学院：数据科学与计算机学院　　专业：计算机科学与技术
学号：郑 康 泽　　学号：17341213

智能控制与计算智能

第七章作业

7-1　采用BP网络进行模式识别。训练样本为3对两输入单输出样本，见表7-3。

表7-3 训练样本

输入		输出
1	0	1
0	0	0
0	1	-1

试采用BP网络对训练样本进行训练，并针对一组实际样本进行测试。用于测试的3组样本输入分别为1, 0.1；0.5, 0.5；0.1, 1。

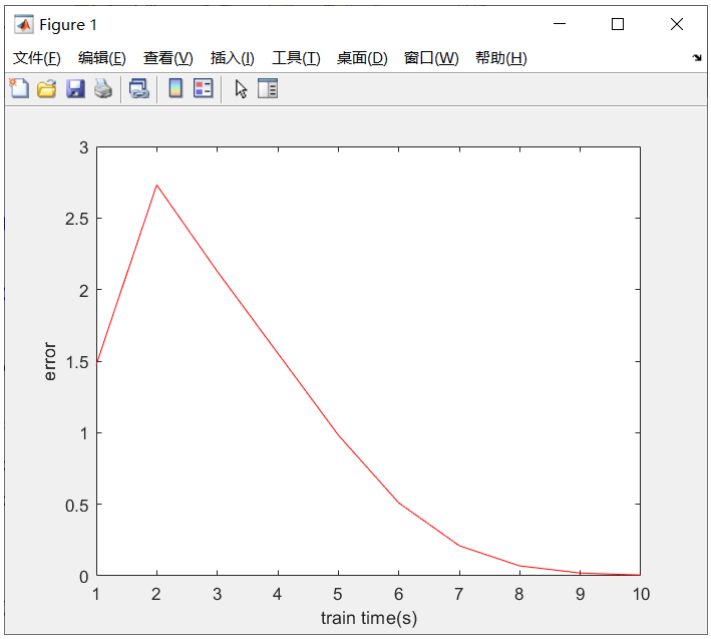
训练网络的程序如下：

```
% BP网络模式识别  训练程序
clear;
close;

xite = 0.50;                        % 学习率
alfa = 0.05;                        % 动量因子

w1 = rands(2, 6);                   % 输入层到隐藏层的权值
w1_1 = w1; w1_2 = w1;               % 前两步的权值
dw1 = 0 * w1;                       % w1的梯度

w2 = rands(6, 1);                   % 隐藏层到输出层的权值
w2_1 = w2; w2_2 = w2_1;             % 前两步的权值
```

```
I = [0, 0, 0, 0, 0, 0]';            % 隐藏层的输入
Iout = [0, 0, 0, 0, 0, 0]';         % 隐藏层的输出
FI = [0, 0, 0, 0, 0, 0]';           % 为计算dw1

NS = 3;                             % 训练样本数
xs=[1, 0; 0, 0; 0, 1];              % 样本的x
ys=[1; 0; -1];                      % 样本的y
k = 0;                              % 训练次数
E = 1.0;                            % 误差

while E >= 1e-2
    k = k+1;
    times(k) = k;                   % x轴

    % 对于每个样本进行训练
    for s=1:NS
        % 计算隐藏层输入、输出
        x = xs(s, :);
        for j=1:6
            I(j) = x * w1(:,j);
            Iout(j) = 1 / (1 + exp(-I(j)));
        end

        % 计算网络输出
        yl = w2' * Iout;

        % 计算单个样本的误差
        y = ys(s);
        el = 0.5 * (y - yl)^2;
        es(s) = el;

        % 计算所有样本的误差
        E=0;
        if s == NS
           for s = 1:NS
              E = E + es(s);
           end
        end

        % 更新w2
        ey = y - yl;
        w2 = w2_1 + xite * Iout * ey + alfa * (w2_1 - w2_2);

        % 更新w1
        for j = 1:6
           S = 1 / (1 + exp(-I(j)));
           FI(j) = S * (1 - S);
        end
        for i = 1:2
           for j = 1:6
               dw1(i, j) = xite * FI(j) * x(i) * ey(1) * w2(j,1);
           end
        end
        w1 = w1_1 + dw1 + alfa * (w1_1 - w1_2);

        % 更新参数
        w1_2 = w1_1; w1_1 = w1;
```

```
        w2_2 = w2_1; w2_1 = w2;
    end

    % 记录本次误差
    Ek(k)=E;
end

figure(1);
plot(times, Ek, 'r');
xlabel('train time(s)'); ylabel('error');

save wfile w1 w2;
```

训练误差随训练次数的变化：



利用训练好的参数进行测试，得到以下预测输出：

测试样本

输入		输出
1	0.1	0.9001
0.5	0.5	0.0189
0.1	1	-0.9093

7-2　采用BP网络、RBF网络逼近非线性对象

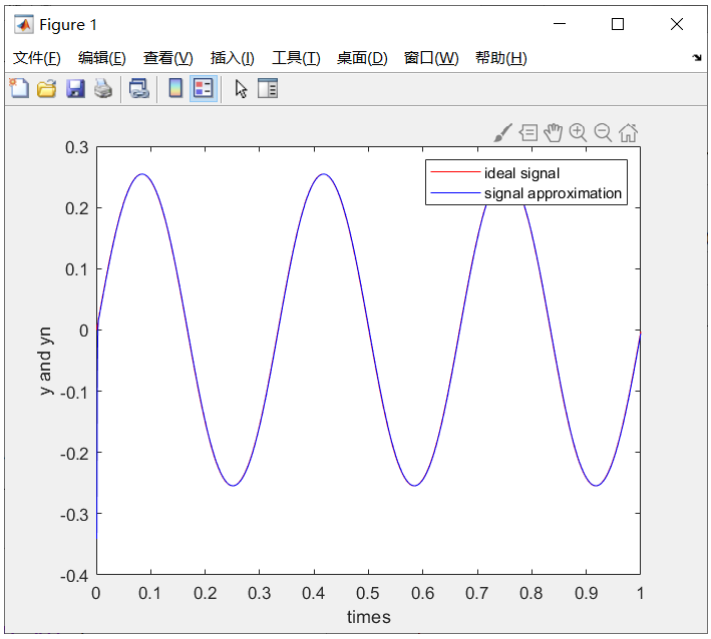
$y(k)=(u(k-1)-0.9\,y(k-1))/(1+y(k-1)^2)$，分别进行Matlab仿真。

BP网络逼近程序如下：

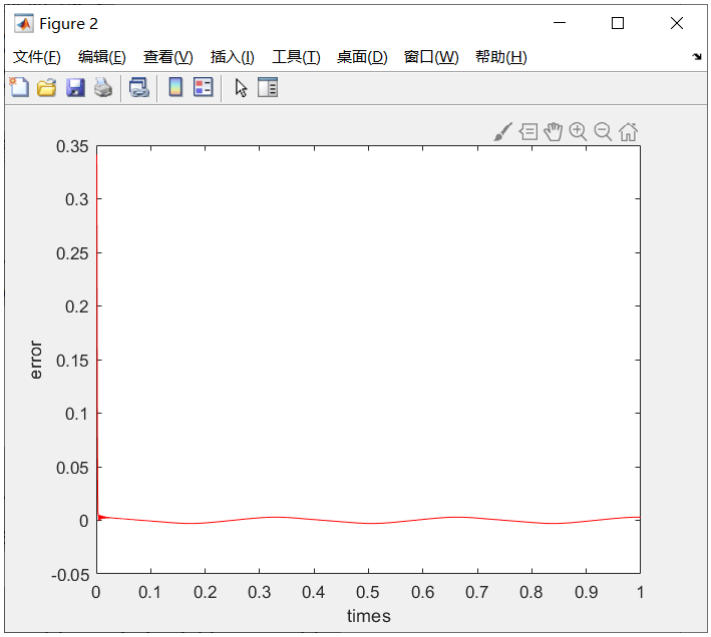
```
% BP网络逼近对象
clear;
close;

xite = 0.50;                % 学习率
alfa = 0.05;                % 动量因子

w1 = rands(2,6);            % 输入层到隐藏层的权值
w1_1 = w1; w1_2 = w1;       % 前两步的权值
dw1 = 0*w1;                 % w1梯度

w2 = rands(6,1);            % 隐藏层到输出层的权值
w2_1 = w2; w2_2 = w2_1;     % 前两步的权值

x = [0, 0]';                % 网络的输入

u_1 = 0;                    % 上一步的u
y_1 = 0;                    % 上一步的y

I = [0, 0, 0, 0, 0, 0]';    % 隐藏层的输入
Iout = [0, 0, 0, 0, 0, 0]'; % 隐藏层的输出
FI = [0, 0, 0, 0, 0, 0]';   % 为计算dw1

ts = 0.001;                 % 采样时间
for k = 1:1000
    time(k) = k * ts;       % x轴
    % 对象的输入输出
    u(k) = 0.50 * sin(6 * pi * k * ts);
    y(k) = (u_1 - 0.9 * y_1) / (1 + y_1^2);

    % 计算隐藏层的输入输出
    for j = 1:6
         I(j) = x' * w1(:,j);
         Iout(j) = 1 / (1 + exp(-I(j)));
    end

    yn(k) = w2'*Iout;       % 网络的输出
    e(k) = y(k)-yn(k);      % 误差

    % 更新w2
    w2 = w2_1+ xite * e(k) * Iout + alfa * (w2_1 - w2_2);

    % 更新w1
    for j = 1:6
       FI(j) = exp(-I(j)) / (1 + exp(-I(j)))^2;
    end
    for i = 1:2
       for j = 1:6
          dw1(i, j) = e(k) * xite * FI(j) * w2(j) * x(i);
       end
    end
    w1 = w1_1 + dw1 + alfa * (w1_1 - w1_2);

    % 计算Jacobian
    yu = 0;
```

```
    for j = 1:6
        yu = yu + w2(j) * w1(1,j) * FI(j);
    end
    dyu(k)=yu;

    % 设置网络输入
    x(1)=u(k);
    x(2)=y(k);

    % 更新参数
    w1_2 = w1_1; w1_1 = w1;
    w2_2 = w2_1; w2_1 = w2;
    u_1 = u(k);
    y_1 = y(k);
end

%画图
figure(1);
plot(time, y ,'r', time, yn, 'b');
xlabel('times'); ylabel('y and yn');
legend('ideal signal', 'signal approximation');
figure(2);
plot(time, y-yn, 'r');
xlabel('times'); ylabel('error');
figure(3);
plot(time, dyu);
xlabel('times'); ylabel('dyu');
```

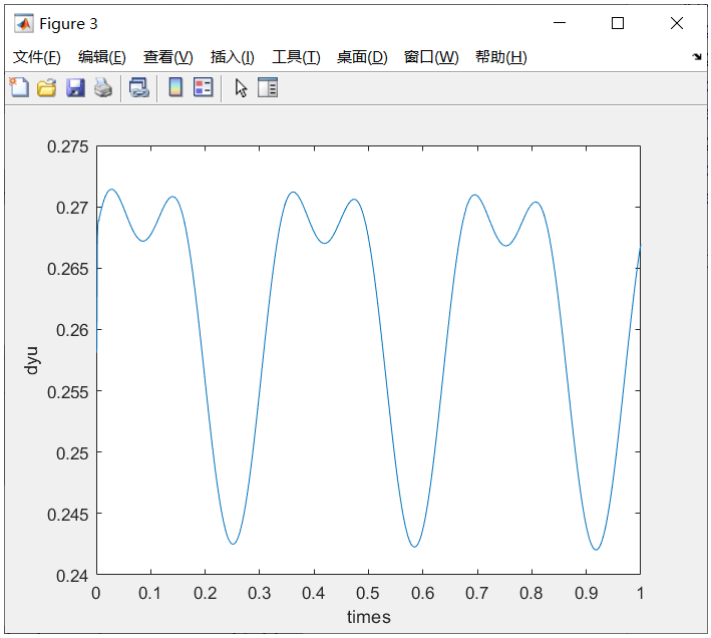
逼近结果如图：



误差如图：



Jacobian如图：



RBF网络逼近程序如下：

```
% RBF网络逼近对象
clear;
close;

alfa = 0.05;                        % 动量因子
xite = 0.15;                        % 学习率
x = [0,1]';                         % 网络输入

b = 3 * ones(5, 1);                 % 宽度
c = [-1 -0.5 0 0.5 1;               % 中心
     -1 -0.5 0 0.5 1];

w = rands(5, 1);                    % 隐藏层到输出层的权值
w_1 = w; w_2 = w_1;                 % 前两步的权值
d_w = 0 * w;                        % w的梯度

u_1 = 0;                            % 前一步的u
```

```
y_1 = 0;                                % 前一步的y

ts = 0.001;                             % 采样时间

for k = 1:10000

    time(k) = k * ts;                   % x轴
    % 对象的输入输出
    u(k) = 0.5 * sin(3 * k * ts);
    y(k) = (u_1 - 0.9 * y_1) / (1 + y_1^2);

    % 设置网络输入
    x(1) = u(k);
    x(2) = y(k);

    % 计算隐藏层输出
    for j=1:5
        h(j) = exp(-norm(x - c(:, j))^2 / (2 * b(j) * b(j)));
    end

    % 计算网络输出
    ym(k) = w' * h';

    % 计算误差
    em(k) = y(k) - ym(k);

    % 更新w
    d_w(j) = xite * em(k) * h(j);
    w = w_1 + d_w + alfa * (w_1 - w_2);

    % 更新参数
    u_1 = u(k);
    y_1 = y(k);

    w_2 = w_1;
    w_1 = w;
end
figure(1);
plot(time, y, 'r', time, ym, 'k:', 'linewidth', 2);
xlabel('time(s)'); ylabel('y and ym');
legend('ideal signal', 'signal approximation');
figure(2);
plot(time, y-ym, 'k', 'linewidth', 2);
xlabel('time(s)'); ylabel('error');
```

逼近结果如图：

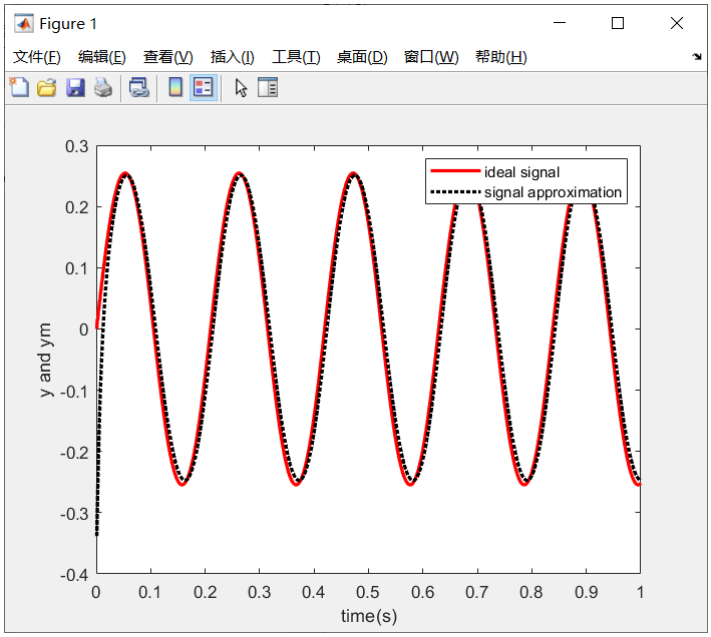


Figure 1
文件(F) 编辑(E) 查看(V) 插入(I) 工具(T) 桌面(D) 窗口(W) 帮助(H)
ideal signal
signal approximation
y and ym
time(s)

误差如图：

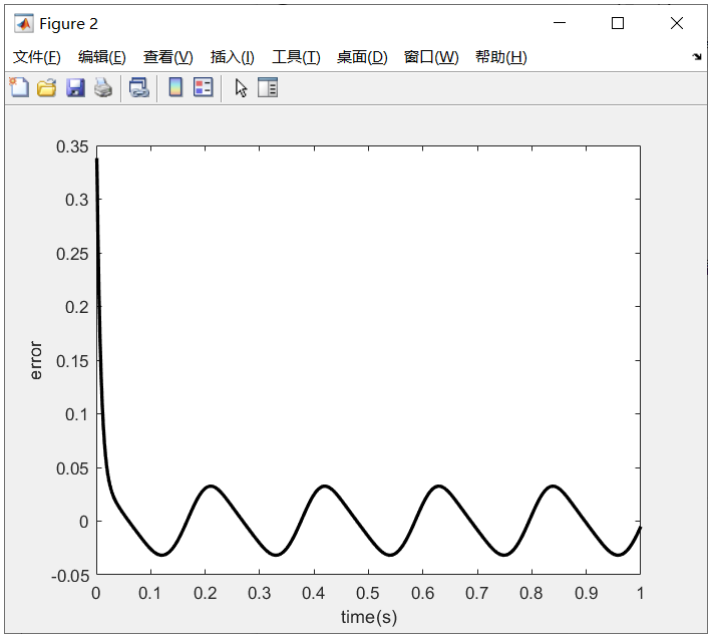


Figure 2
文件(F) 编辑(E) 查看(V) 插入(I) 工具(T) 桌面(D) 窗口(W) 帮助(H)
error
time(s)