

**实验用例：**

用模拟退火算法解决如下 10 个城市的 TSP 问题，该问题最优解为  $f_{opt} = 2.691$ 。

表 1 10 个城市的坐标

城市	X 坐标	Y 坐标	城市	X 坐标	Y 坐标
1	0.6683	0.2536	6	0.2293	0.7610
2	0.6195	0.2634	7	0.5171	0.9414
3	0.4000	0.4439	8	0.8732	0.6536
4	0.2439	0.1463	9	0.6878	0.5219
5	0.1707	0.2293	10	0.8488	0.3609

**编程实现**

用 MATLAB 实现模拟退火算法时，共编制了 5 个 m 文件，分别如下

## 1、swap.m

```
function [ newpath , position ] = swap( oldpath , number )
% 对 oldpath 进行互换操作
% number 为产生的新路径的个数
% position 为对应 newpath 互换的位置
m = length( oldpath ); % 城市的个数
newpath = zeros( number , m );
position = sort( randi( m , number , 2 ) , 2 ); % 随机产生交换的位置
for i = 1 : number
    newpath( i , : ) = oldpath ;
% 交换路径中选中的城市
    newpath( i , position( i , 1 ) ) = oldpath( position( i , 2 ) );
    newpath( i , position( i , 2 ) ) = oldpath( position( i , 1 ) );
end
```

## 2、pathfare.m

```
function [ objval ] = pathfare( fare , path )
% 计算路径 path 的代价 objval
% path 为 1 到 n 的排列，代表城市的访问顺序；
% fare 为代价矩阵，且为方阵。
[ m , n ] = size( path );
objval = zeros( 1 , m );
for i = 1 : m
    for j = 2 : n
        objval( i ) = objval( i ) + fare( path( i , j - 1 ) , path( i , j ) );
    end
    objval( i ) = objval( i ) + fare( path( i , n ) , path( i , 1 ) );
end
```

## 3、distance.m

```
function [ fare ] = distance( coord )
% 根据各城市的距离坐标求相互之间的距离
% fare 为各城市的距离，coord 为各城市的坐标
[~, m] = size( coord );    % m 为城市的个数
fare = zeros( m );
for i = 1 : m              % 外层为行
    for j = i : m          % 内层为列
        fare( i, j ) = ...
            ( sum( ( coord( :, i ) - coord( :, j ) ).^2 ) ) ^ 0.5 ;
        fare( j, i ) = fare( i, j );    % 距离矩阵对称
    end
end
```

## 4、myplot.m

```
function [ ] = myplot( path, coord, pathfar )
% 做出路径的图形
% path 为要做图的路径，coord 为各个城市的坐标
% pathfar 为路径 path 对应的费用
len = length( path );
clf;
hold on;
title( [ '近似最短路径如下，费用为', num2str( pathfar ) ] );
plot( coord( 1, : ), coord( 2, : ), 'ok' );
pause( 0.4 );
for ii = 2 : len
    plot( coord( 1, path( [ ii - 1, ii ] ) ), coord( 2, path( [ ii - 1, ii ] ) ), '-b' );
    x = sum( coord( 1, path( [ ii - 1, ii ] ) ) ) / 2 ;
    y = sum( coord( 2, path( [ ii - 1, ii ] ) ) ) / 2 ;
    text( x, y, [ '(' , num2str( ii - 1 ), ')' ] );
    pause( 0.4 );
end
plot( coord( 1, path( [ 1, len ] ) ), coord( 2, path( [ 1, len ] ) ), '-b' );
x = sum( coord( 1, path( [ 1, len ] ) ) ) / 2 ;
y = sum( coord( 2, path( [ 1, len ] ) ) ) / 2 ;
text( x, y, [ '(' , num2str( len ), ')' ] );
pause( 0.4 );
hold off;
```

## 5、mySAA.m

```
% 模拟退火算法 ( Simulated Annealing Algorithm ) MATLAB 程序
```

```

clear;
% 程序参数设定
Coord = ... % 城市的坐标 Coordinates
    [0.6683 0.6195 0.4    0.2439 0.1707 0.2293 0.5171 0.8732 0.6878 0.8488; ...
     0.2536 0.2634 0.4439 0.1463 0.2293 0.761  0.9414 0.6536 0.5219 0.3609];
t0 = 1; % 初温 t0
iLk = 20; % 内循环最大迭代次数 iLk
oLk = 50; % 外循环最大迭代次数 oLk
lam = 0.95; %  $\lambda$  lambda
istd = 0.001; % 若内循环函数值方差小于 istd 则停止
ostd = 0.001; % 若外循环函数值方差小于 ostd 则停止
ilen = 5; % 内循环保存的目标函数值个数
olen = 5; % 外循环保存的目标函数值个数
% 程序主体
m = length(Coord); % 城市的个数 m
fare = distance(Coord); % 路径费用 fare
path = 1:m; % 初始路径 path
pathfar = pathfare(fare, path); % 路径费用 path fare
ores = zeros(1, olen); % 外循环保存的目标函数值
e0 = pathfar; % 能量初值 e0
t = t0; % 温度 t
for out = 1:oLk % 外循环模拟退火过程
    ires = zeros(1, ilen); % 内循环保存的目标函数值
    for in = 1:iLk % 内循环模拟热平衡过程
        [newpath, ~] = swap(path, 1); % 产生新状态
        e1 = pathfare(fare, newpath); % 新状态能量
        % Metropolis 抽样稳定准则
        r = min(1, exp(-(e1 - e0)/t));
        if rand < r
            path = newpath; % 更新最佳状态
            e0 = e1;
        end
        ires = [ires(2:end) e0]; % 保存新状态能量
    % 内循环终止准则: 连续 ilen 个状态能量波动小于 istd
    if std(ires, 1) < istd
        break;
    end
end
ores = [ores(2:end) e0]; % 保存新状态能量
% 外循环终止准则: 连续 olen 个状态能量波动小于 ostd
if std(ores, 1) < ostd
    break;
end
t = lam * t;

```

```

end
pathfar = e0;
% 输入结果
fprintf('近似最优路径为: \n')
%disp(char([path,path(1)]+64));
disp(path)
fprintf('近似最优路径费用\tpathfare=');
disp(pathfar);
myplot(path,Coord,pathfar);

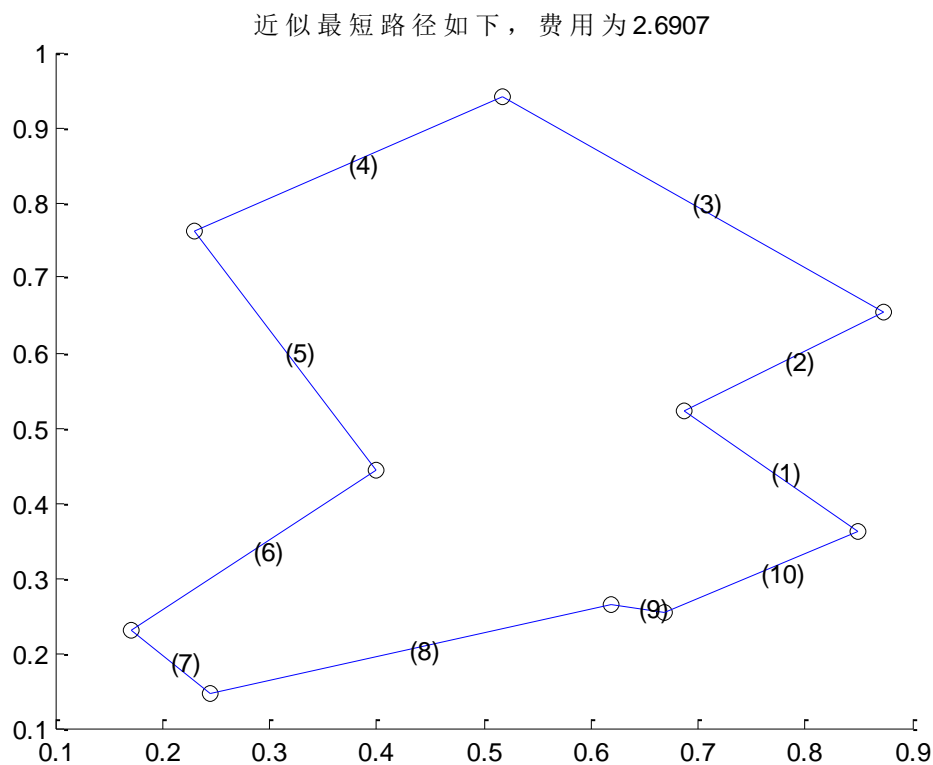
```

一次运行结果如下：

```

>> mySAA
近似最优路径为:
    10     9     8     7     6     3     5     4     2     1
近似最优路径费用 pathfare=    2.6907
>>

```



我试着运行了几次（只是改变了一下初温，也可以更改一下其他参数），发现初始温度  $t_0=1$  时程序的最后结果与最优解差距小的概率比较大。

希望对大家有用!!!