程序 safe

```
function  [M]=safe(c,t,f,u,v,w,k1,k2,k3)%输入 c,t,f 三种不同的风险因素,u,v,w 为各指标权向量，为横向量。k1,k2,k3 为三要素的相对重要程度,默认相同
if nargin<7
   k1=1/3;
   k2=1/3;
   k3=1/3;
end
m=size(c,2);%取出 c 的列数,c,t,f 同矩阵
n=size(c,1);%取出 c 的行数
k=1;
eck=ones(1,n);%保留 ei 的值
etk=ones(1,n);%保留 ei 的值
efk=ones(1,n);%保留 ei 的值
while k<n+1
   eckm=-1/log(m)*c(k,:).*log(c(k,:));
   eckm(find(isnan(eckm)==1)) = 0;%将可能出现的 0*ln(0)的值设成 0
   eck(1,k)=sum(eckm);
   etkm=-1/log(m)*t(k,:).*log(t(k,:));
   etkm(find(isnan(etkm)==1)) = 0;%将可能出现的 0*ln(0)的值设成 0
   etk(1,k)=sum(etkm);
   efkm=-1/log(m)*f(k,:).*log(f(k,:));
   efkm(find(isnan(efkm)==1)) = 0;%将可能出现的 0*ln(0)的值设成 0
   efk(1,k)=sum(efkm);
   k=k+1;
end

FaiC=1/(n-sum(eck))*(1-eck)%求出Φc 的值
FaiT=1/(n-sum(etk))*(1-etk);%求出Φt 的值
FaiF=1/(n-sum(eck))*(1-efk);%求出Φf 的值
Rc=FaiC*c*u'%各要素风险值
Rt=FaiT*t*v';%各要素风险值
Rf=FaiF*f*w';%各要素风险值
M=k1*Rc+k2*Rt+k3*Rf;%安全风险值
```

BP 程序

```
%下载输入输出数据
load date input output
%随机选择 m 组训练数据和 n 组预测数据
disp('请输入训练数据的数量 m，和预测数据的数量 n');
m=input('m=');
n=input('n=');
k=rand(1,m+n);
[m,n]=sort(k);
input_train=input(n(1:n),:)';
```

```
output_train=output(n(1:n),:)';
input_test=input(n(m+1:m+n),:)';
output_test=output(n(m+1:m+n),:)';
%BP 神经网络构建

net=newff(input_train,output_train,4,{'logsig','logsig'},'trainglm','
learngdm','mse');
%网络参数配置
net.trainparam.epochs=100;
net.trainparam.lr=0.05;
net.trainparm.goal=0.0004;
%bp 神经网络训练
net=train(net,inputn,outputn);
%BP 神经网络预测输出,误差值
an=sim(net,inputn_test)
MSE=mse(output_test-an)
disp('请输入判断矩阵 A');
A=input('A=');
B=sim(net,A)
```

判断矩阵与一致性检验程序

```
disp('请输入判断矩阵 A(n 阶)');
A=input('A=');
n=size(A,1);
x=ones(n,n^2);
y=ones(n,n^2);
m=zeros(1,n^2);
m(1)=1;
i=1;
p=0.0001;
k=1;
while  k>p
    i=i+1;
    x(:,i)=A*y(:,i-1);
    m(i)=max(x(:,i));
    y(:,i)=x(:,i)/m(i);
    k=abs(m(i)-m(i-1));
end
a=sum(y(:,i));
t=m(i)
w=y(:,i)/a
%以下是一致性检验
CI=(t-n)/(n-1);
RI=[0  0  0.52  0.89  1.12  1.26  1.36  1.41  1.46  1.49  1.52  1.54  1.56  1.58
```

```
1.59];
CR=CI/RI(n);
if CR<0.10
CI
CR
disp('此矩阵的一致性可以接受!');
end
```