用智能控制理论所学的知识对下面的二维函数建模： $z=\dfrac{\sin(x)\sin(y)}{xy}$ **。**

训练数据取 x=（-10:2:10）=[-10，-8，-6，-4，-2，0，2，4，6，8，10]共 11 个点，y=（-10:2:10）=[-10，-8，-6，-4，-2，0，2，4，6，8，10]共 11 个点，总共的训练数据为（2 维输入 1 维输出）11×11=121 组。建模完成后，用 x=（-10:0.5:10），y=（-10:0.5:10）
的输入数据进行模型测试，并与直接利用解析公式的结果进行对比，体会非线性建模过程中各个参数对性能的影响。

一、 采用 BP 网络

1. **新建脚本文件，按以下思路输入代码：**
   初始化→输入训练数据并输出精确结果曲面→输入测试数据并输出精确结果曲面→生成符合 newff 输入参数格式的训练数据和测试数据矩阵→建立 BP 网络结构→初始化 BP 网络，调整训练过程参数→生成训练后 BP 网络的训练数据和测试数据的输入/输出曲面和输出误差曲面。

2. **具体代码：**

```
close all%关闭打开的figure
clear%清空工作区变量

x=(-10:2:10);
y=(-10:2:10);
z1=sin(x+eps)./(x+eps);  %eps为MATLAB的最小数,
%加上eps是为了避免0/0的错误结果，保证sin(0)/0=1.0
z2=sin(y+eps)./(y+eps);
z=z1'*z2;
mesh(x,y,z);
title('训练数据精确结果曲面');

x2=(-10:0.5:10);
y2=(-10:0.5:10);
z1_2=sin(x2+eps)./(x2+eps);
z2_2=sin(y2+eps)./(y2+eps);
z_2=z1_2'*z2_2;
figure;
mesh(x2,y2,z_2);
title('测试数据精确结果曲面');

%得到训练数据集
a=size(x,2);
b=size(y,2);
trnInput=zeros(2,a*b);
```

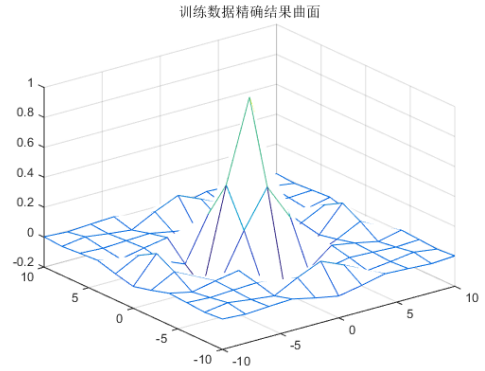
```
for i=1:a
    for j=1:b
        trnInput(1,(i-1)*a+j)=x(i);
        trnInput(2,(i-1)*a+j)=y(j);
    end
end
trnOutput=reshape(z,1,a*b);
%得到测试数据集
c=size(x2,2);
d=size(y2,2);
chkInput=zeros(2,c*d);
for i=1:c
    for j=1:d
        chkInput(1,(i-1)*c+j)=x2(i);
        chkInput(2,(i-1)*c+j)=y2(j);
    end
end
chkOutput=reshape(z_2,1,c*d);

net=newff([minmax(x);minmax(y)],[20 1], {'tansig','purelin'},'trainlm');
net=init(net);                  %初始化网络
net.trainParam.show=50;         %每迭代50次显示一次
net.trainParam.lr=0.05;         %学习速率
net.trainParam.epochs=200;      %训练次数
net.trainParam.goal=0.00001;%误差指标

net=train(net,trnInput,trnOutput);

%训练后BP网络的训练数据输入/输出曲面和输出误差曲面
BPtrnOut=sim(net,trnInput);
BPtrnOut2=reshape(BPtrnOut',a,b);
figure;
mesh(x,y,BPtrnOut2);
title('训练后BP网络的训练数据输入/输出曲面');
figure;
mesh(x,y,BPtrnOut2-z);
title('训练后BP网络的训练数据输出误差曲面');

%训练后BP网络的测试数据输入/输出曲面和输出误差曲面
BPchkOut=sim(net,chkInput);
BPchkOut2=reshape(BPchkOut',c,d);
figure;
mesh(x2,y2,BPchkOut2);
title('训练后BP网络的测试数据输入/输出曲面');
```

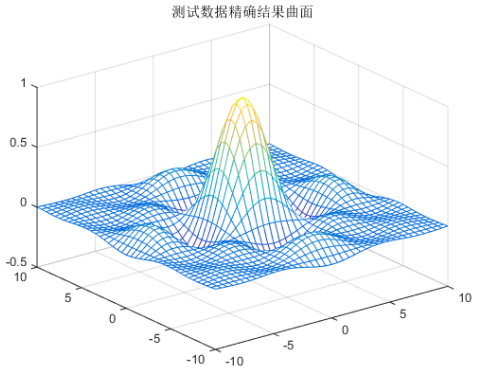
```
figure;
mesh(x2,y2,BPchkOut2-z_2);
title('训练后BP网络的测试数据输出误差曲面');
```

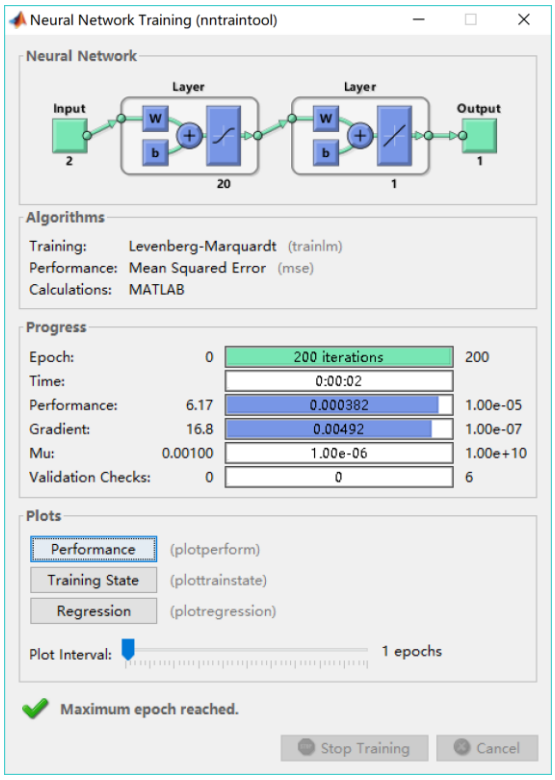
3. 输出结果

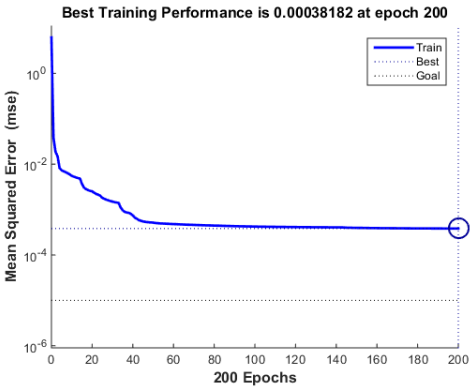
该输出结果对应的BP网络模型参数为隐含层包含20个神经元，采用trainlm训练函数，0.05的学习速率，200次训练次数，0.00001的误差指标。

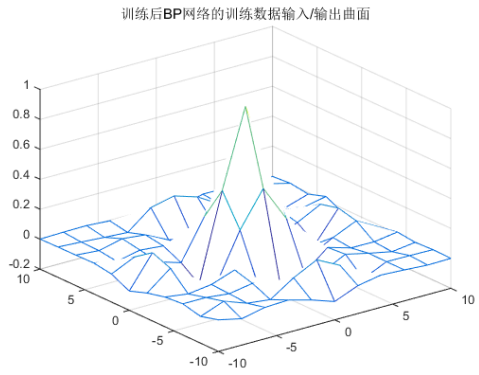
此时BP网络对训练数据的均方根差最小值达不到目标值的0.00001，训练后BP网络的训练数据输出误差在0.15内，测试数据输出误差在0.2内。

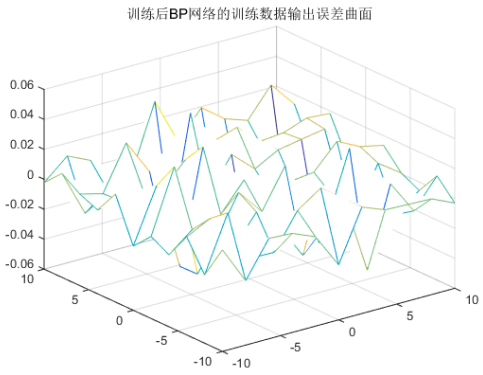


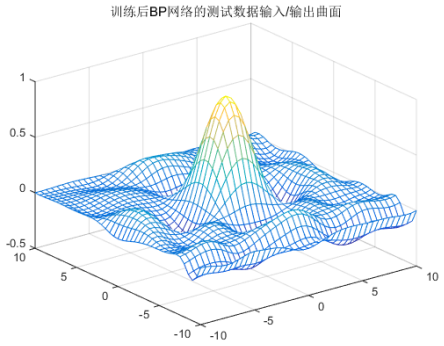


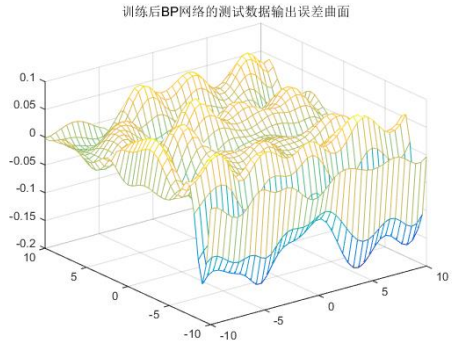






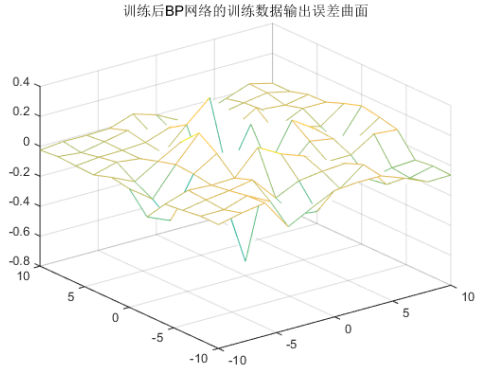


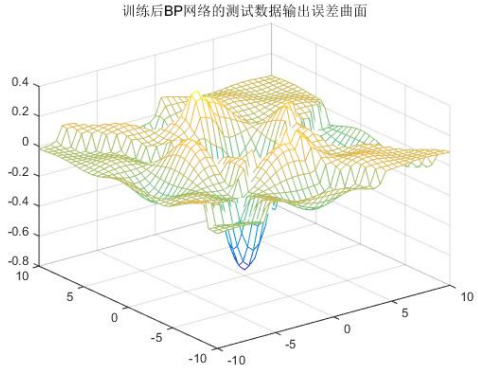




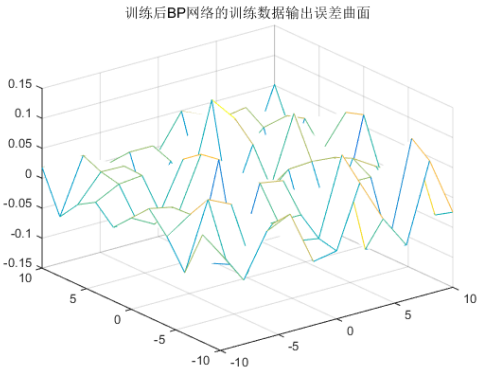
4. BP网络结构和参数对输出结果的影响

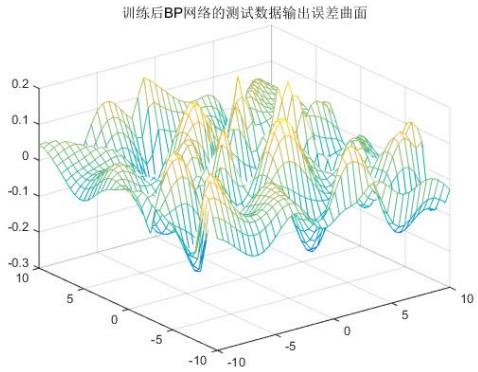
1） 隐含层神经元个数



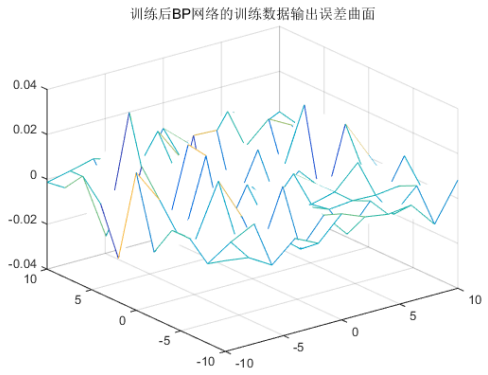


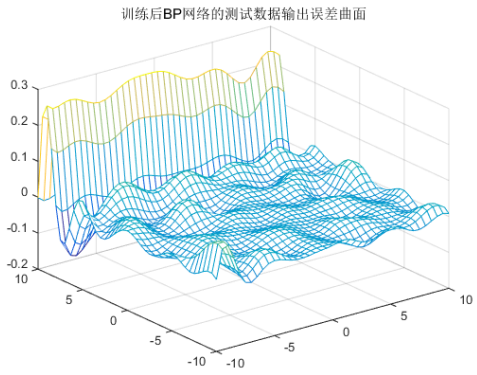
神经元个数为 5





神经元个数为 15

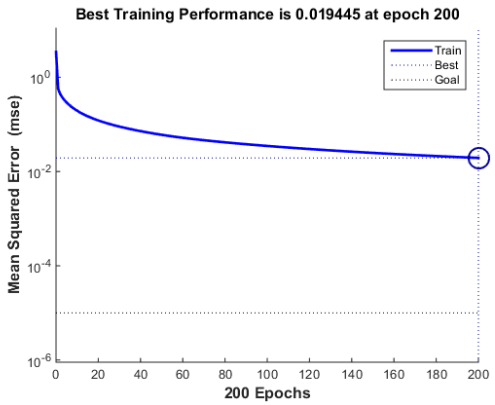




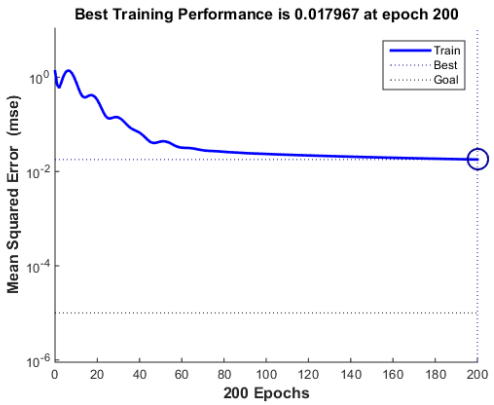
神经元个数为 25

可知随着隐含层神经元个数的增加，训练后 BP 网络的训练数据输出误差逐渐减小，从而提高网络精度。但是注意到一个问题，当神经元个数为 25 时，训练后 BP 网络的测试数据输出误差存在误差较大的局部范围，这反映了 BP 网络训练的相对不稳定性。

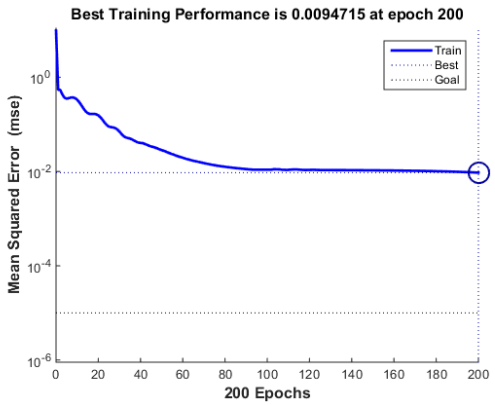
2） 训练函数的选择



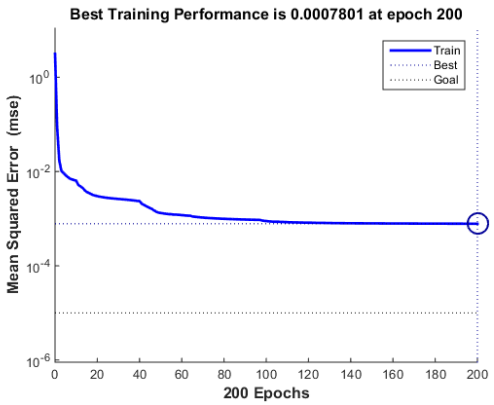
基本梯度下降法 traingd



带有动量项的梯度下降法 traingdm



带有动量项的自适应学习算法 traingdx

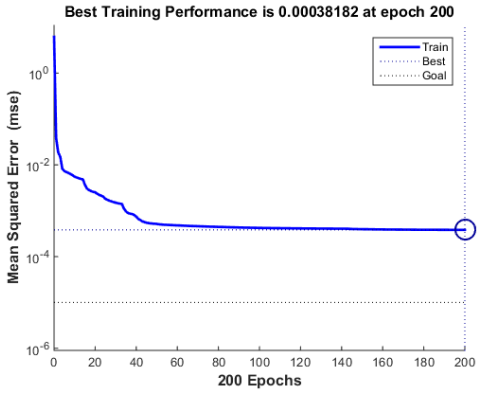


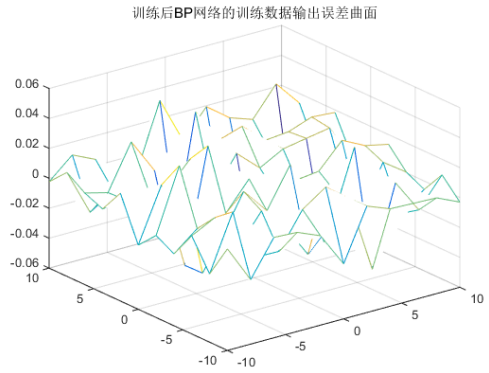
Levenberg-Marquardt 算法 trainlm

可知不同的训练算法具有不同的收敛速度和精度。

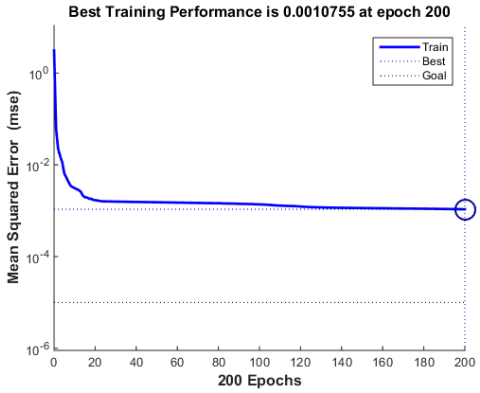
- traingd：基本梯度下降法，收敛速度比较慢
- traingdm：带有动量项的梯度下降法，比 traingd 速度快
- traingdx：带有动量项的自适应学习算法，速度和精度优于 traingdm
- trainlm：速度最快的一种训练算法，其缺点是占用内存较大

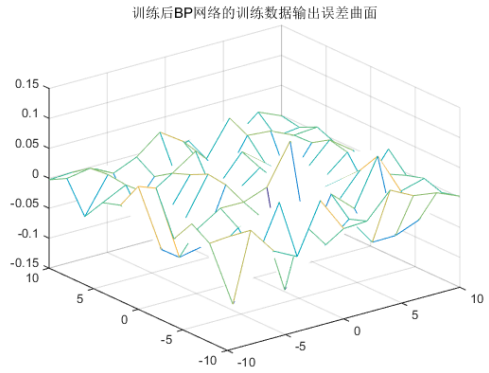
3） 学习速率



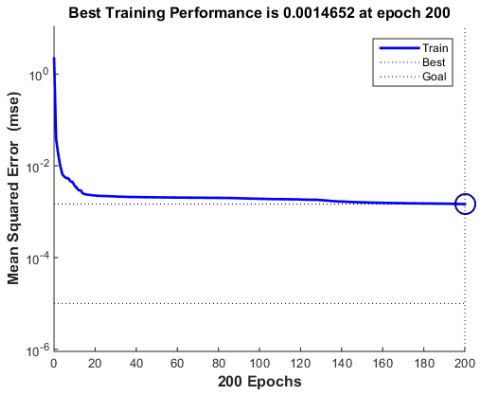


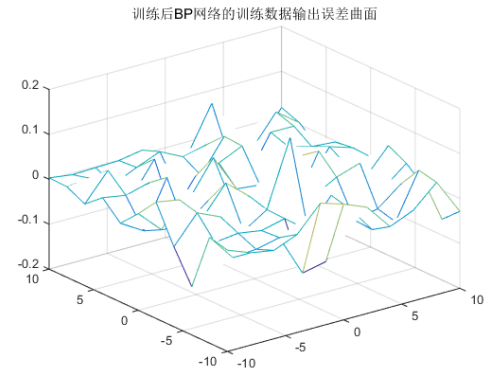
学习速率为 0.05





学习速率为 0.2

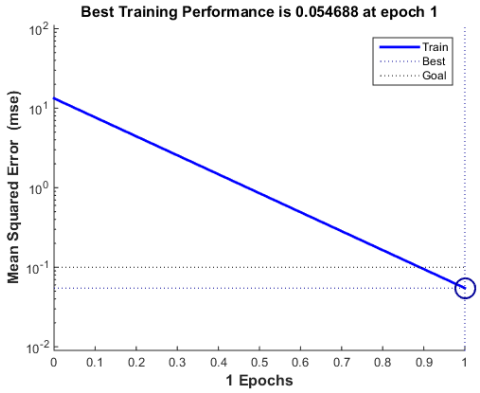


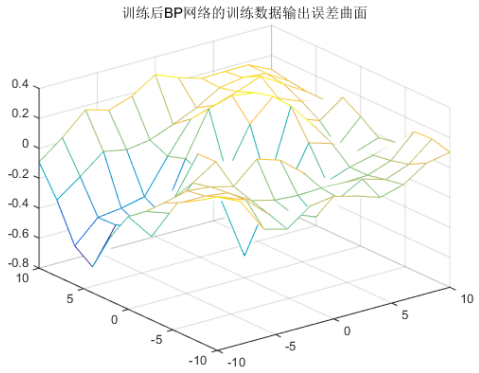


学习速率为 0.8

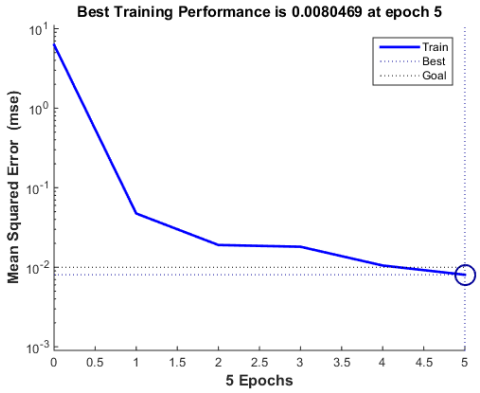
学习速率决定每一次循环训练中所产生的权值变化量。大的学习速率可能导致系统的不稳定；小的学习速率导致较长的训练时间，可能收敛很慢，不过能保证系统不跳出误差表面的低谷而最终趋于最小误差值。

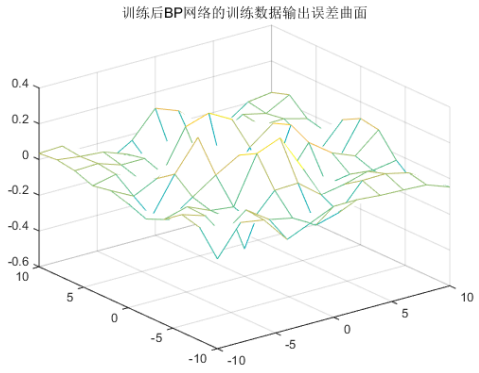
4) 期望误差



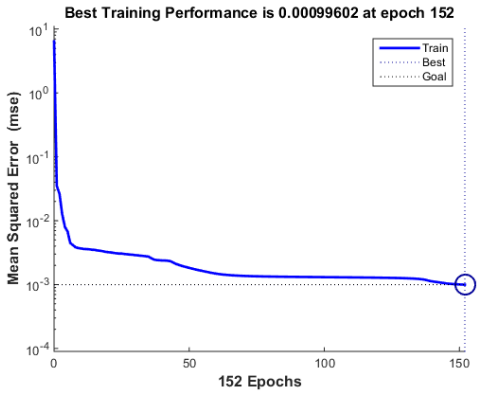


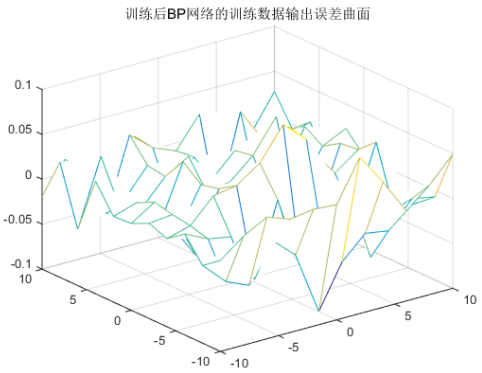
期望误差为 0.1



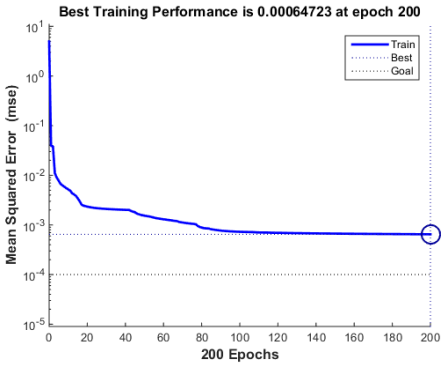


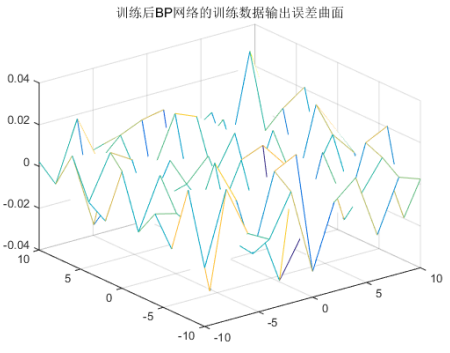
期望误差为 0.01



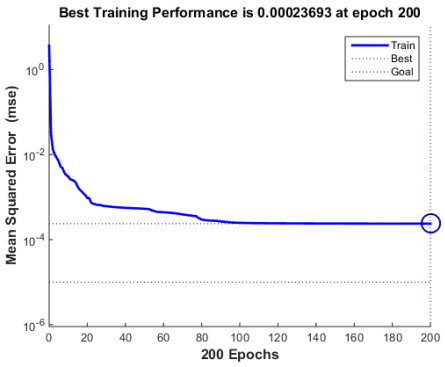


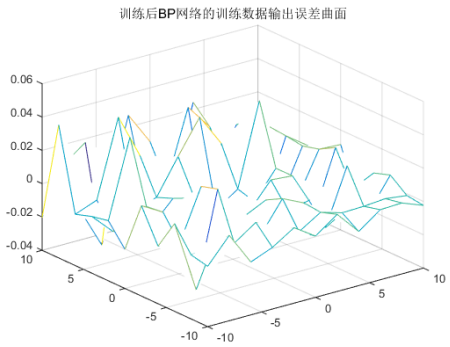
期望误差为 0.001





期望误差为 0.0001





期望误差为 0.00001

在训练次数为 200 次时，若期望误差为 0.1，0.01 和 0.001，训练过程都因达到期望误差而停止训练，若期望误差为 0.0001 和 0.00001，则经过 200 次训练后系统无法达到期望误差。可知，随期望误差的降低，训练后 BP 网络的训练数据输出误差明显减小，精度提高，但当期望误差低于一定程度后，若不调整训练次数，则再降低期望误差已没有实际意义。因而，在设计网络的过程中，期望误差值应当通过对比训练后确定一个合适的值，合适是相对于所需要的隐含层的节点数来确定，因为较小的期望误差值是要靠增加隐含层的节点，以及训练时间来获得的。

二、 采用 ANFIS 网络

1. 新建脚本文件，按以下思路输入代码：

初始化→输入训练数据并输出精确结果曲面→输入测试数据并输出精确结果曲面→生成符合 anfis 输入参数格式的训练数据和测试数据矩阵→指定初始的模糊推理系统参数（包括隶属度函数类型、参数和训练次数）→绘制训练过程均方根误差的变化曲线→生成训练后系统的训练数据和测试数据的输入/输出曲面和输出误差曲面。

2. 具体代码：

```
close all%关闭打开的figure
clear%清空工作区变量

x=(-10:2:10);
y=(-10:2:10);
z1=sin(x+eps)./(x+eps);  %eps为MATLAB的最小数,
%加上eps是为了避免0/0的错误结果，保证sin(0)/0=1.0
z2=sin(y+eps)./(y+eps);
z=z1'*z2;
mesh(x,y,z);
title('训练数据精确结果曲面');

x2=(-10:0.5:10);
y2=(-10:0.5:10);
z1_2=sin(x2+eps)./(x2+eps);
```

```
z2_2=sin(y2+eps)./(y2+eps);
z_2=z1_2'*z2_2;
figure;
mesh(x2,y2,z_2);
title('测试数据精确结果曲面');

%得到训练数据集
a=size(x,2);
b=size(y,2);
trnData=zeros(a*b,3);
for i=1:a
for j=1:b
trnData((i-1)*a+j,1)=x(i);
trnData((i-1)*a+j,2)=y(j);
trnData((i-1)*a+j,3)=z(i,j);
end
end
%得到测试数据集
c=size(x2,2);
d=size(y2,2);
chkData=zeros(c*d,3);
for i=1:c
for j=1:d
chkData((i-1)*c+j,1)=x2(i);
chkData((i-1)*c+j,2)=y2(j);
chkData((i-1)*c+j,3)=z_2(i,j);
end
end

numMFs=7;% x和y都采用7条隶属度函数
mfType='gaussmf';% 采用高斯型隶属度函数
epoch_n=100;% 训练的次数
in_fismat=genfis1(trnData,numMFs,mfType);
[fismat,trnErr,ss]=anfis(trnData,in_fismat,epoch_n);

%绘制训练过程均方根误差的变化曲线，如果训练数据的误差减小，模型才是有效的
epoch=1:epoch_n;
figure;
plot(epoch,trnErr)
title('训练数据误差');

%训练后系统的训练数据输入/输出曲面和输出误差曲面
trnOut=evalfis(trnData(:,[1,2]),fismat);
trnOut2=reshape(trnOut',a,b);
```

```
figure;
mesh(x,y,trnOut2);
title('训练后系统的训练数据输入/输出曲面');
figure;
mesh(x,y,trnOut2-z);
title('训练后系统的训练数据输出误差曲面');

%训练后系统的测试数据输入/输出曲面和输出误差曲面
chkOut=evalfis(chkData(:,[1,2]),fismat);
chkOut2=reshape(chkOut',c,d);
figure;
mesh(x2,y2,chkOut2);
title('训练后系统的测试数据输入/输出曲面');
figure;
mesh(x2,y2,chkOut2-z_2);
title('训练后系统的测试数据输出误差曲面');
```

3. 输出结果

输出结果包括 MATLAB 命令窗口中显示的 ANFIS info 和弹出的七个 Figure 窗口。该输出结果对应每个输入参数的隶属度函数个数为 7 个，采用高斯型隶属度函数，训练 100 次。

由输出结果可知训练过程中，数据误差不断减小，说明训练过程是有效的，且在训练到 80 次后，训练数据均方根误差已小于万分之一。训练后系统的测试数据输出误差也在 0.03 之内。

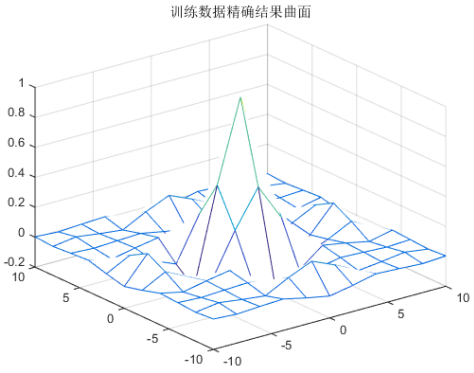
输出结果具体如下所示：

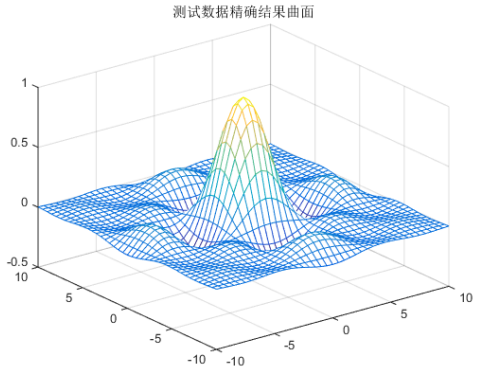
```
ANFIS info:
    Number of nodes: 131
    Number of linear parameters: 147
    Number of nonlinear parameters: 28
    Total number of parameters: 175
    Number of training data pairs: 121
    Number of checking data pairs: 0
    Number of fuzzy rules: 49
Warning: number of data is smaller than number of modifiable parameters

Start training ANFIS ...
    1       0.00675826
    2       0.00664123
    3       0.00652717
    4       0.00641598
    5       0.0063076
Step size increases to 0.011000 after epoch 5.
   ……
   Designated epoch number reached --> ANFIS training completed at epoch 100.
```

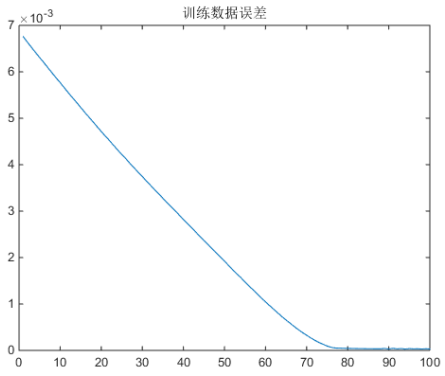
97	3.51141e-05
98	2.96955e-05
99	3.46816e-05
100	3.09716e-05

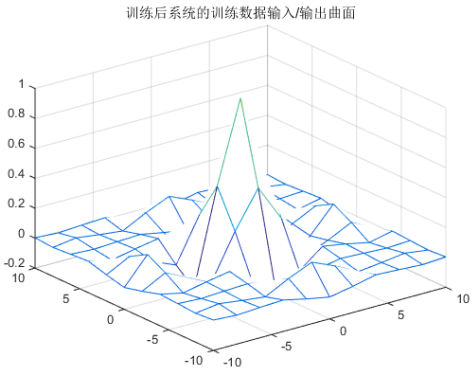
Step size decreases to 0.053948 after epoch 100.

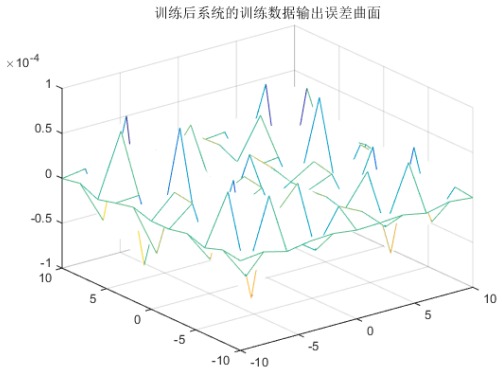
Designated epoch number reached --> ANFIS training completed at epoch 100.

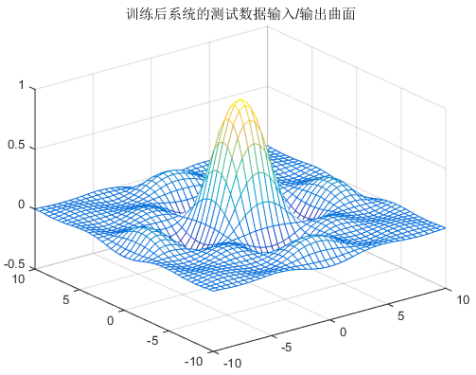


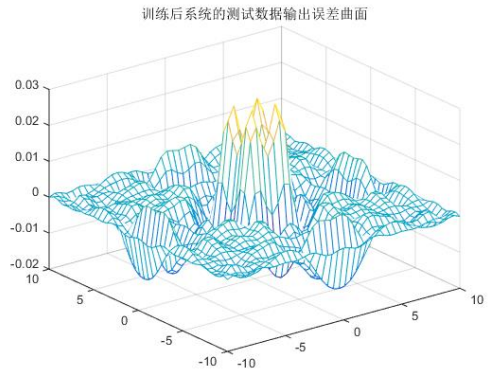






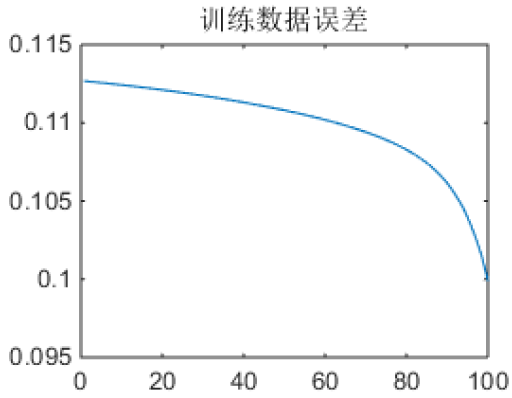


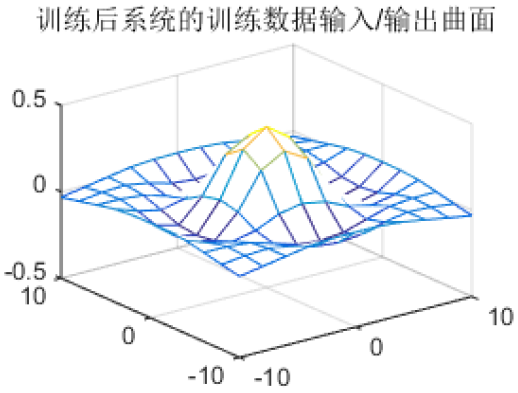


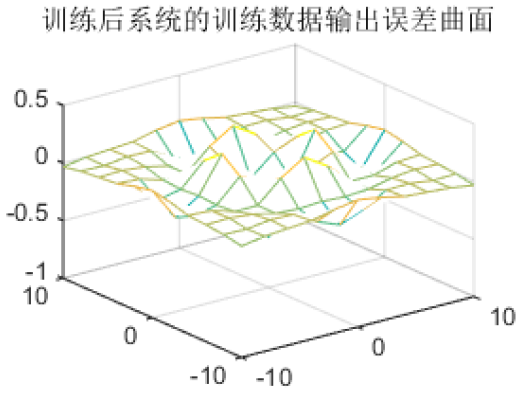


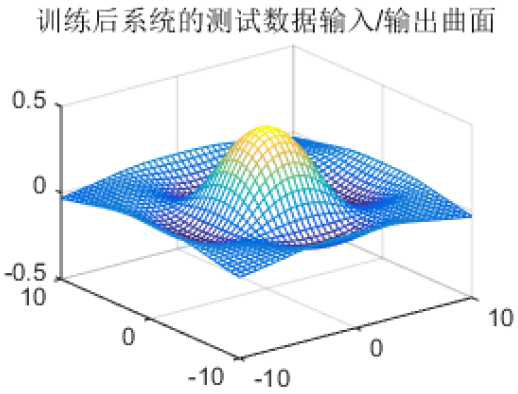
4. FIS结构和参数的调整对输出结果的影响

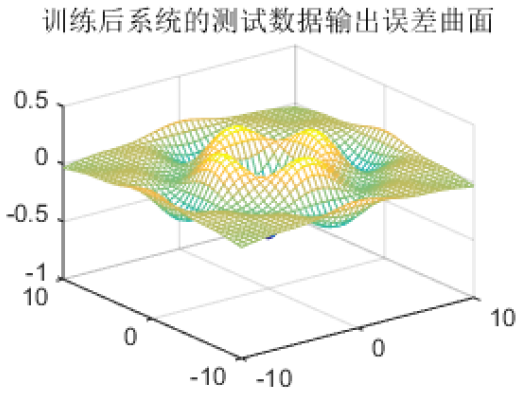
1) 改变隶属度函数条数为3条，由下图结果明显可知系统收敛速度变慢，均方误差变大，这是由于系统的结构不足以反映数据的复杂特性。











2) 改变隶属度函数条数为30条，点击运行后MATLAB命令窗口弹出警告字符，提示所建fis网络可能超出内存。实际上，太多的隶属度函数并不会在系统输出结果精度上体现出优势，反而会造成参数冗余。

模糊逻辑系统对系统的结构和数据要求不是十分严格的，在一定范围内改变模糊规则数目，系统仍然能取得较好的效果，这正是自适应神经网络结构的特点。

```
>> Anfis
警告: genfis1 has created a large rulebase in the FIS.
MATLAB may run out of memory if this FIS is tuned using ANFIS.

> In genfis1 at 147
  In Anfis at 48
```

3) 训练次数对系统输出结果的影响

由训练误差曲线可知，若系统训练过程收敛，则误差曲线衰减程度先快后慢，即当训练达到一定次数后，误差已达到允许范围内。根据这一特点选择合适的训练次数，才能同时满足系统精度和速度的要求。

三、 BP网络和ANFIS网络对比

对比BP网络和ANFIS网络可知，ANFIS的预测结果更接近实际函数值，即ANGIS的函数拟合能力要比BP强。

- ANFIS网络收敛速度更快
- ANFIS的拟合能力和预测能力比BP强
- ANFIS的映射关系更稳定，不随训练数据输入顺序的变化而变化，也不因训练而改变，相反，BP映射关系较不稳定，易出现拟合偏差较大的情况

由于在现实应用中，参数间的关系常常为非线性的，因此ANFIS更适合用来建立现实生活中的参数映射关系模型。

四、 遇到的问题

由于我先编写ANFIS程序而后编写BP网络程序，而两程序除了核心算法不同，其他架构相近，因而我遇到的问题基本都出现在第一次编写ANFIS程序中。

1. 编完ANFIS程序后开始运行，matlab主页面一直显示正忙，按“Ctrl+C”键强行终止程序后，弹出“训练数据精确结果曲面”和“测试数据精确结果曲面”两个窗口。

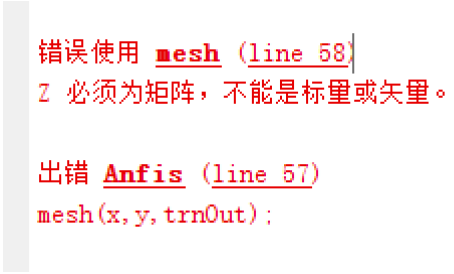
原因及解决办法：

在形成训练数据集时数据形式出错： trnData=[x’ y’ z’];

应改为：

```
a=size(x,2);
b=size(y,2);
trnData=zeros(a*b,3);
for i=1:a
    for j=1:b
        trnData((i-1)*a+j,1)=x(i);
        trnData((i-1)*a+j,2)=y(i);
        trnData((i-1)*a+j,3)=z(i,j);
    end
end
```

2. 在测试ANFIS程序时，遇到问题如下：



原因：由trnOut=evalfis(trnData(:,[1,2]),fismat);语句产生的trnOut向量为一维横向量，不能直接填入mesh函数的输出参数。

解决办法：通过reshape函数将trnOut向量整形为符合格式要求的矩阵，具体语句为trnOut2=reshape(trnOut',a,b);