

《计算机先进控制》实验报告

实验六：RBF神经网络实验

学 院： 机电学院

专 业： 机械工程

学生姓名： 朱天启

学 号： 24126147

北京交通大学

1.自行设计一个Hopfield神经网络：

代码如下：

clc;

clear;

close all;

% 参数初始化

% 权值矩阵（6节点）

W = [0 1.2 -1.6 0 0 1;

1 0 1 -1 0 0.5;

-1 1 0 1.7 -1 0.7;

0 -1 1.1 0 1.6 -1;

0 0.7 -1.1 1 0 1;

1 0 0 -1.9 1 0.5];

% 初态

S = [0.3; -0.5; 1.1; 0.6; -0.4; 0.3]; % 随机设置初始状态

theta = [0.5; 0.5; 0.5; 0.5; 0.5; 0.5]; % 阈值向量

iterations = 10; % 更新迭代次数

% 定义sign函数

sign\_fn = @(x) (x >= 0) \* 2 - 1; % 1 if x >= 0, -1 otherwise

% 能量函数计算

calculate\_energy = @(S, W, theta) ...

-0.5 \* S' \* W \* S + sum(theta .\* S);

% 输出初始状态和能量

disp('Initial state:'), disp(S');

E = calculate\_energy(S, W, theta);

fprintf('Initial Energy: %.4f\n\n', E);

% 逐步更新节点状态

for t = 1:iterations

fprintf('Iteration %d:\n', t);

state\_changed = false; % 标记状态是否变化

for i = 1:length(S)

% 计算净输入

net\_input = W(i,:) \* S - theta(i);

new\_state = sign\_fn(net\_input);

% 检查节点状态是否发生变化

if new\_state ~= S(i)

state\_changed = true;

end

% 更新节点状态

S(i) = new\_state;

fprintf('Node %d updated to %d\n', i, S(i));

end

% 显示当前状态

disp('Current state:'), disp(S');

% 计算能量

E = calculate\_energy(S, W, theta);

fprintf('Energy: %.4f\n\n', E);

% 如果状态不再变化，停止迭代

if ~state\_changed

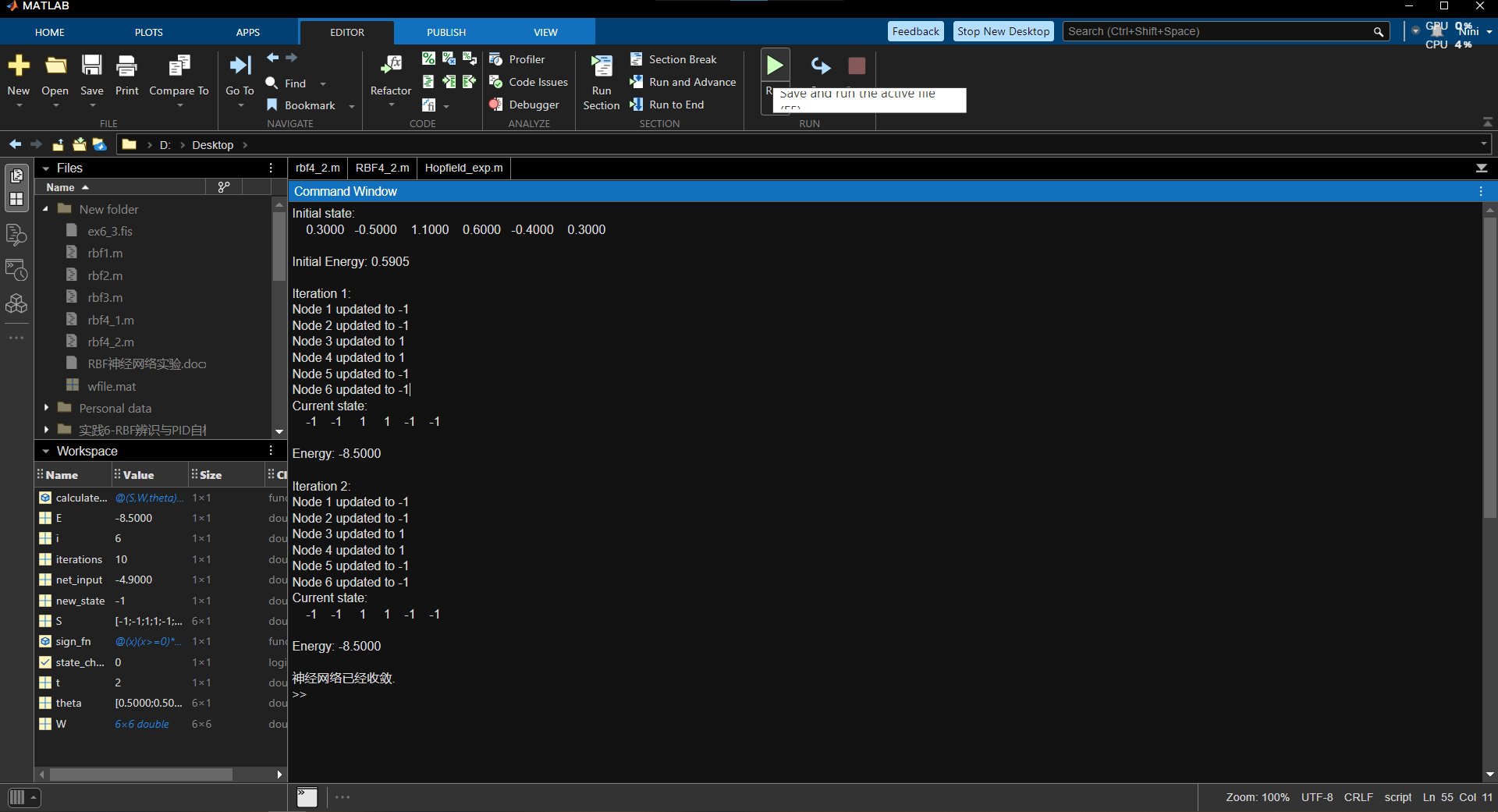
fprintf('神经网络已经收敛.\n');

break;

end

end

运行结果如下：



2. G(s)=400/s(s+50)，Kp的取值范围为[0，20]，Ki，Ka的取值范围为[0,1]

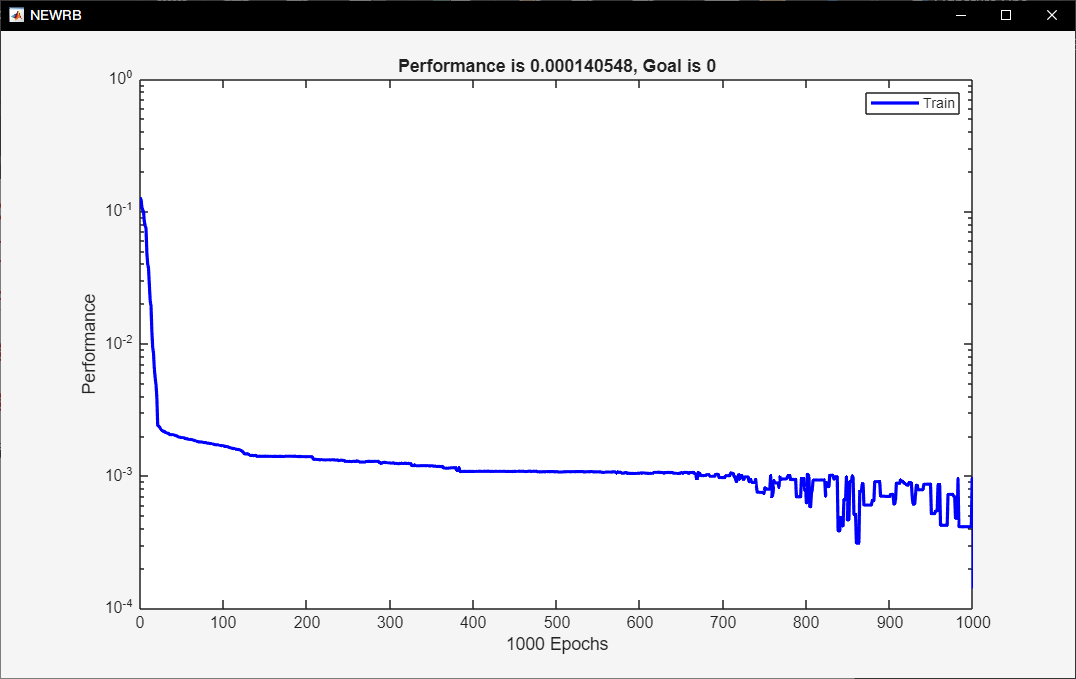
1.设计PID控制器进行信号r=0.5sin(10t)跟踪。

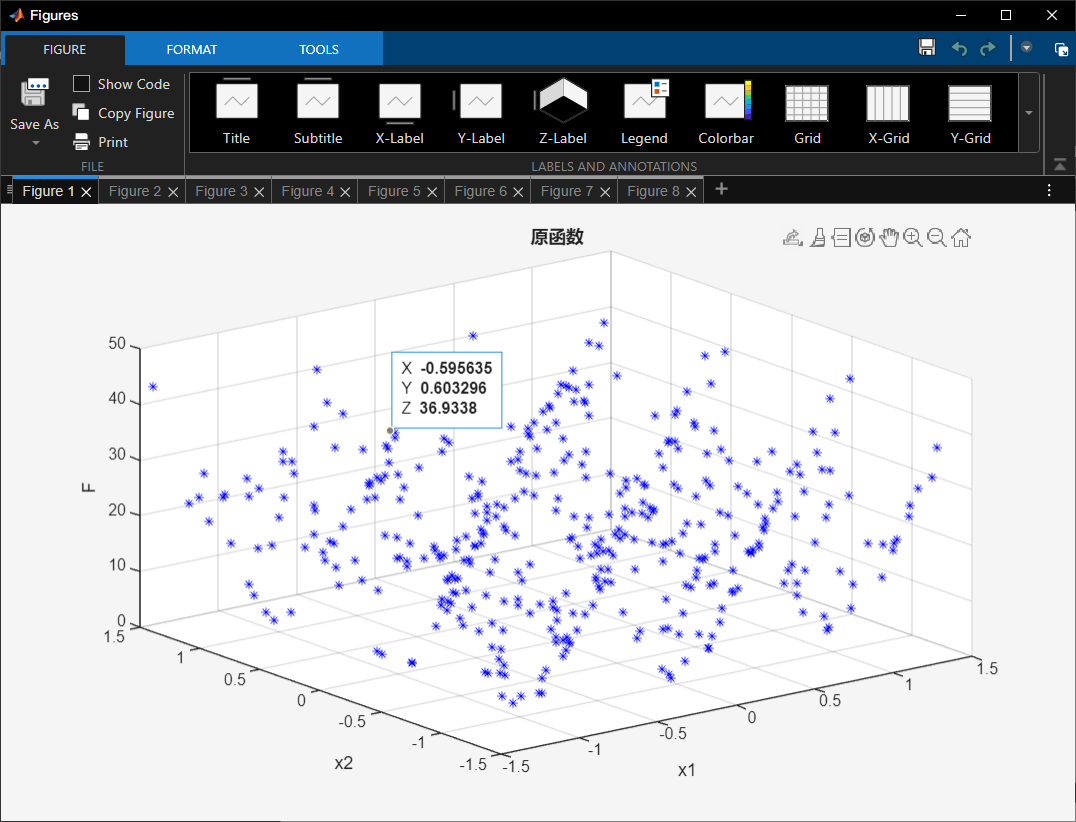
2.训练RBF和BP网络跟踪PID控制器。

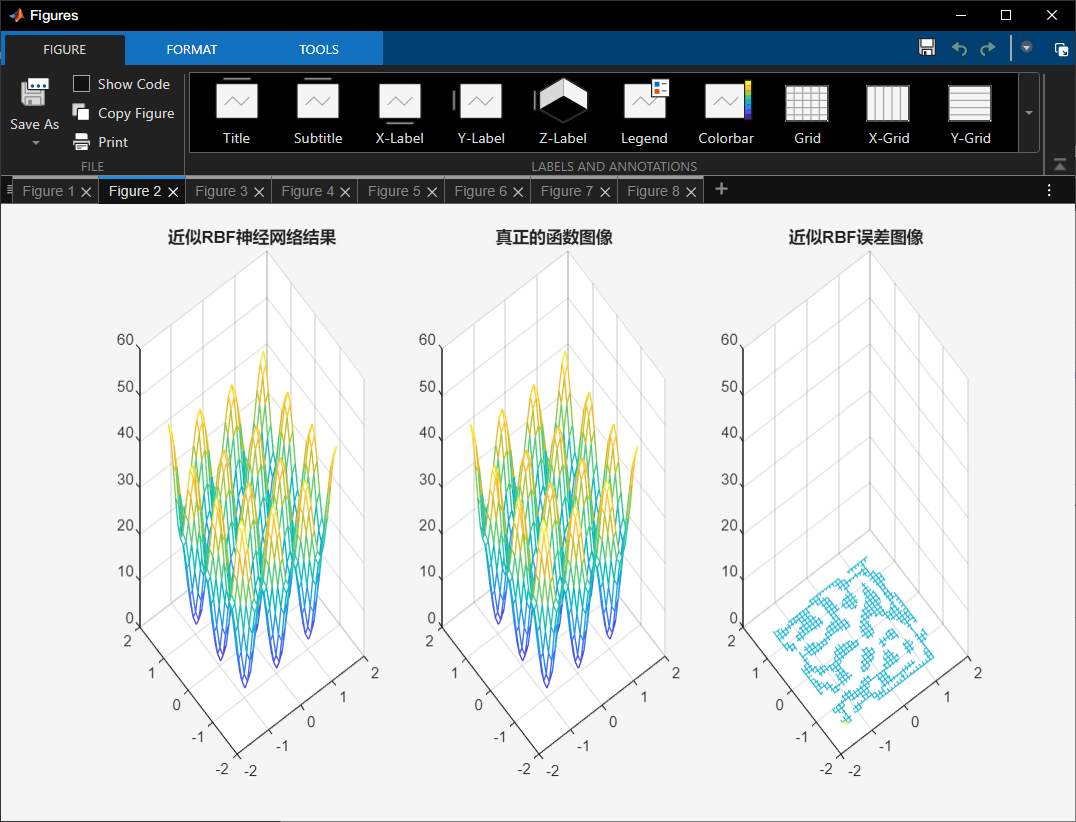
3.训练模糊神经网络跟踪PID控制器，并比较三种网络的跟踪效果。

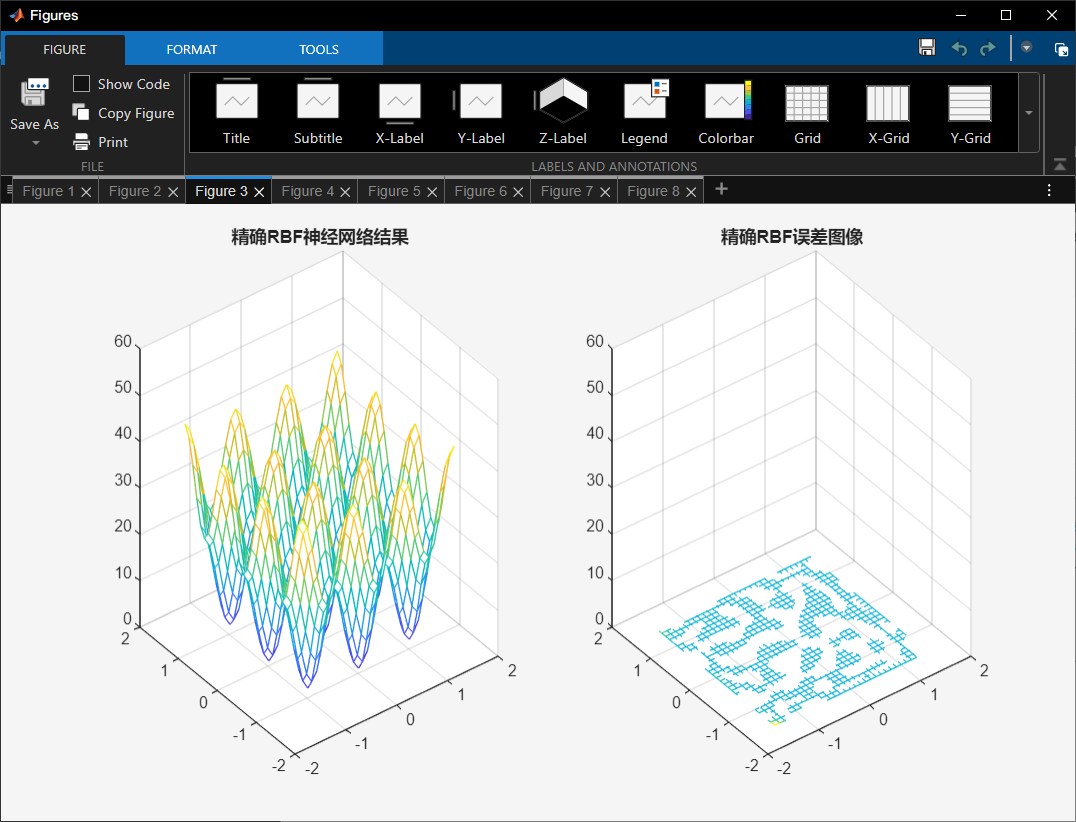
代码见Exp2.m

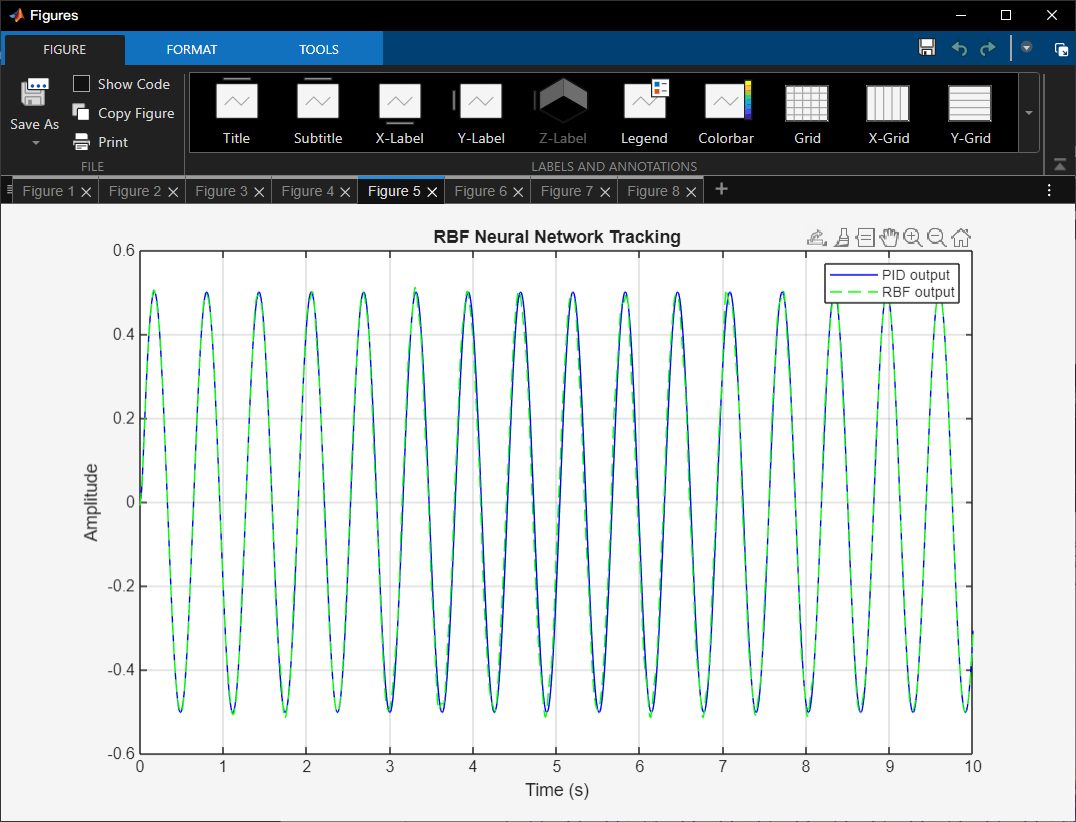
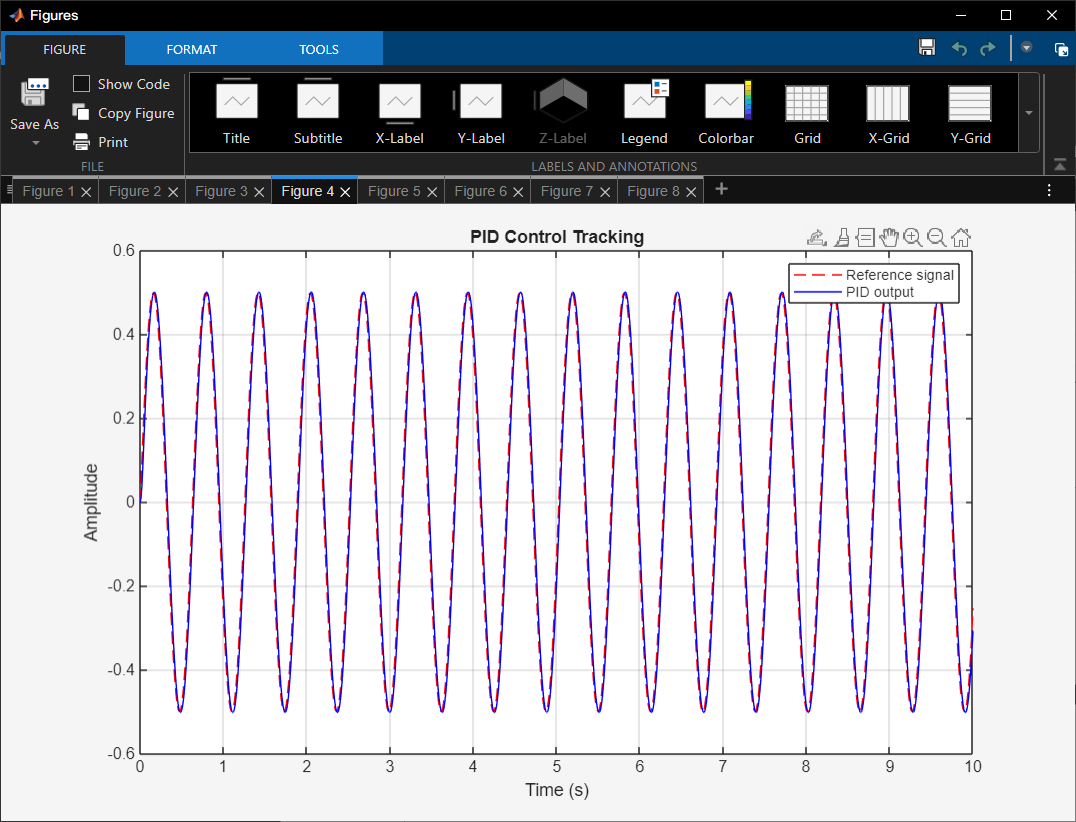
运行结果：

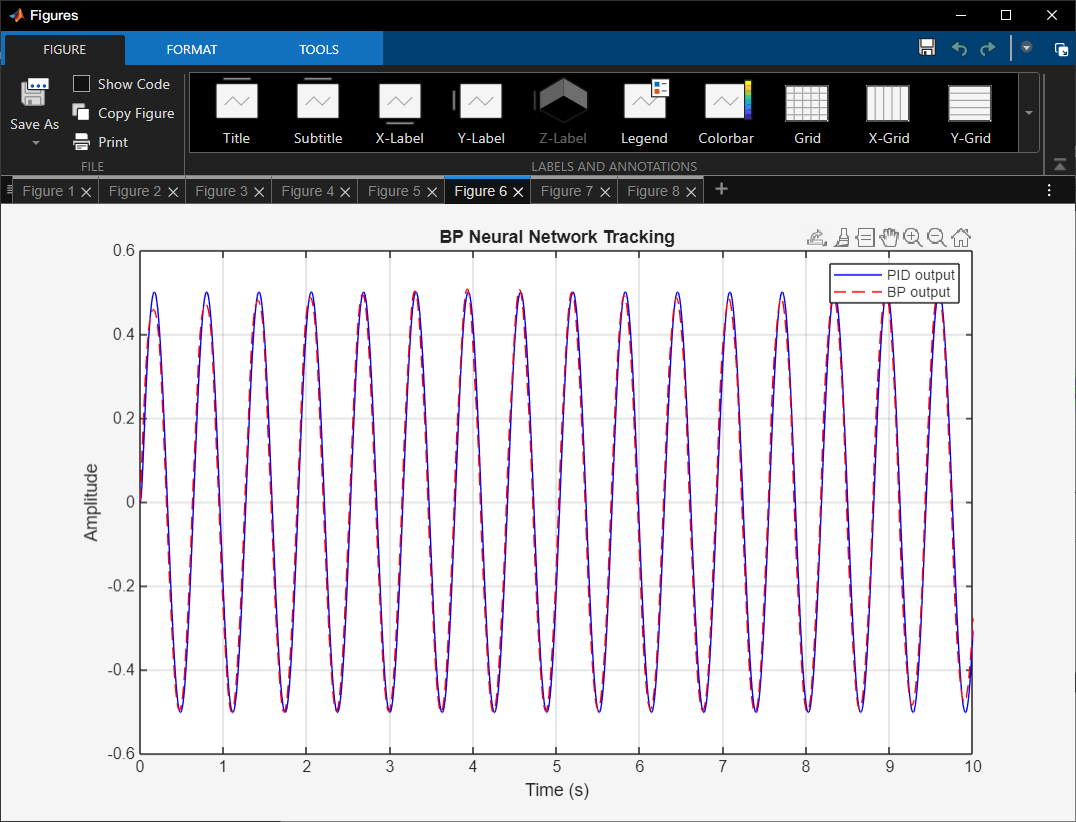


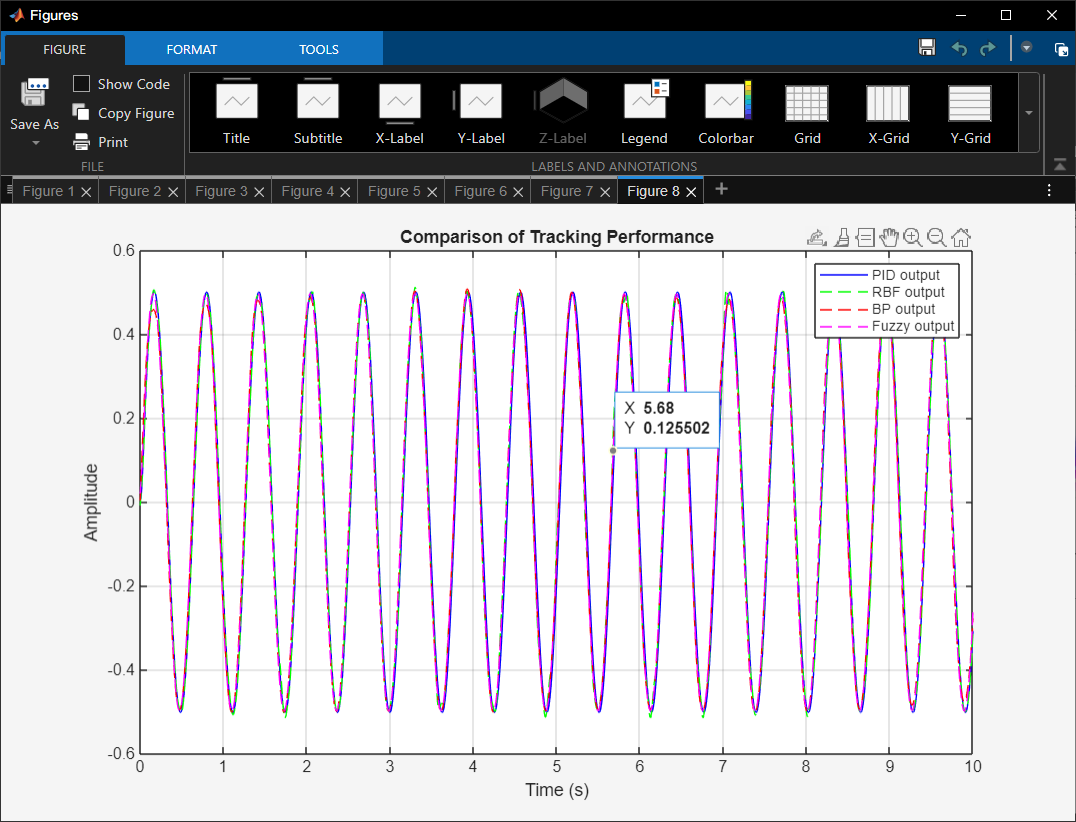
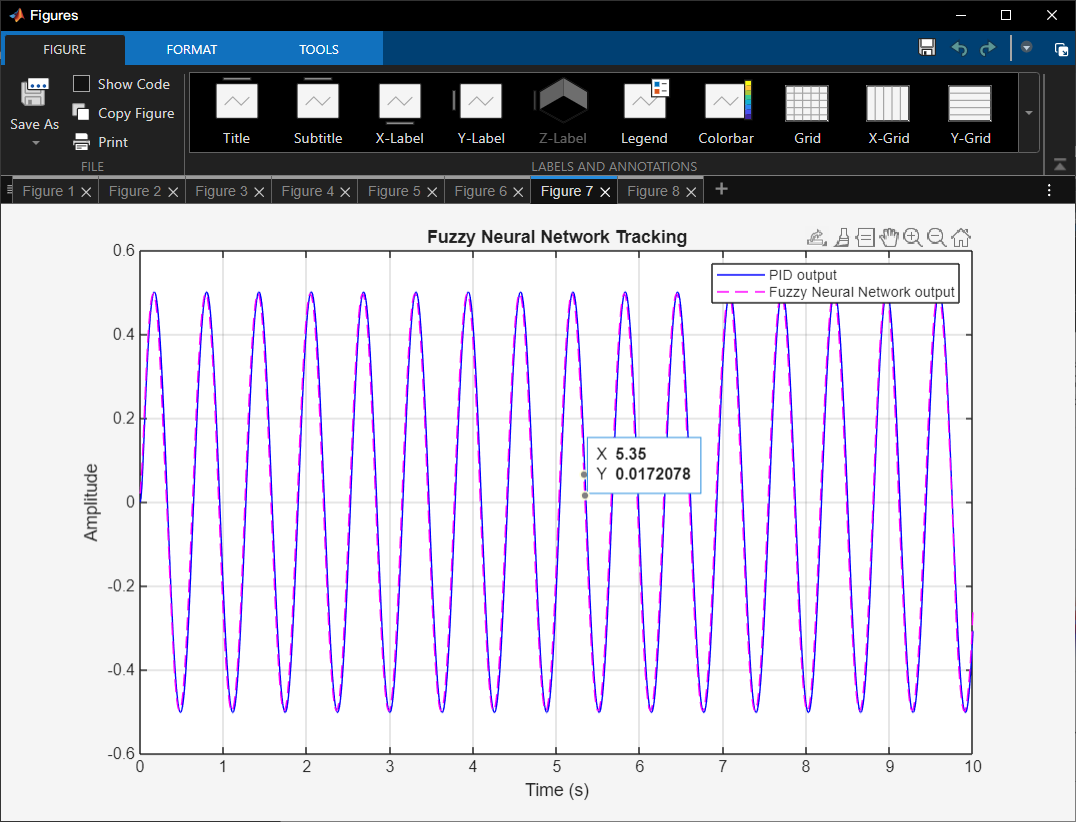


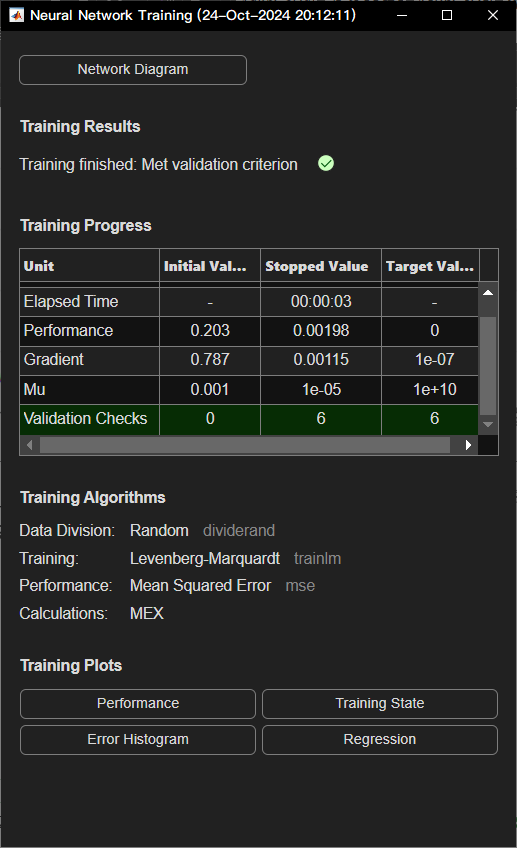












3.修改rbf4\_2.m，使用同样数据分别建立近似和精确RBF网络进行回归分析，比较两种网络的拟合效果。

代码如下：

clc;

clear;

close all;

% 数据生成

ld = 400;

x = rand(2, ld); % 生成一个2行ld列的随机矩阵，值在0到1之间

x = (x - 0.5) \* 1.5 \* 2; %将数据缩放到-1.5到1.5之间

x1 = x(1,:); % 提取第一行作为x1

x2 = x(2,:); % 提取第二行作为x2

F = 20 + x1.^2 - 10 \* cos(2 \* pi \* x1) + x2.^2 - 10 \* cos(2 \* pi \* x2); % 目标函数

% 创建RBF径向基网络 - 近似解

net\_rb = newrb(x, F); % 近似 RBF 网络

% 创建RBF径向基网络 - 精确解

net\_rbe = newrbe(x, F); % 精确 RBF 网络

% 生成测试数据

interval = 0.1; %步长

[i, j] = meshgrid(-1.5:interval:1.5, -1.5:interval:1.5); % 生成网格点

row = size(i); % 获取网格大小

% 将i，j转换为行向量作为输入数据

tx1 = i(:)';

tx2 = j(:)';

tx = [tx1; tx2];

% 使用两种网络进行测试

ty\_rb = sim(net\_rb, tx); % 近似 RBF 网络的输出

ty\_rbe = sim(net\_rbe, tx); % 精确 RBF 网络的输出

% 使用图像查看两种网络对非线性函数的拟合效果

figure(1)

plot3(x1, x2, F, 'b\*');

hold on;

title('原函数')

grid on

xlabel('x1')

ylabel('x2')

zlabel('F')

% 将网络结果重塑为网格形状

v\_rb = reshape(ty\_rb, row); % 近似 RBF 网络的拟合结果

v\_rbe = reshape(ty\_rbe, row); % 精确 RBF 网络的拟合结果

figure(2)

subplot(1,3,1)

mesh(i, j, v\_rb);

zlim([0, 60])

title('近似RBF神经网络结果')

% 绘制原函数图像

interval = 0.1;

[x1\_mesh, x2\_mesh] = meshgrid(-1.5:interval:1.5);

F\_mesh = 20 + x1\_mesh.^2 - 10 \* cos(2 \* pi \* x1\_mesh) + x2\_mesh.^2 - 10 \* cos(2 \* pi \* x2\_mesh);

subplot(1,3,2)

mesh(x1\_mesh, x2\_mesh, F\_mesh);

zlim([0, 60])

title('真正的函数图像')

% 绘制误差图

subplot(1,3,3)

mesh(x1\_mesh, x2\_mesh, F\_mesh - v\_rb); % 误差图

zlim([0, 60])

title('近似RBF误差图像')

% 显示精确RBF结果

figure(3)

subplot(1,2,1)

mesh(i, j, v\_rbe);

zlim([0, 60])

title('精确RBF神经网络结果')

% 精确RBF误差图

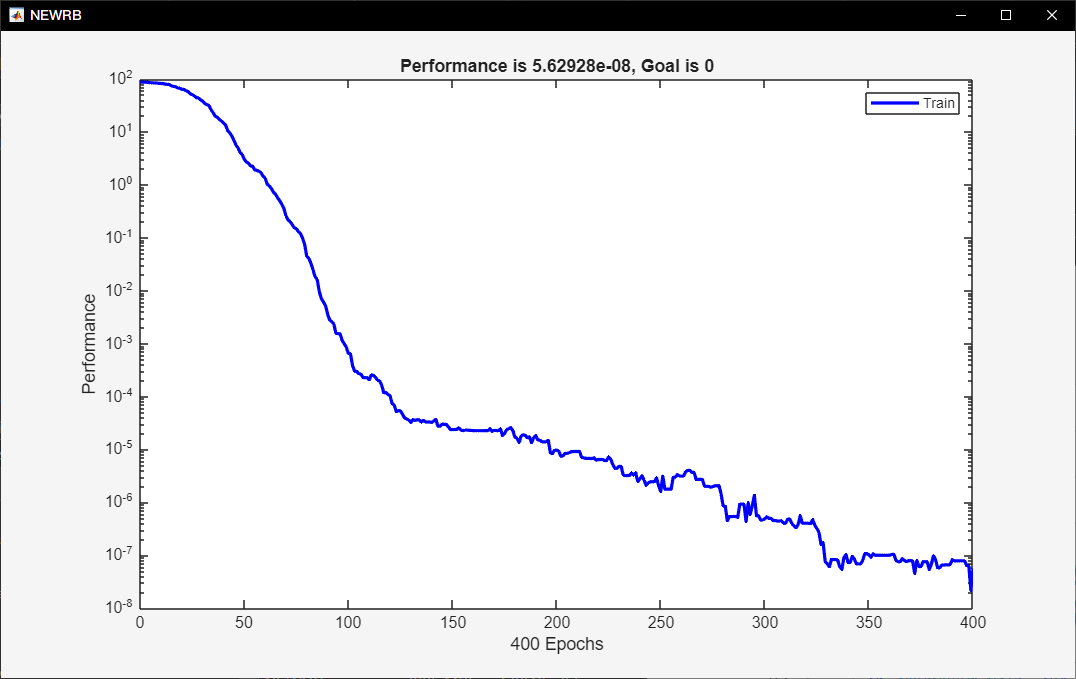
subplot(1,2,2)

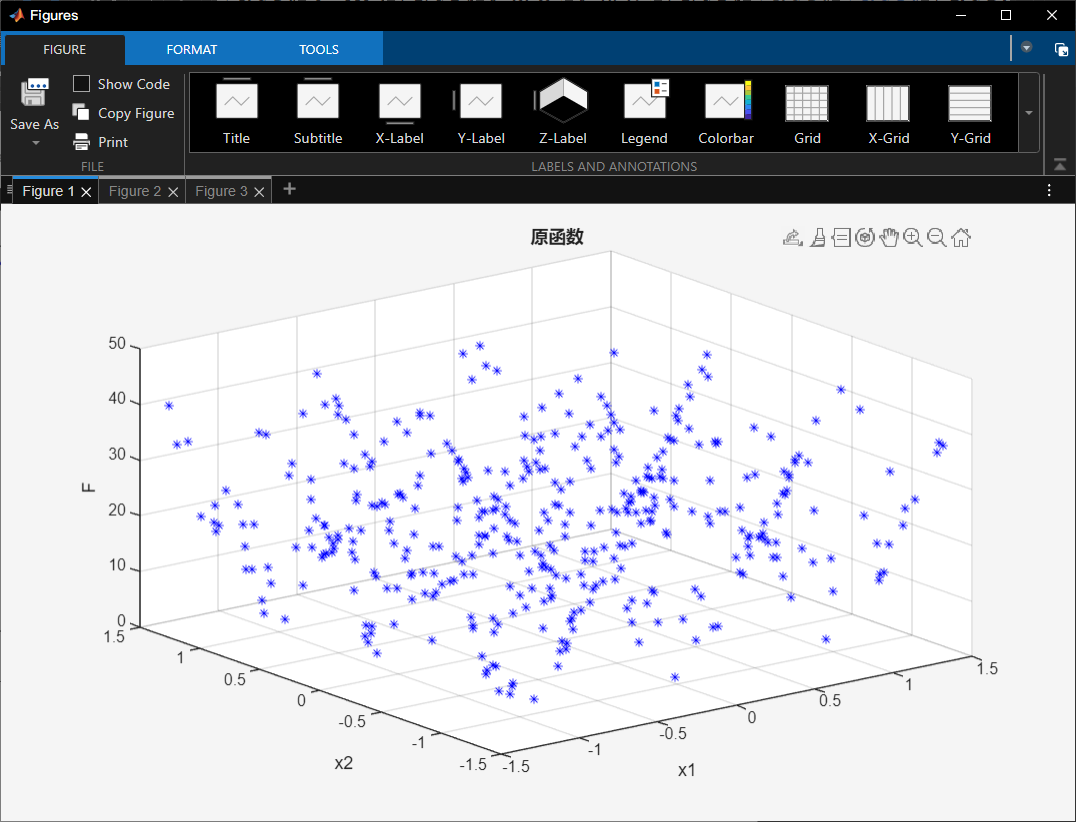
mesh(x1\_mesh, x2\_mesh, F\_mesh - v\_rbe); % 误差图

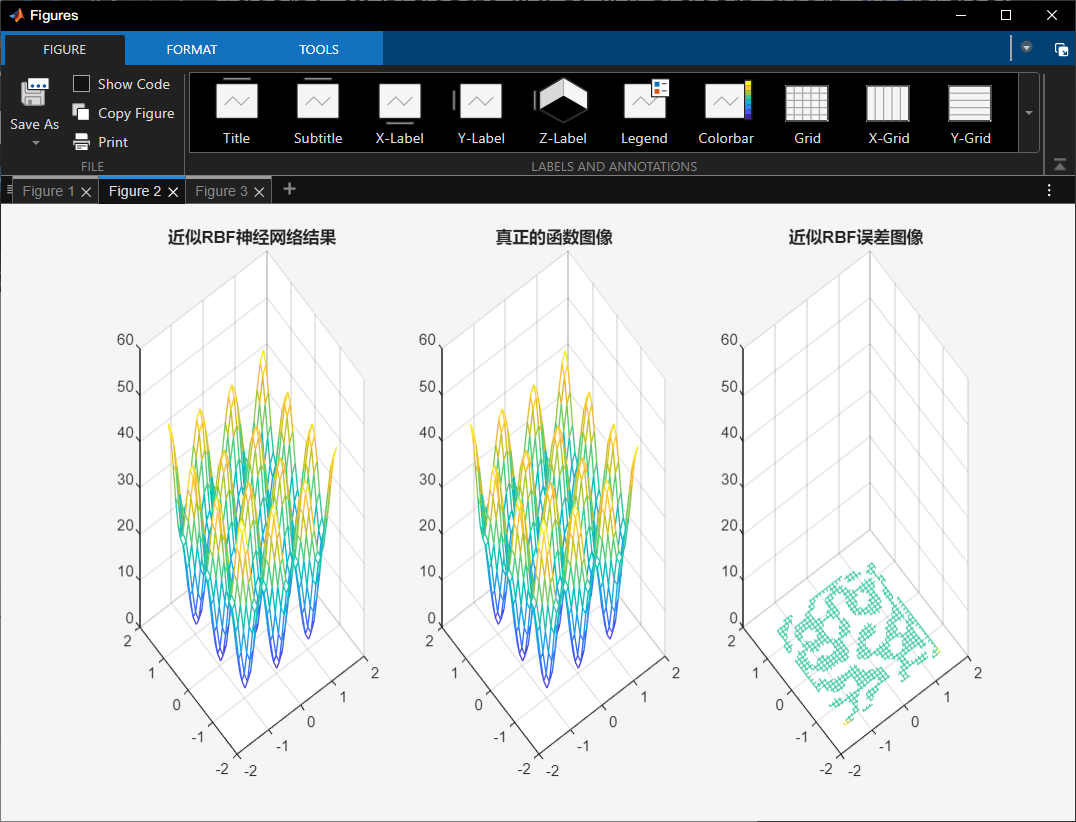
zlim([0, 60])

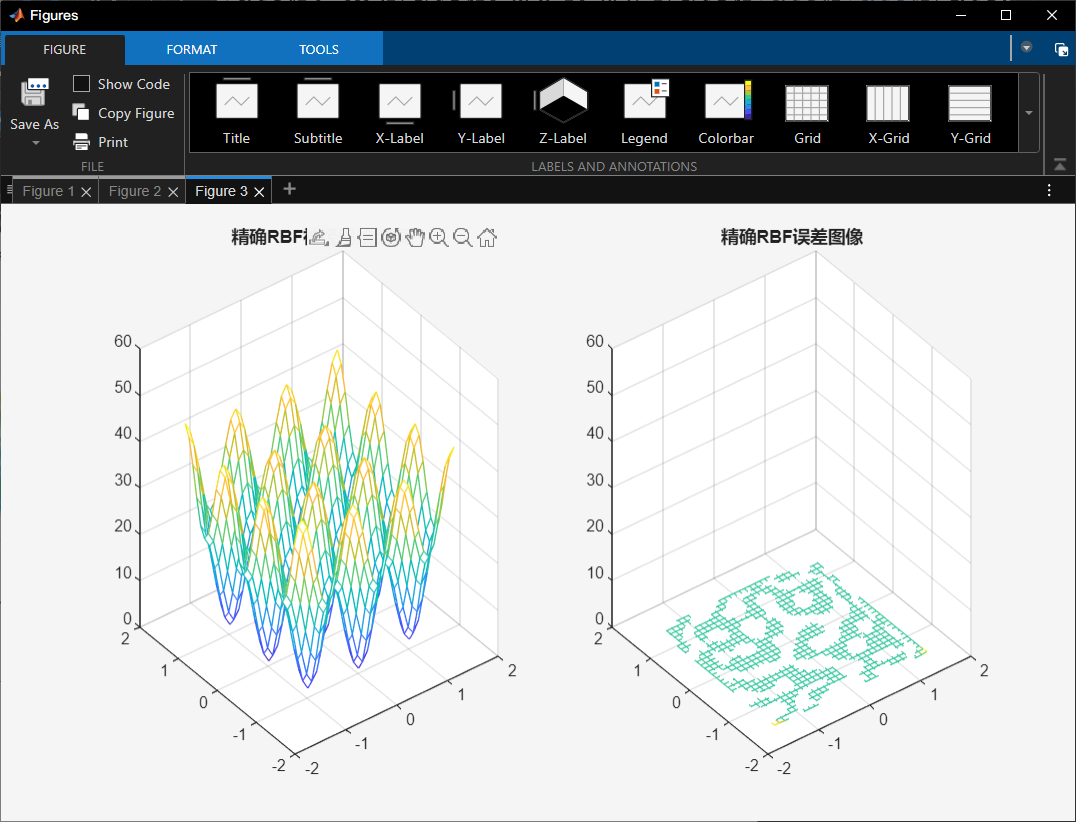
title('精确RBF误差图像')

运行结果如下：









结论：

可以看到近似RBF网络能够对数据进行合理的拟合，但可能无法完全逼近所有训练点，误差稍大。

而精确RBF网络会严格通过所有训练点，误差很小，几乎为0，但是有可能会出现过拟合的现象。