一．问题重述

大气污染防治对于生态环境和人民生命健康安全具有重大意义。在实际生产生活中，会产生众多大气污染物，通过建立空气污染预测模型可以提前预测空气污染状况，使相关部门能够提前采取行动，减少空气污染造成的危害[1]。但目前使用的 WRF-CMAQ 预报模型受到实际气象条件、污染物排放种类以及二次污染物产生机理不明确等因素限制，导致其预报结果不够准确，所以需要结合空气质量监测站的实测数据进行二次建模，对 CMAQ 模型的预报结果进行优化[2]。

要求根据给出的监测站实测数据，结合相关技术标准和实际情况建立模型研究以下问题，同时需要注意处理数据为空值或者异常值的情况。

问题 1. 根据相关的计算公式和规则，结合实测数据计算检测点 A 在 2020 年 8 月 25 日至 8 月 28 日之间，每天的环境空气质量指数(AQI)和首要污染物信息。

问题 2. 假设某地区的污染物排放状况没有发生变化，当地的 AQI 数值也会因为气象条件不同而产生变化，例如气象条件适合污染物扩散或沉降时，当地 AQI 会发生下降。要求使用相关数据，根据污染物浓度变化规律，对其中展现的气象条件进行分类并描述特征。

问题 3. 建立一个同时适用于三个监测点的二次预报数学模型来预测未来三天内每一天的污染物数值，并且要尽可能减小 AQI 预测值的最大误差，以及对于首要污染物的预测精度尽可能高。

问题 4. 在实际情况中，往往相邻地区的污染物浓度数值会有一定关联，多个区域协同预报可以提高预报的准确性。要求建立一个包含四个监测点的协同预报模型，使模型预测结果中 AQI 预测值的最大相对误差尽可能减小，对首要污染物的预测准确度尽可能高。

二．问题分析

问题 1：根据附录中给出的计算公式和相关规则，编写 MATLAB 程序，然后将相应数据代入计算即可得出结果。

问题 2：由于监测站实测数据存在空值和异常数据情况，所以先进行预处理对数据矩阵进行平滑和填充。对于空值(NA)采用查值替换法处理，使用相邻数据对空值进行替换，相邻两天的数据相关性较高，替换后对模型预测结果影响较小。对于异常数据采用箱线图法进行插值。对预处理后的数据进行计算得到对应的 AQI 数据 ，导入 SOM 神经网络模型进行聚类，假定有四种分类，随后对一次预报的气息数据进行分类，两者相结合即可得到气象条件分类情况。

问题 3：二次预报修正模型的任务就是修正一次预报结果与实际值之间的误差，根据一次预报的误差，使用神经网络模型，得到误差变化规律，随后将其应用到一次预报当中，减少一次预报的误差。

问题 4：四个监测站要建立一个区域协同模型，可以将四个监测站的数据进行拟合，将拟合后的数据替换掉预报数据，输入问题 3 中的二次预报修正模型，检验拟合模型能否减少误差。

三．模型假设

1. 假设污染物排放状况不变，即只有天气等因素对 AQI 产生影响。

2. 使用 MATLAB 进行计算时，对有效数字取舍的误差忽略。

四．符号说明

符号	解释
$IAQI_P$	污染物P的空气质量分指数
C_P	污染物P的质量浓度值
BP_{Hi}, BP_{Lo}	与C_P相近的污染物浓度限值的高位值与低位值
$IAQI_{Hi}$, $IAQI_{Lo}$	与BP_{Hi},BP_{Lo}对应的空气质量分指数。

五．模型建立与求解

5.1 问题一的求解

要得到空气质量指数(AQI)首先需要计算各项污染物的空气质量分指数(IAQI)，其计算公式为：

$$\mathrm{IAQI_P} = \frac{\mathrm{IAQI_{Hi}} - \mathrm{IAQI_{Lo}}}{\mathrm{BP_{Hi}} - \mathrm{BP_{Lo}}} \cdot (\mathrm{C_P} - \mathrm{BP_{Lo}}) + \mathrm{IAQI_{Lo}}$$

经过查表可以得知各项污染物浓度限制及其对应的 IAQI 数值。计算得出各项污染物 IAQI 数据后，选取其中最大值者即为当天的首要污染物，其计算规则如下：

$$\mathrm{AQI} = \max\{\mathrm{IAQI_{SO_2}}, \mathrm{IAQI_{NO_2}}, \mathrm{IAQI_{PM_{10}}}, \mathrm{IAQI_{PM_{2.5}}}, \mathrm{IAQI_{O_3}}, \mathrm{IAQI_{CO}}\}$$

最终计算结果如下表：

监测日期	地点	AQI 计算	
		AQI	首要污染物
2020/8/25	监测点 A	65	O_3
2020/8/26	监测点 A	46	O_3
2020/8/27	监测点 A	108	O_3
2020/8/28	监测点 A	137	O_3

5.2 问题二的求解

5.2.1 数据预处理

根据附件 1 中的数据，首先处理空值的情况。对于空值采用查值替换法处理，具体方法为，先找到 NA 值，再使用同列的前一个数据进行替换。

然后处理异常值的情况。对于异常值采用箱线图法进行平滑处理，简单箱线图由五部分组成，分别是最小值、中位数、最大值和两个四分位数。第一四分位数 Q1 又称“下四分位数”，等于该样本中所有数值由小到大排列后第 25%的数字。中位数 F 又称第二四分位数(Q2)，又称“中位数”，等于该样本中所有数值由小到大排列后第 50%的数字。第三四分位数又称“上四分位数”，等于该样本中所有数值由小到大排列后第 75%的数字。其中 QR=Q3-Q1。

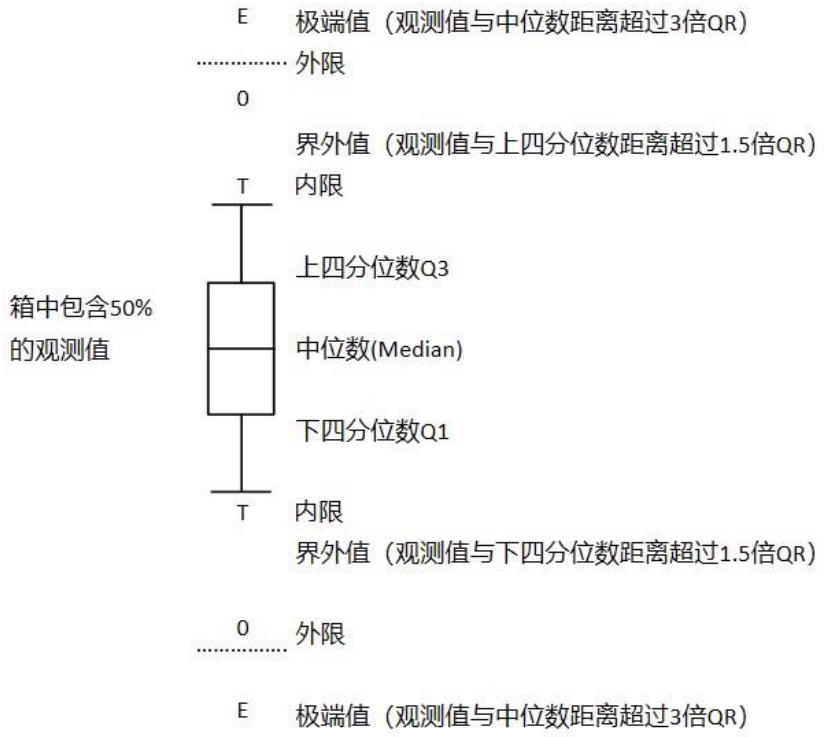

图 1 简单箱线图

5.2.2 SOM 自组织神经网络聚类

自组织映射神经网络可以对数据进行无监督聚类。SOM 本质上是一种有输入层和隐藏层的神经网络，隐藏层中的一个节点代表一个类。训练神经网络时采用竞争学习的方式，每个输入的样例在隐藏层中找到一个和它最匹配的节点，称为它的激活节点。随后用随机梯度下降法更新激活节点的参数。同时和激活节点临近的点也根据它们距离激活节点的远近而适当地更新参数[3]。

SOM 模型的一个特点是隐藏层的节点具有拓扑关系，二维拓扑关系会形成一个平面。

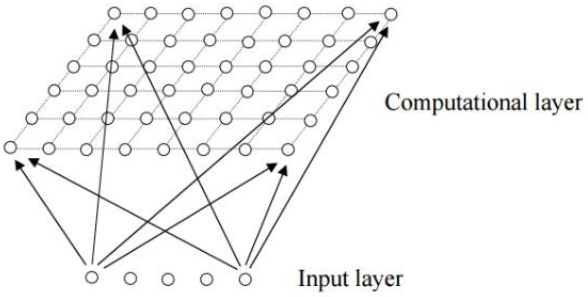

图 2 二维拓扑关系

SOM 可以将任意维度的输入离散化到二维离散空间上，Computation layer 里面的节点与 Input layer 的节点是全连接的[4]。

随后开始计算过程：

1. 初始化：每个节点随机初始化参数，每个节点的参数个数与输入维度相同。

2. 对每一个输入数据找到与其最匹配的节点。假设输入 D 维数据，即 $X = \{x_i, i = 1, \ldots, D\}$,那么判别函数可以为欧几里得距离：

$$d_j(x) = \sum_{i=1}^{D} (x_i - w_{ji})^2$$

3. 找到激活节点 I(x)之后，也要更新和它临近的节点。令S_{ij}表示节点 i 和 j 之间的距离，对于 I(x)临近的节点，分配一个更新权重：

$$T_{j,I(x)} = \exp(-S_{j,I(x)}^2 / 2\sigma^2)$$

简单来说，临近节点根据距离的远近，更新程度要打折扣。

4. 更新节点参数，按照梯度下降法更新：

$$\Delta w_{ji} = \eta(t) \cdot T_{j,I(x)}(t) \cdot (x_i - w_{ji})$$

迭代直至收敛。

5.2.3 计算结果

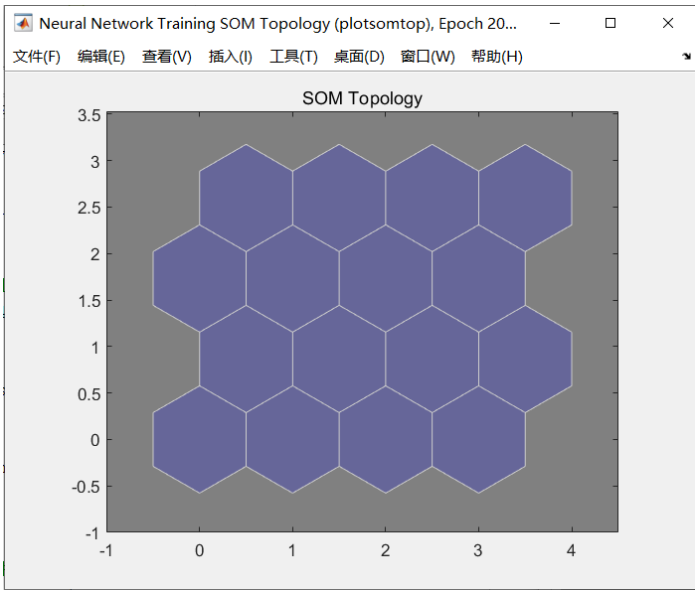

图 3 共 16 个计算层节点

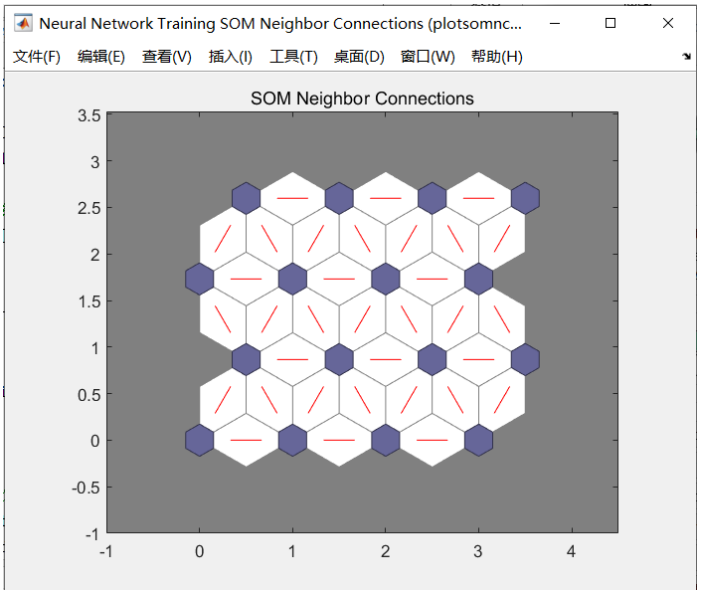

图 4 节点之间的联系

各个节点之间的相关程度通过颜色划分，颜色越浅表示相关性越高，颜色越深表示相关性越低。如图所示，左下为 1 号节点，以此排号右上为 16 号节点，举例说明：1 号节点与 2 号节点之间为浅色，说明这两个节点更加相似，很可能为同一分类，查阅计算得出的分类表格可知，1 号节点为 2 类，2 号节点也为 2 类。同理，6 号节点与 11 号节点之间为深色，查表可知 6 号节点为 1 类，11 号节点为 2 类，两个节点为不同分类。

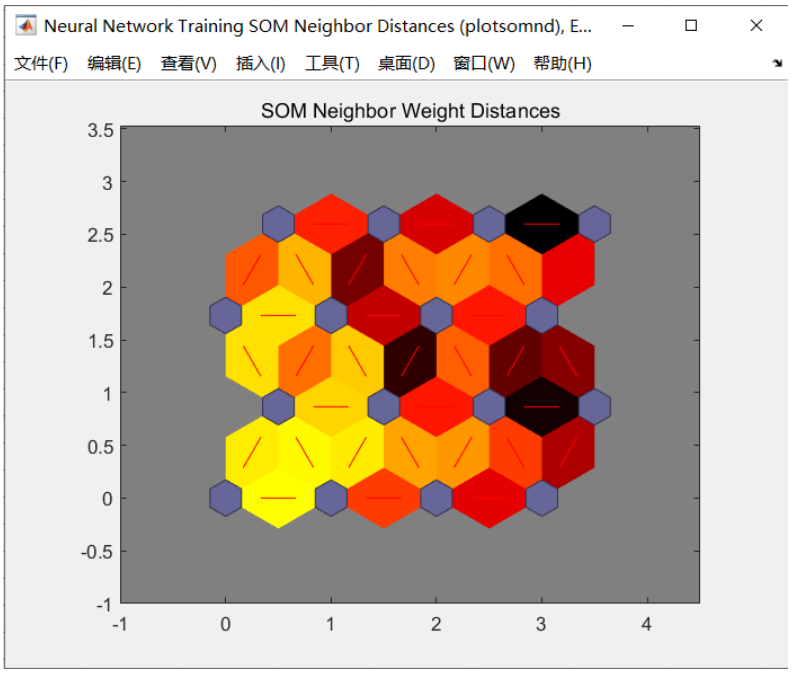

图 5 节点之间的相关程度

71	2	2
72	1	3
73	1	6
74	2	2
75	2	1
76	1	6
77	2	2

图 6 部分分类结果表

根据分类结果与监测点实测数据对比，可以看出当风速较高、湿度较高时，有利于污染物扩散，AQI 数值会下降。

5.3 问题三的求解

5.3.1 基于遗传算法优化的 BP 神经网络结构

网络结构图如下所示：

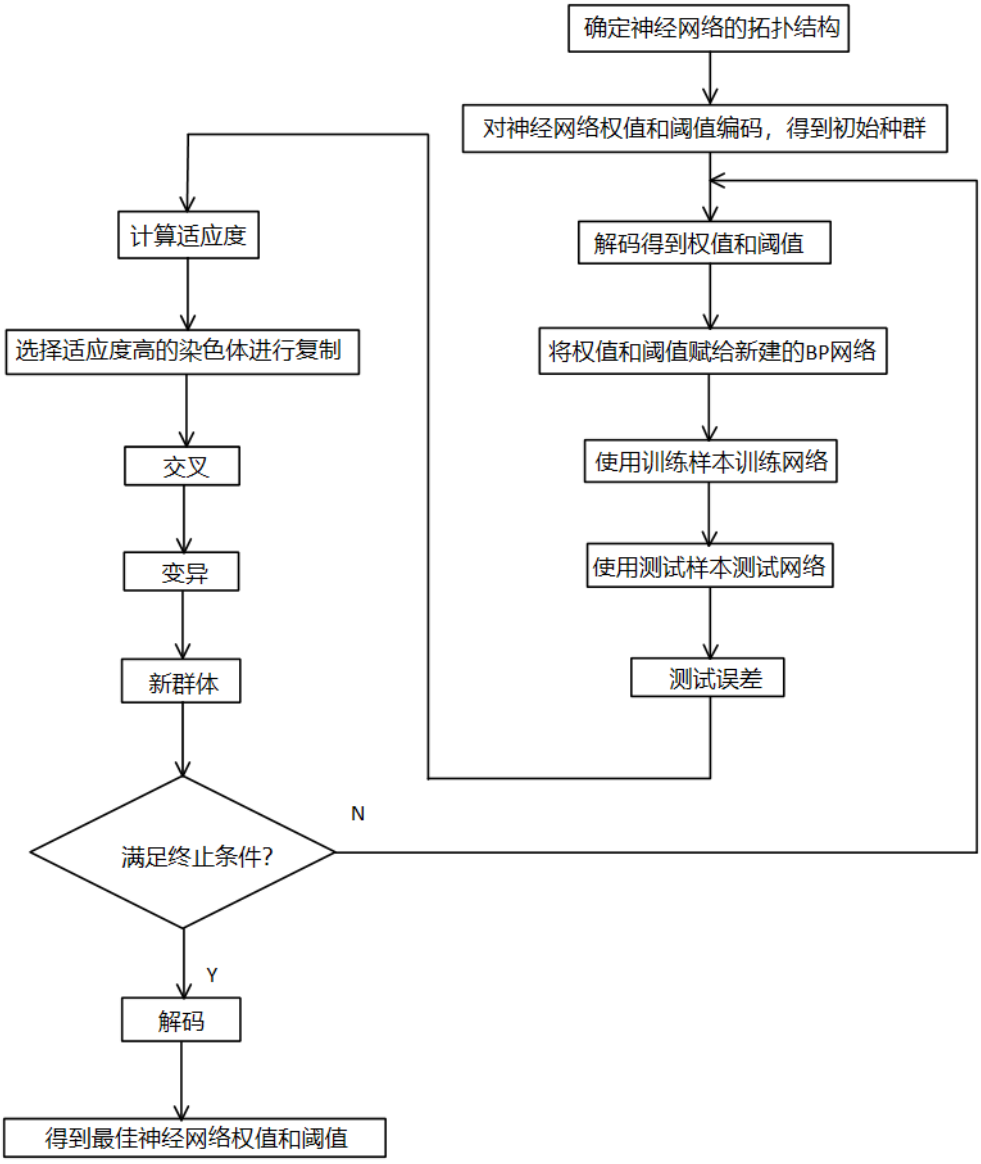

图 7 基于遗传算法优化的 BP 神经网络算法结构图

5.3.2 神经网络算法实现

BP 神经网络算法的实现主要由以下几步[5]：

1. 网络创建

   BP 网络结构的确定有以下两条比较重要的指导原则：

   (1) 对于一般的模式识别问题，三层网络可以很好地解决问题。

   (2) 在三层网络中，隐含层神经网络个数n_2和输入层神经元个数n_1之间有近似关系：

$$n_2 = 2 \times n_1 + 1$$

神经网络的隐含层神经元传递函数采用 S 型正切函数 tansig()，输出层神经元传递函数采用 S 型对数函数 logsig()，这是因为输出模式为 0-1，正好满足网络的输出要求。

2. 网络训练和测试

   网络训练是一个不断修正权值和阈值的过程，通过训练使网络的输出误差越来越小。训练函数 trainlm()可以利用 Levenberg-Marquardt 算法对网络进行训练。

   网络训练后，还需要对网络进行测试。

5.3.3 遗传算法实现

遗传算法优化 BP 神经网络是用遗传算法来优化 BP 神经网络的初始权值和阈值，使优化后的 BP 神经网络能够更好的进行样本预测，遗传算法优化主要包括种群初始化、适应度函数、选择算子、交叉算子和变异算子。

1. 种群初始化

   个体编码采用二进制编码，每个个体均为一个二进制串，由输入层与隐含层连接权值、隐含层阈值、隐含层与输出层连接权值、输出层阈值四部分组成，每个权值和阈值使用 M 位二进制编码，将所有权值和阈值的编码连接起来即为一个个体的编码。

2. 适应度函数

   要求预测值与期望值的残差尽可能小，选择预测样本的预测值与期望值的误差矩阵的范数作为目标函数的输出。

3. 选择算子

   选择算子采用随机遍历抽样。

4. 交叉算子

   采用单点交叉算子。

5. 变异算子

   变异以一定概率产生变异基因数，随机选出变异基因。

六．模型评价与推广

优点：

1. 模型对于不同气象条件下污染物浓度变化规律反映较为清晰，数据分类较为直观。

2. 算法有较大推广价值。

缺点：

1. 采用的神经网络模型准确度和预测稳定性有待提升。

2. 模型对于气象数据的利用不够充分。

七．参考文献

[1]姚文强. 基于数值预报的空气质量预测模型的研究[D].浙江理工大学,2017.

[2]李昊. 基于机器学习的兰州市空气质量预报方法研究[D].兰州大学,2017.

[3]景华,陈世平.融合 SOM 和多目标寻优的 Web 服务组合优化方法[J].小型微型计算机系统,2021,42(09):1810-1817.

[4]Cai Ying,Wang Xu,Xiong LiRan,Ahmed Syed Hassan. Difference Analysis of Regional Economic Development Based on the SOM Neural Network with the Hybrid Genetic Algorithm[J]. Computational Intelligence and Neuroscience,2021,2021:

[5]武晨晨,苗雾,祝佳楠,张文惠.遗传算法优化 BP 神经网络的电能质量预测预警研究[J].电工电气,2021(09):18-22.

八．附录

```
clear
clc
A=xlsread('C:\Users\WR\Desktop\A1\A1\附件1 监测点A空气质量预报基础数据.xlsx',3);
Cp=zeros(100,2);
for i=1:100
    Cp(i,1)=A(i,5);
end

BP=[0,100,160,215,265,800];
IAQI=[0,50,100,150,200,300];
for i=1:100
    for j=1:5
     if Cp(i,1)>BP(j)&&Cp(i,1)<=BP(j+1)
      BPhi=BP(j+1);
      BPlo=BP(j);
      IAQIhi=IAQI(j+1);
      IAQIlo=IAQI(j);
      IAQIp=(IAQIhi-IAQIlo)/(BPhi-BPlo)*(Cp(i,1)-BPlo)+IAQIlo;
      Cp(i,2)=round(IAQIp);
      break
     end
    end
end
```

```
%AS2为使用excel导入的table数据，程序开始运行前需要手动将表A的数据导入到Matlab中
table_1=AS2(1:819,3:end);     %数据预处理，使用相邻元素取代NA值
data=table2array(table_1);
new_data=findNaN(data);
new_data_1=new_data;

%异常值处理
%采用箱线法
[m,n]=size(new_data);
w1=round(m/4);  %第一四分位位置，
%m1=m/2%中位数位置，
w3=round(3*m/4);%第三四分位位置
```

```
%将所有异常值处理，最后new_data更新为一个处理好异常值的新矩阵
for j=1:n
    b11=new_data(:,j);
    [a1,b1]=sort(b11);%[a,b]=sort(x);是从小到大排列，a是排序后结果，b是a结果中各元素的原始位置。
    q11=a1(w1,1);    %第一四分位数
    q13=a1(w3,1);    %第三四分位数
    qr1=q13-q11;       %四分位距
    sx1=q13+1.5*qr1;   %上限
    xx1=q11-1.5*qr1;   %下限
    ycz1=[];%正常值矩阵
    ycz2=[];%异常值矩阵
    s1=1;
    s2=1;
    for i=1:m
        if b11(i,1)<=sx1&&b11(i,1)>=xx1   %求正常值矩阵
            ycz1(s1,1)=b11(i,1);
            s1=s1+1;
        else
            ycz2(s2,1)=b11(i,1);     %求异常值举证
            ycz2(s2,2)=i;
            s2=s2+1;
        end
    end
    average_1=round(mean(ycz1,1));  %四舍五入取整
```

```
    for i=1:m       %使用正常值部分的均值将异常值替换
        if b11(i,1)>sx1||b11(i,1)<xx1
            b11(i,1)=average_1;
        end
    end
    new_data(i,j)=b11(i,1);
    j=j+1;
end
%以上就是对监测点A空气质量预报基础数据的预处理代码


Cp=zeros(819,2);          %求取臭氧的空气质量指标IAQI
for i=1:819
    Cp(i,1)=new_data_1(i,5);
end
```

```
BP=[0,100,160,215,265,800];
IAQI=[0,50,100,150,200,300];
for i=1:819
    for j=1:5
     if Cp(i,1)>BP(j)&&Cp(i,1)<=BP(j+1)
      BPhi=BP(j+1);
      BPlo=BP(j);
      IAQIhi=IAQI(j+1);
      IAQIlo=IAQI(j);
      IAQIp=(IAQIhi-IAQIlo)/(BPhi-BPlo)*(Cp(i,1)-BPlo)+IAQIlo;
      Cp(i,2)=round(IAQIp);
      break
     end
    end
end                          %Cp为臭氧为首要污染条件下的IAQI数据
```

```
%根据求取的IAQI将空气气象状况标注分类，分类等级再考虑分为几类，说明这几类大概的气象条件
classes=Cp(:,2);          %将气象条件分类
classes_min=min(classes);
classes_max=max(classes);
classes_q=(classes_max-classes_min)/4;
for i=1:m
    if classes(i,1)<classes_min+round(classes_q)
        classes(i,1)=1;
    elseif classes(i,1)<classes_min+round(classes_q*2)
        classes(i,1)=2;
    elseif classes(i,1)<classes_min+round(classes_q*3)
        classes(i,1)=3;
    else
        classes(i,1)=4;
    end
end
```

```
88 —    classes=classes';
89
90
91      %数据归一化并进行矩阵转置
92 —    new_data=mapminmax(new_data);
93 —    new_data=new_data';
94      %训练集——800个样本
95 —    P_train=new_data(:,1:700);
96 —    T_train=classes(:,1:700);
97      %测试集——19个样本
98 —    P_test=new_data(:,701:end);
99 —    T_test=classes(:,701:end);
100
101     %SOFM神经网络(一种自组织的神经网络）的创建、训练及仿真测试
102 —    net=newsom(P_train,[4 4]);
103
104     %设置训练参数
105 —    net.trainParam.epochs=200;
106
107     %训练网络
108 —    net=train(net,P_train);
```

```
110     %仿真测试
111
112     %训练集测试        %数据量有800个，太大了暂时无需测试。只需对测试集中的数据测试，看分类效果即可。
113 —    t_sim_sofm_1=sim(net,P_train);
114 —    T_sim_sofm_1=vec2ind(t_sim_sofm_1);
115
116     %测试集测试
117 —    t_sim_sofm_2=sim(net,P_test);
118 —    T_sim_sofm_2=vec2ind(t_sim_sofm_2);
119
120     %sofm神经网络对测试集分类结果
121 —    result_sofm_1=[T_train' T_sim_sofm_1'];
122 —    result_sofm_2=[T_test' T_sim_sofm_2'];
```

```
4       %% 加载神经网络的训练样本 测试样本每列一个样本 输入P 输出T，T是标签
5       %样本数据就是前面问题描述中列出的数据
6       %epochs是计算时根据输出误差返回调整神经元权值和阀值的次数
7       load data
8       % 初始隐层神经元个数
9       hiddennum=31;
10      % 输入向量的最大值和最小值
11      threshold=[0 1;0 1;0 1;0 1;0 1;0 1;0 1;0 1;0 1;0 1;0 1;0 1;0 1;0 1;0 1];
12      inputnum=size(P,1); % 输入层神经元个数
13      outputnum=size(T,1); % 输出层神经元个数
14      w1num=inputnumhiddennum; % 输入层到隐层的权值个数
15      w2num=outputnumhiddennum;% 隐层到输出层的权值个数
16      N=w1num+hiddennum+w2num+outputnum; %待优化的变量的个数
```

```
%% 定义遗传算法参数
NIND=40; %个体数目
MAXGEN=50; %最大遗传代数
PRECI=10; %变量的二进制位数
GGAP=0.95; %代沟
px=0.7; %交叉概率
pm=0.01; %变异概率
trace=zeros(N+1,MAXGEN); %寻优结果的初始值

FieldD=[repmat(PRECI,1,N);repmat([-0.5;0.5],1,N);repmat([1;0;1;1],1,N)]; %区域描述器
Chrom=crtbp(NIND,PRECI*N); %初始种群
```

```
%% 优化
gen=0; %代计数器
X=bs2rv(Chrom,FieldD); %计算初始种群的十进制转换
ObjV=Objfun(X,P,T,hiddennum,P_test,T_test); %计算目标函数值
while gen<MAXGEN
fprintf('%d\n',gen)
FitnV=ranking(ObjV); %分配适应度值
SelCh=select('sus',Chrom,FitnV,GGAP); %选择，随机遍历抽样
SelCh=recombin('xovsp',SelCh,px); %重组，单点交叉
SelCh=mut(SelCh,pm); %变异
X=bs2rv(SelCh,FieldD); %子代个体的十进制转换
ObjVSel=Objfun(X,P,T,hiddennum,P_test,T_test); %计算子代的目标函数值
[Chrom,ObjV]=reins(Chrom,SelCh,1,1,ObjV,ObjVSel); %重插入子代到父代，得到新种群，注意插入后新种群与老种群的规模
%代沟只是说选择子种群的时候是选择95%的个体作为待插入的子种群
%1，父代chrome和子代selch中的子种群个数都是1，1，基于适应度的选择，子代代替父代中适应度最小的个体
X=bs2rv(Chrom,FieldD);%插入完成后，重新计算个体的十进制值
gen=gen+1; %代计数器增加
%获取每代的最优解及其序号，Y为最优解,I为个体的序号
[Y,I]=min(ObjV);%Objv是目标函数值，也就是预测误差的范数
trace(1:N,gen)=X(I,:); %记下每代个体的最优值，即各个权重值
trace(end,gen)=Y; %记下每代目标函数的最优值，即预测误差的范数
end
```

```
%% 画进化图
figure(1);
plot(1:MAXGEN,trace(end,:));
grid on
xlabel('遗传代数')
ylabel('误差的变化')
title('进化过程')
bestX=trace(1:end-1,end);%注意这里仅是记录下了最优的初始权重，训练得到的最终的网络的权值并未记录下来
bestErr=trace(end,end);
fprintf(['最优初始权值和阈值:\nX=',num2str(bestX'),'\n最小误差err=',num2str(bestErr),'\n'])
```

```
%% 比较优化前后的训练&测试
callbackfun

子函数：
function Obj=Objfun(X,P,T,hiddennum,P_test,T_test)
%% 用来分别求解种群中各个个体的目标值
%% 输入
% X：所有个体的初始权值和阈值
% P：训练样本输入
% T：训练样本输出
% hiddennum：隐含层神经元数
% P_test:测试样本输入
% T_test:测试样本期望输出
%% 输出
% Obj：所有个体的预测样本的预测误差的范数
%这个函数的目的就是用种群中所有个体所代表的神经网络的初始权重值去进行网络的训练，训练次数是1000次，然
%后得出所有个体作为初始权重训练网络1000次所得出的预测误差，也就是这里的obj，返回到原函数中，迭代maxgen=50次
%记录下每一代的最优权重值和最优目标值(最小误差值)
[M,N]=size(X);
Obj=zeros(M,1);
for i=1:M%M是40，即有40个个体，每个个体就是一次初始权重，在BPfun中用每个个体作为初始值去进行了1000次的训练
Obj(i)=BPfun(X(i,:),P,T,hiddennum,P_test,T_test);%Obj是一个40*1的向量，每个值对应的是一个个体作为初始权重值
%网络1000次得出来的误差
end
```

```
function err=BPfun(x,P,T,hiddennum,P_test,T_test)
%% 训练&测试BP网络
%% 输入
% x：一个个体的初始权值和阈值
% P：训练样本输入
% T：训练样本输出
% hiddennum：隐含层神经元数
% P_test:测试样本输入
% T_test:测试样本期望输出
%% 输出
% err：预测样本的预测误差的范数
%用每一个个体的初始权值去训练1000次
inputnum=size(P,1); % 输入层神经元个数
outputnum=size(T,1); % 输出层神经元个数
%% 新建BP网络
%神经网络的隐含层神经元的传递函数采用S型正切函数tansing（），输出层神经元的函数采用S型对数函数logsig（）
net=newff(minmax§,[hiddennum,outputnum],{'tansig','logsig'},'trainlm');
```

```
%% 设置网络参数：训练次数为1000，训练目标为0.01，学习速率为0.1
net.trainParam.epochs=1000;%允许最大训练次数，实际这个网络训练到迭代次数是3时就已经到达要求结束了
net.trainParam.goal=0.01;%训练目标最小误差，应该是mean square error， 均方误差，就是网络输出和目标值
LP.lr=0.1;%学习速率学习率的作用是不断调整权值阈值。w(n+1)=w(n)+LP.lr*(d(n)-y(n))x(n),d(n)是期望的相
%量化的实际响应，x(n)是输入向量，如果d(n)与y(n)相等的话，则w(n+1)=w(n),这里是指输入到隐含层的调整方式
%隐含层到输出层的调整 Iout(j)=1/(1+exp(-I(j)));
%dw2=eIout;db2=e';w2=w2_1+xitedw2';e是错误值
%b2=b2_1+xitedb2';xite是学习率
%对于traingdm等函数建立的BP网络，学习速率一般取0.01-0.1之间。
net.trainParam.show=NaN;
% net.trainParam.showwindow=false; %高版MATLAB
%% BP神经网络初始权值和阈值
w1num=inputnumhiddennum; % 输入层到隐层的权值个数
w2num=outputnumhiddennum;% 隐层到输出层的权值个数
w1=x(1:w1num); %初始输入层到隐层的权值
B1=x(w1num+1:w1num+hiddennum); %初始隐层阈值
w2=x(w1num+hiddennum+1:w1num+hiddennum+w2num); %初始隐层到输出层的阈值
B2=x(w1num+hiddennum+w2num+1:w1num+hiddennum+w2num+outputnum); %输出层阈值
net.iw{1,1}=reshape(w1,hiddennum,inputnum);%输入到隐藏层的权重
net.lw{2,1}=reshape(w2,outputnum,hiddennum);%隐藏到输出层的权重
net.b{1}=reshape(B1,hiddennum,1);
net.b{2}=reshape(B2,outputnum,1);
```

```
%% 训练网络以
net=train(net,P,T);
%% 测试网络
disp(['1、使用随机权值和阈值 '])
disp('测试样本预测结果：')
Y1=sim(net,P_test)%测试样本的网络仿真输出
err1=norm(Y1-T_test); %测试样本的仿真误差
err11=norm(sim(net,P)-T); %训练样本的仿真误差
disp(['测试样本的仿真误差:',num2str(err1)])
disp(['训练样本的仿真误差:',num2str(err11)])

%% 使用遗传算法
%% 使用优化后的权值和阈值，利用遗传算法得出来的最优的初始权重和阈值去进行网络的初始化
inputnum=size(P,1); % 输入层神经元个数
outputnum=size(T,1); % 输出层神经元个数
%% 新建BP网络
net=newff(minmax(P),[hiddennum,outputnum],{'tansig','logsig'},'trainlm');
%% 设置网络参数：训练次数为1000，训练目标为0.01，学习速率为0.1
net.trainParam.epochs=1000;
net.trainParam.goal=0.01;
LP.lr=0.1;
```

```
%% BP神经网络初始权值和阈值
w1num=inputnumhiddennum; % 输入层到隐层的权值个数
w2num=outputnumhiddennum;% 隐层到输出层的权值个数
w1=bestX(1:w1num); %初始输入层到隐层的权值
B1=bestX(w1num+1:w1num+hiddennum); %初始隐层阈值
w2=bestX(w1num+hiddennum+1:w1num+hiddennum+w2num); %初始隐层到输出层的阈值
B2=bestX(w1num+hiddennum+w2num+1:w1num+hiddennum+w2num+outputnum); %输出层阈值
net.iw{1,1}=reshape(w1,hiddennum,inputnum);
net.lw{2,1}=reshape(w2,outputnum,hiddennum);
net.b{1}=reshape(B1,hiddennum,1);
net.b{2}=reshape(B2,outputnum,1);
%% 训练网络以
net=train(net,P,T);
%% 测试网络
disp(['2、使用优化后的权值和阈值'])
disp('测试样本预测结果：')
Y2=sim(net,P_test)%测试样本的仿真输出
err2=norm(Y2-T_test);%测试样本的仿真误差
err21=norm(sim(net,P)-T);%训练样本的仿真误差
disp(['测试样本的仿真误差:',num2str(err2)])
disp(['训练样本的仿真误差:',num2str(err21)])
```