用训练好的网络进行预测

pnew = [73.39 75.55

    3.9635 4.0975

    0.9880 1.0268];

pnewn = tramnmx(pnew,minp,maxp);

%利用原始输入数据的归一化参数对新参数进行归一化

anewn = sim(net,pnewn);

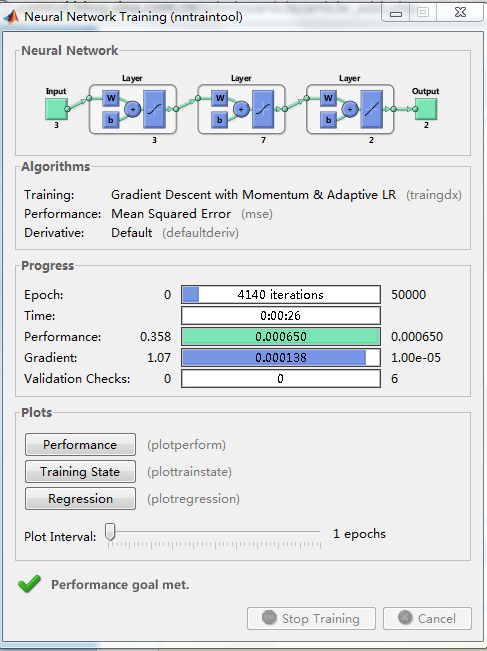
%利用归一化后的数据进行仿真

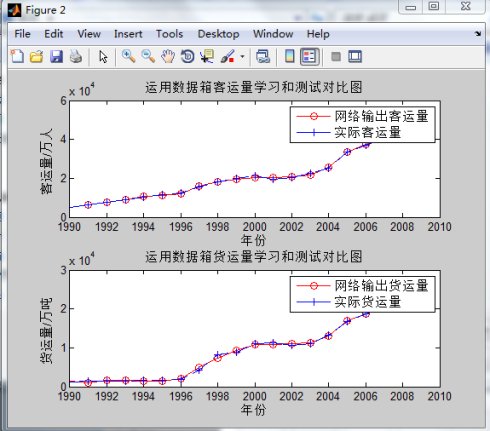
anew = postmnmx(anewn,mint,maxt)

%把仿真得到的数据还原原始的数量级

图形：

具体参数自己设置，这里我改小了些，时间用的比较短。

[](http://photo.blog.sina.com.cn/showpic.html#blogid=82a9278801011yxf&url=http://s15.sinaimg.cn/orignal/82a92788gc8a5beef8eae)

[](http://photo.blog.sina.com.cn/showpic.html#blogid=82a9278801011yxf&url=http://s8.sinaimg.cn/orignal/82a92788gc8a5c86e02a7)