蚁群算法实现

蚁群算法的基本原理:

1、蚂蚁在路径上释放信息素。

2、碰到还没走过的路口，就随机挑选一条路走。同时，释放与路径长度有关的信息素

3、信息素浓度与路径长度成反比。后来的蚂蚁再次碰到该路口时，就选择信息素浓度较高路径。

4、最优路径上的信息素浓度越来越大。

5、最终蚁群找到最优寻食路径。

在代码中一个旅行商人要拜访全国31个省会城市，需要选择最短的路径，然后我用蚁群算法实现这个过程。

第一步：变量初始化

第二步：将m只蚂蚁放到n个城市上

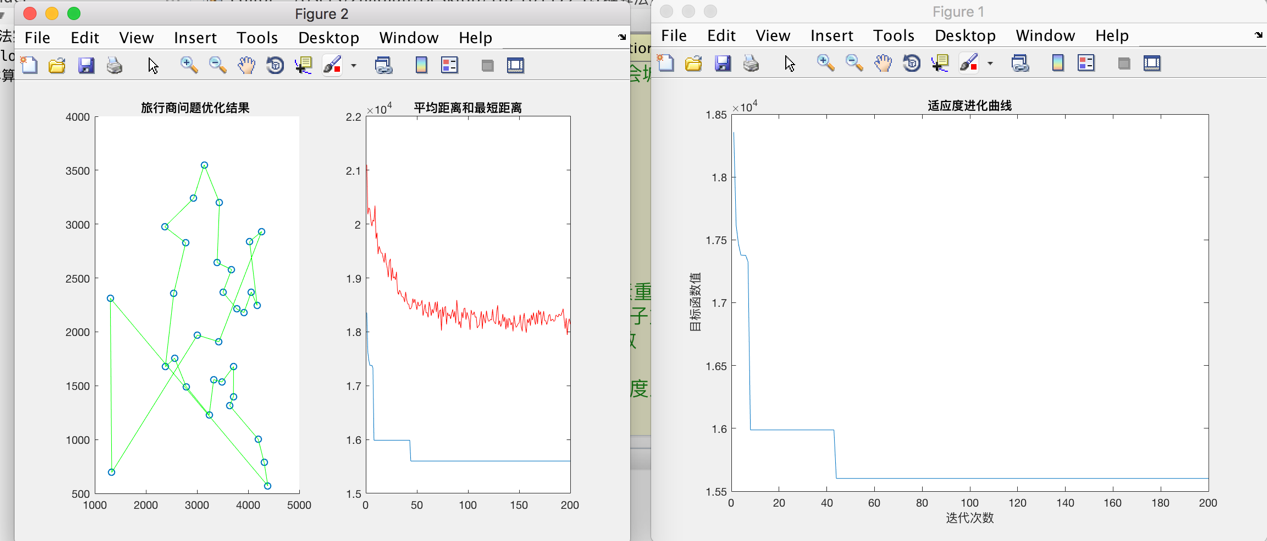
第三步：m只蚂蚁按概率函数选择下一座城市，完成各自的周游

在这边使用两重大循环，外层是城市，内层是蚂蚁。也就是说每次迭代每只蚂蚁向前走一步，而不是一只蚂蚁走完所有城市再开始下一只。

第四步：记录本次迭代最佳路线

第五步：更新信息素

最终的实验结果运行图如下：



总结：

蚁群算法有一下特点：

1、从算法的性质而言，蚁群算法是在寻找一个比较好的局部最优解，而不是强调全局最优解

2、开始时算法收敛速度较快，在随后寻优过程中，迭代一定次数后，容易出现停滞现象

3、蚁群算法对TSP及相似问题具有良好的适应性，无论城市规模大还是小，都能进行有效地求解，而且求解速度相对较快

4、蚁群算法解得稳定性较差，及时参数不变，每次执行程序都有可能得到不同界，为此需要多执行几次，已寻找最佳解。

5、蚁群算法中有多个需要设定的参数，而且这些参数对程序又都有一定的影响，所以选择合适的参数组合在算法设计过程中也非常重要。

代码实现：

clear all; %清除所有变量

close all; %清图

clc ; %清屏

m=50; %% m 蚂蚁个数

Alpha=1; %% Alpha 表征信息素重要程度的参数

Beta=5; %% Beta 表征启发式因子重要程度的参数

Rho=0.1; %% Rho 信息素蒸发系数

NC\_max=200; %%最大迭代次数

Q=100; %%信息素增加强度系数

C=[

1304 2312;

3639 1315;

4177 2244;

3712 1399;

3488 1535;

3326 1556;

3238 1229;

4196 1004;

4312 790;

4386 570;

3007 1970;

2562 1756;

2788 1491;

2381 1676;

1332 695;

3715 1678;

3918 2179;

4061 2370;

3780 2212;

3676 2578;

4029 2838;

4263 2931;

3429 1908;

3507 2367;

3394 2643;

3439 3201;

2935 3240;

3140 3550;

2545 2357;

2778 2826;

2370 2975

]; %%31个省会坐标

%%-------------------------------------------------------------------------

%% 主要符号说明

%% C n个城市的坐标，n×2的矩阵

%% NC\_max 最大迭代次数

%% m 蚂蚁个数

%% Alpha 表征信息素重要程度的参数

%% Beta 表征启发式因子重要程度的参数

%% Rho 信息素蒸发系数

%% Q 信息素增加强度系数

%% R\_best 各代最佳路线

%% L\_best 各代最佳路线的长度

%%=========================================================================

%%第一步：变量初始化

n=size(C,1);%n表示问题的规模（城市个数）

D=zeros(n,n);%D表示完全图的赋权邻接矩阵

for i=1:n

for j=1:n

if i~=j

D(i,j)=((C(i,1)-C(j,1))^2+(C(i,2)-C(j,2))^2)^0.5;

else

D(i,j)=eps; %i=j时不计算，应该为0，但后面的启发因子要取倒数，用eps（浮点相对精度）表示

end

D(j,i)=D(i,j); %对称矩阵

end

end

Eta=1./D; %Eta为启发因子，这里设为距离的倒数

Tau=ones(n,n); %Tau为信息素矩阵

Tabu=zeros(m,n); %存储并记录路径的生成

NC=1; %迭代计数器，记录迭代次数

R\_best=zeros(NC\_max,n); %各代最佳路线

L\_best=inf.\*ones(NC\_max,1); %各代最佳路线的长度

L\_ave=zeros(NC\_max,1); %各代路线的平均长度

while NC<=NC\_max %停止条件之一：达到最大迭代次数，停止

%%第二步：将m只蚂蚁放到n个城市上

Randpos=[]; %随即存取

for i=1:(ceil(m/n))

Randpos=[Randpos,randperm(n)];

end

Tabu(:,1)=(Randpos(1,1:m))';

%%第三步：m只蚂蚁按概率函数选择下一座城市，完成各自的周游

for j=2:n %所在城市不计算

for i=1:m

visited=Tabu(i,1:(j-1)); %记录已访问的城市，避免重复访问

J=zeros(1,(n-j+1)); %待访问的城市

P=J; %待访问城市的选择概率分布

Jc=1;

for k=1:n

if length(find(visited==k))==0 %开始时置0

J(Jc)=k;

Jc=Jc+1; %访问的城市个数自加1

end

end

%下面计算待选城市的概率分布

for k=1:length(J)

P(k)=(Tau(visited(end),J(k))^Alpha)\*(Eta(visited(end),J(k))^Beta);

end

P=P/(sum(P));

%按概率原则选取下一个城市

Pcum=cumsum(P); %cumsum，元素累加即求和

Select=find(Pcum>=rand); %若计算的概率大于原来的就选择这条路线

to\_visit=J(Select(1));

Tabu(i,j)=to\_visit;

end

end

if NC>=2

Tabu(1,:)=R\_best(NC-1,:);

end

%%第四步：记录本次迭代最佳路线

L=zeros(m,1); %开始距离为0，m\*1的列向量

for i=1:m

R=Tabu(i,:);

for j=1:(n-1)

L(i)=L(i)+D(R(j),R(j+1)); %原距离加上第j个城市到第j+1个城市的距离

end

L(i)=L(i)+D(R(1),R(n)); %一轮下来后走过的距离

end

L\_best(NC)=min(L); %最佳距离取最小

pos=find(L==L\_best(NC));

R\_best(NC,:)=Tabu(pos(1),:); %此轮迭代后的最佳路线

L\_ave(NC)=mean(L); %此轮迭代后的平均距离

NC=NC+1 %迭代继续

%%第五步：更新信息素

Delta\_Tau=zeros(n,n); %开始时信息素为n\*n的0矩阵

for i=1:m

for j=1:(n-1)

Delta\_Tau(Tabu(i,j),Tabu(i,j+1))=Delta\_Tau(Tabu(i,j),Tabu(i,j+1))+Q/L(i);

%此次循环在路径（i，j）上的信息素增量

end

Delta\_Tau(Tabu(i,n),Tabu(i,1))=Delta\_Tau(Tabu(i,n),Tabu(i,1))+Q/L(i);

%此次循环在整个路径上的信息素增量

end

Tau=(1-Rho).\*Tau+Delta\_Tau; %考虑信息素挥发，更新后的信息素

%%第六步：禁忌表清零

Tabu=zeros(m,n); %%直到最大迭代次数

end

%%第七步：输出结果

Pos=find(L\_best==min(L\_best)); %找到最佳路径（非0为真）

Shortest\_Route=R\_best(Pos(1),:) %最大迭代次数后最佳路径

Shortest\_Length=L\_best(Pos(1)) %最大迭代次数后最短距离

figure(1)

plot(L\_best)

xlabel('迭代次数')

ylabel('目标函数值')

title('适应度进化曲线')

figure(2)

subplot(1,2,1) %绘制第一个子图形

%画路线图

%%=========================================================================

%% DrawRoute.m

%% 画路线图

%%-------------------------------------------------------------------------

%% C Coordinate 节点坐标，由一个N×2的矩阵存储

%% R Route 路线

%%=========================================================================

N=length(R);

scatter(C(:,1),C(:,2));

hold on

plot([C(R(1),1),C(R(N),1)],[C(R(1),2),C(R(N),2)],'g')

hold on

for ii=2:N

plot([C(R(ii-1),1),C(R(ii),1)],[C(R(ii-1),2),C(R(ii),2)],'g')

hold on

end

title('旅行商问题优化结果 ')

subplot(1,2,2) %绘制第二个子图形

plot(L\_best)

hold on %保持图形

plot(L\_ave,'r')

title('平均距离和最短距离') %标题