

蒙特卡洛模拟

(以下所有代码以及函数均来自于Matlab)

一.名字的由来:

蒙特卡洛是摩纳哥公国的一座城市，位于欧洲地中海之滨、法国的东南方，属于一个版图很小的国家摩纳哥公国，世人称之为“赌博之国”。

这种模拟的方法则是通过随机模拟的方法不停地“赌博”进而寻找逼近于最优解的解，从而被命名为“蒙特卡洛模拟”。

二.原理以及逻辑:

由大数定理可知，当样本容量足够大时，事件发生的频率可近似作为时间发生的概率。

在变量所在的定义域里面，通过randi函数进行随机取值，带入目标函数求得解作为最优解，然后不停地循环，若找到更优解则替换掉上一个最优解，循环次数越多解就越优。

三.算法的优点、缺点:

1.优点:

- 简单粗暴，不需要有过多的代码逻辑就可以完成；
- 在模型正确且重复数量足够的前提下，求得的解近似为最优解；
- 也可作为一种验证模型科学性的一种方法。

2.缺点:

- 由于方法过于简单粗暴，没有数学建模里面必要的数学逻辑以及模型，所以无法作为直接解决题目问题的方法，仅可作为求解模型的方法；
- 在对于某些约束条件极多的规划、TSP问题等方面，蒙特卡洛模拟的时间复杂度尤其之高，甚至超百年，无法采用此方法（现在计算机每秒可运行3亿次运算，可用prod函数判断能否使用蒙特卡洛模拟）。

（例如2022年Mathorcup的D题，完全可以直接使用蒙特卡洛模拟进行求解，但是需要极高的运算时间以及花费，也没有模型，不符合数学建模大赛的目的以及意义）

3.改进方法:

- 可以通过智能算法进行优化，比如粒子群算法、蚁群算法、模拟退火算法等等，这些算法运算起来时间复杂度低，比如20个城市的TSP问题，在没有任何约束的情况下蒙特卡洛模拟需要计算633年，而蚁群算法在0.05秒内就能找到最优解，但是需要的思维量较大，比较难以掌握。
- 在实在不会智能算法的情况下，可以稍微减少蒙特卡洛模拟的循环次数，然后多次进行模拟根据解的分布情况逐步缩小上下界的范围，进而使解更优。

四、蒙特卡洛模拟的典型例子

引例：布丰投针实验

取一张白纸，在上面画许多条间距为 a 的平行线，取一根长为 l 的针 ($l \leq a$)，随机向画有平行线的纸投掷 n 次，计算针与直线相交的概率。

在现实里，这个问题可以用几何模型来计算，最终答案为 $\frac{2l}{\pi n}$ ，进而可以把 π 给计算出近似解，但是在计算机里可用蒙特卡洛模拟解决这个问题。

代码：

```
clear
clc
l = 1; %针的长度（任意）
a = 2; %平行线的间距（大于l即可）
n = 100000; %循环次数，n越大求出来的值越接近π
m = 0;
x = rand(1,n)*a/2; %在[0,a/2]内随机取n个数，x表示针中点和最近一条平行线之间的距离
phi = rand(1,n)*pi; %在[0,pi]内随机取n个数，phi表示针与最近平行线的夹角
for i=1:n
    if x(i) < 1/2*sin(phi(i)) %判断是否相交
        m = m+1
    end
end
p = m/n %计算概率
realpi = (2*l)/(a*p) %验证
```

1. 蒙特卡洛求解有约束的非线性规划问题

目标函数	约束条件	变量
$\max f(x) = x_1 x_2 x_3$	$-x_1 + 2x_2 + 2x_3 \geq 0$ $x_1 + 2x_2 + 2x_3 \leq 72$ $10 \leq x_2 \leq 20$ $x_1 - x_2 = 10$	x_1, x_2, x_3

思路：

从约束条件中求出每个变量受限的大致范围，从中用随机数生成若干组成实验点，并验证它们是否满足所有的约束条件，若满足，则将其划分到可行组，再从可行组中找到函数的最大值和最小值。

技巧：

- 由 $x_1 - x_2 = 10$ ，可知 $x_2 = x_1 - 10$ ，则 $f(x) = x_1(x_1 - 10)x_3$;
- $10 \leq x_2 \leq 20$ ，可知 $20 \leq x_1 \leq 30$;
- 由 $-x_1 + 2x_2 + 2x_3 \geq 0$ ，可知： $x_3 \geq \frac{1}{2}(x_1 - 2x_2) \geq \frac{1}{2}(20 - 2 * 20) = -10$;
- 由 $x_1 + 2x_2 + 2x_3 \leq 72$ ，可知： $x_3 \leq \frac{1}{2}(72 - x_1 - 2x_2) = \frac{1}{2}(72 - 20 - 2 * 10) = 16$ ，即最终 $x_3 \in [-10, 16]$ 。

代码：

```
n = 1000000; %生成随机数组数
x1 = unifrnd(20,30,n,1); %生成在[20,30]分布，n行1列的向量构成x1
x2 = x1 - 10;
x3 = unifrnd(-10,16,n,1); %同x1
```

```

fmax = -inf; % 初始化最大数为负无穷 (找到比这个数大的数就迭代)
for i = 1:n;
    x = [x1(i),x2(i),x3(i)]; % 构建x向量
    if (-x(1) + 2*x(2) + 2*x(3) >= 0) & (x(1) + 2*x(2) + 2*x(3) <= 72)
        result = x(1)*x(2)*x(3);
        if result > fmax
            fmax = result;
        end
    end
end
disp(strcat('蒙特卡洛模拟得到的最大值为: ', num2str(fmax)))
disp('最大处x1,x2,x3的取值为: ')
disp(x)

```

注：此代码求出来的为初始值，还需缩小范围再次模拟

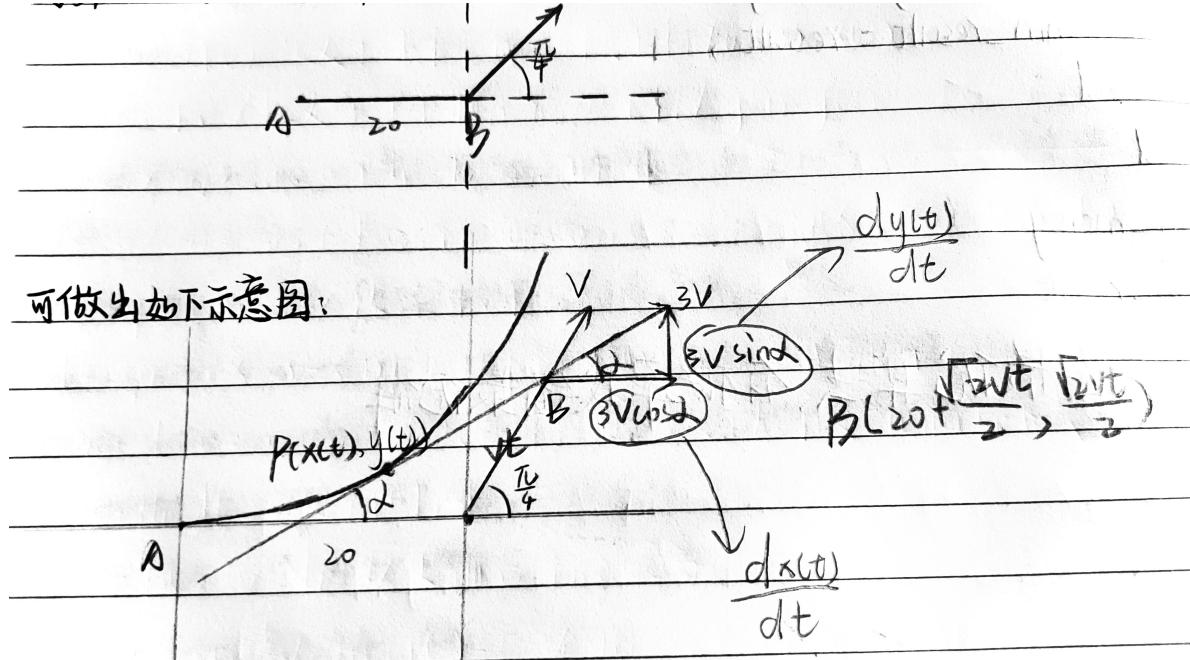
2. 蒙特卡洛求解导弹追击问题

问题：

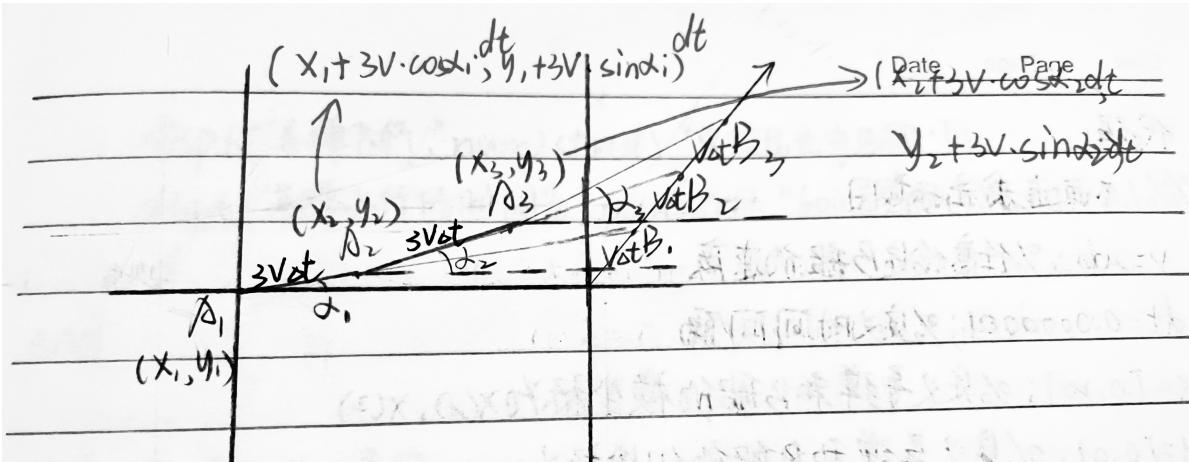
位于坐标原点的 A 船向位于其正东方 20 个单位的 B 船发射导弹， B 船以时速 v 单位（常数）沿东北方向逃逸。若导弹的速度为 $3v$ ，导弹的射程是 50 个单位，画出导弹运行的路线，并且判断导弹能否在射程内击中 B 船。

(本题用微分方程可以直接求出精确解)

首先我们要对图像进行分析，如下图：



设导弹飞行的总时间为 T ，我们可以将 T 分为 n 个时间间隔 Δt ， Δt 越小越好，不妨取 $\Delta t = 0.0000001$ ，则原图可近似分解为：



如上图，假设 B 船先走 Δt 个时间到达 B_1 ，导弹才开始从 A_1 发射（如果同时进行代码难以实现），那么在下一个 Δt 内，导弹的方向一定是沿着 A_1B_1 ，斜率为 $\tan \alpha_1$, α_1 为夹角，以此类推，且 $|A_i, A_{i+1}| = 3v\Delta t$

记 dd 表示导弹和 B 的距离，当 $|dd| = 0$ 时，表示相撞，但是用此方法使用浮点数计算几乎不可能达到 0，所以我们规定当 dd 小于一定值时表示相撞。

代码：

```

clear;clc
v=200; % 任意给定B船的速度（后期我们可以再改的）
dt=0.0000001; % 定义时间间隔
x=[0,20]; % 定义导弹和B船的横坐标分别为x(1)和x(2)
y=[0,0]; % 定义导弹和B船的纵坐标分别为y(1)和y(2)
t=0; % 初始化导弹击落B船的时间
d=0; % 初始化导弹飞行的距离
m=sqrt(2)/2; % 将sqrt(2)/2定义为一个常量，使后面看起来很简洁
dd=sqrt((x(2)-x(1))^2+(y(2)-y(1))^2); % 导弹与B船的距离
for i=1:2
    plot(x(i),y(i),'.k','MarkerSize',1); % 画出导弹和B船所在的坐标，点的大小为1，颜色为黑色(k)，用小点表示
    grid on; % 打开网格线
    hold on; % 不关闭图形，继续画图
end
axis([0 30 0 10]) % 固定x轴的范围为0-30 固定y轴的范围为0-10
k = 0; % 引入一个变量 为了控制画图的速度（因为Matlab中画图的速度超级慢）
while(dd>=0.001) % 只要两者的距离足够大，就一直循环下去。（两者距离足够小时表示导弹击中，这里的临界值要结合dt来取，否则可能导致错过交界处的情况）
    t=t+dt; % 更新导弹击落B船的时间
    d=d+3*v*dt; % 更新导弹飞行的距离
    x(2)=20+t*v*m; y(2)=t*v*m; % 计算新的B船的位置（注：m=sqrt(2)/2）
    dd=sqrt((x(2)-x(1))^2+(y(2)-y(1))^2); % 更新导弹与B船的距离
    tan_alpha=(y(2)-y(1))/(x(2)-x(1)); % 计算斜率，即tan(alpha)
    cos_alpha=sqrt(1/(1+tan_alpha^2)); % 利用公式：sec(alpha)^2 = (1+tan(alpha)^2) 计算出cos(alpha)
    sin_alpha=sqrt(1-cos_alpha^2); % 利用公式：sin(alpha)^2 + cos(alpha)^2 = 1 计算出sin(alpha)
    x(1)=x(1)+3*v*dt*cos_alpha; y(1)=y(1)+3*v*dt*sin_alpha; % 计算新的导弹的位置
    k = k +1 ;
    if mod(k,500) == 0 % 每刷新500次时间就画出下一个导弹和B船所在的坐标 mod(m,n)表示求m/n的余数
        for i=1:2
            plot(x(i),y(i),'.k','MarkerSize',1);
            hold on; % 不关闭图形，继续画图
        end
    end

```

```

    pause(0.001); % 暂停0.001s后再继续下面的操作
end
if d>50 % 导弹的有效射程为50个单位
    disp('导弹没有击中B船');
    break; % 退出循环
end
if d<=50 & dd<0.001 % 导弹飞行的距离小于50个单位且导弹和B船的距离小于0.001 (表示击中)
    disp(['导弹飞行',num2str(d),'个单位后击中B船'])
    disp(['导弹飞行的时间为',num2str(t*60),'分钟'])
end
end

```

3.蒙特卡洛求解简单TSP问题

问题：

一个售货员必须访问10个城市，这十个城市是一个完全图，售货员需要恰好访问所有城市一次，并且回到最初的城市，求路径最短的路线

数据如下：

城市	X	Y
1	0.6683	0.2536
2	0.6683	0.2634
3	0.4000	0.4439
4	0.2429	0.1463
5	0.1707	0.2293
6	0.2293	0.7610
7	0.5171	0.9414
8	0.8732	0.6536
9	0.6878	0.5219
10	0.8488	0.3609
代码：		

```

clear;clc
% 只有10个城市的简单情况
coord =[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]'; % 城市坐标矩阵, n行2列

n = size(coord,1); % 城市的数目

figure(1) % 新建一个编号为1的图形窗口
plot(coord(:,1),coord(:,2),'o'); % 画出城市的分布散点图
for i = 1:n

```

```

text(coord(i,1)+0.01,coord(i,2)+0.01,num2str(i)) % 在图上标上城市的编号（加上
0.01表示把文字的标记往右上方偏移一点)
end
hold on % 等一下要接着在这个图形上画图的

d = zeros(n); % 初始化两个城市之间的距离矩阵全为0
for i = 2:n
    for j = 1:i
        coord_i = coord(i,:); x_i = coord_i(1); y_i = coord_i(2); % 城市i
        的横坐标为x_i, 纵坐标为y_i
        coord_j = coord(j,:); x_j = coord_j(1); y_j = coord_j(2); % 城市j
        的横坐标为x_j, 纵坐标为y_j
        d(i,j) = sqrt((x_i-x_j)^2 + (y_i-y_j)^2); % 计算城市i和j的距离
    end
end
d = d+d'; % 生成距离矩阵的对称的一面

min_result = +inf; % 假设最短的距离为min_result, 初始化为无穷大, 后面只要找到比它小的就对
其更新
min_path = [1:n]; % 初始化最短的路径就是1-2-3-...-n
N = 10000000; % 蒙特卡罗模拟的次数
for k = 1:N % 开始循环
    result = 0; % 初始化走过的路程为0
    path = randperm(n); % 生成一个1-n的随机打乱的序列
    for i = 1:n-1
        result = d(path(i),path(i+1)) + result; % 按照这个序列不断的更新走过的路程这个
        值
    end
    result = d(path(1),path(n)) + result; % 别忘了加上从最后一个城市返回到最开始那个城
    市的距离
    if result < min_result % 判断这次模拟走过的距离是否小于最短的距离, 如果小于就更新最短
    距离和最短的路径
        min_path = path;
        min_result = result
    end
end
min_path
min_path = [min_path,min_path(1)]; % 在最短路径的最后面加上一个元素, 即第一个点(我们要
生成一个封闭的图形)
n = n+1; % 城市的个数加一个(紧随着上一步)
for i = 1:n-1
    j = i+1;
    coord_i = coord(min_path(i,:)); x_i = coord_i(1); y_i = coord_i(2);
    coord_j = coord(min_path(j,:)); x_j = coord_j(1); y_j = coord_j(2);
    plot([x_i,x_j],[y_i,y_j],'-') % 每两个点就作出一条线段, 直到所有的城市都走完
    pause(0.5) % 暂停0.5s再画下一条线段
    hold on
end

```

注：以上均为简单的TSP问题，若城市过多则必须使用智能算法，不建议用蒙特卡洛求解TSP，实际情况远比理想模型复杂得多（可以参考2022年电工杯B题）

4.蒙特卡洛求解0-1规划问题

问题：

某同学要从六家线上商城选购B1、B2、B3、B4、B5五本书，请为他选择花费最少的方案

数据如下：

	B1	B2	B3	B4	B5	运费
A	18	39	29	48	59	10
B	24	45	23	54	44	15
C	22	45	23	53	53	15
D	28	47	17	57	47	10
E	24	42	24	47	59	10
F	27	48	20	55	53	15

思路：

目标函数即为总花费，包括五本书的总价格以及总运费

可以引入0-1变量，1表示该同学在第i家店买第j本书，0表示没有买到

代码：

```
min_money = +inf;
min_result = randi([1,6],1,5); %取出5个整数，意为五本书在取出的5个店铺里购买
n = 1000000;
M = [18 39 29 48 59
      24 45 23 54 44
      22 45 23 53 53
      28 47 17 57 47
      24 42 24 47 59
      27 48 20 55 53];
freight = [10 15 15 10 10 15];
for k = 1:n
    result = randi([1,6],1,5);
    index = unique(result); %unique函数可以剔除矩阵里面的重复值
                           %在这里可以表示在哪些商店购买了商品
    money = sum(freight(index)); %计算花费
    for i = 1:5
        money = money+M(result(i),i);
    end
    if money < min_money;
        min_money = money;
        min_result = result;
    end
end
min_money
min_result
```

五、总结

蒙特卡洛模拟是一种非常简单粗暴的算法，在许多规划问题里面广泛使用，以及多元回归分析里面探究内生性对结果的影响以及基于熵权法对Topsis模型的修正里面也内涵蒙特卡洛的思想，在TSP问题以及导弹追击模型里面也可使用，但是不建议。

蒙特卡洛模拟没有具体的模型，主要就是靠：随机取值-循环-最优值替代这个逻辑运行，个人认为是必须要掌握的算法，但是相比于其他算法，如果想让评委老师眼睛一亮，还是建议倾向于选择效率更高、准确度更高、逻辑性更强的智能算法，不过前提是得会使用。