一个喝醉的小鸟有多大的概率永远也回不了家了？

拉普拉斯算符
数学话题下的优秀答主

大概一百年前，匈牙利的数学家Polya证明了一个定理，该定理说，在 d 维格点上随机游走粒子，经过足够长的时间能返回出发点的概率依赖于维度 d . 如果 $d=1,2$, 那么该粒子可以返回出发点的概率为1，否则该返回概率小于一。因此就有了这句话，一个喝醉的酒鬼(二维随机游走)一定可以走回家，但是一个喝醉的小鸟（三维随机游走）却有可能再也不能回家了。注意，不是小鸟一定不能回家，而是有一定的概率不能回家。

关于这个定理的证明，这里有一个非常好的总结:arxiv.org/pdf/1301.3916...

根据这篇文章，我们知道在 d 维格点上随机游走的粒子，它最终返回出发点的概率为

$$p=1-\frac{1}{Q}$$

其中

$$Q=\int_0^\infty e^{-t}\left(I_0(tz/d)\right)^d dt$$

这里，我们使用了零阶虚贝塞尔函数，定义为

$$I_0(x)=\frac{1}{\pi}\int_0^\pi e^{x\cos\theta}d\theta$$

为了计算一只喝醉的小鸟不能回家的概率，我们需要计算当 $d=3$ 的时候的回归概率 p . 为此，我们需要做一些数值积分。为了方便计算，将贝塞尔函数的积分表示代入到积分 Q 中得到

$$Q=\frac{1}{\pi^d}\int\int\dots\int\prod_{i=1}^d d\theta_i\frac{1}{1-\frac{1}{d}\sum_{i=1}^d\cos\theta_i}$$

对于三维的情况， 我们得到

$$Q=\frac{1}{\pi^3}\int_0^\pi d\theta\int_0^\pi d\phi\int_0^\pi d\psi\frac{1}{1-\frac{1}{3}\left(\cos\theta+\cos\phi+\cos\psi\right)}$$

显然，该积分在 $\theta=\phi=\psi=0$ 的时候有可能会出现奇异性。将积分在零点附近展开得到

$$Q\approx\frac{1}{\pi^3}\int\int\int\frac{d\theta d\phi d\psi}{\frac{1}{6}(\theta^2+\phi^2+\psi^2)}=\frac{24}{\pi^2}$$

所以这个积分实际上是有限的。因此，小鸟的回归概率小于一。为了得到这个积分的确切值，我们需要做数值积分。为了可以让数值计算更加高效，我们先将能手动积分的那部分算出来。对积分变量 ψ 求积分，得到

$$Q=\frac{1}{\pi^2}\int_0^\pi d\theta\int_0^\pi d\phi\frac{1}{\sqrt{\left(1-\frac{1}{3}\cos\theta-\frac{1}{3}\cos\phi\right)^2-\frac{1}{3^2}}}$$

因为零点仍然是这个积分函数的奇异点，所以我们应该将这个点做特殊处理。在零点附近，我们有

$$\begin{aligned}Q_\epsilon&=\frac{1}{\pi^2}\int_0^\epsilon d\theta\int_0^\epsilon d\phi\frac{1}{\sqrt{\left(1-\frac{1}{3}\cos\theta-\frac{1}{3}\cos\phi\right)^2-\frac{1}{3^2}}}\\&\approx\frac{1}{\pi^2}\int_0^\epsilon d\theta\int_0^\epsilon d\phi\frac{1}{\sqrt{\frac{2}{3}\frac{1}{3}\frac{1}{2}(\theta^2+\phi^2)}}\\&=\frac{3\epsilon}{\pi^2}\left(\mathrm{ArcSinh}1+\log(1+\sqrt{2})\right)\end{aligned}$$

对于其他的积分区域，我们直接做数值积分就可以了，也就是

$$Q=\frac{1}{\pi^2}\int_\epsilon^\pi d\theta\int_\epsilon^\pi d\phi\frac{1}{\sqrt{\left(1-\frac{1}{3}\cos\theta-\frac{1}{3}\cos\phi\right)^2-\frac{1}{3^2}}}+Q_\epsilon$$

求得该数值积分的解，代入前面的方程可以得到小鸟的回归概率。

为了验算我们得到的数值解，我还做了蒙特卡洛模拟。模拟结果表明，数值计算的结果与蒙特卡洛模拟得到的结果非常相似，因此，我们的数值积分的程序是很精确的。

数值积分的程序和蒙特卡洛模拟的程序如下:

```
import numpy as np

def integrate_3d():
    eps = 0.01
    lower = eps
    upper = np.pi
    N = 1000
    delta = (upper - lower)/float(N)
    s = 0.0
    for i in range(N+1):
        for j in range(N+1):
            theta = lower + i * delta
            phi = lower + j * delta
            value = 1.0/np.sqrt((1.0 - 1.0/3.0*np.cos(theta) - 1.0/3.0*np.cos(phi
            s += delta**2 * value
            if False and value > 50:
                print theta, phi, value
    return s/np.pi**2 +  eps * (np.arcsinh(1) + np.log(1 + np.sqrt(2))) * 3.0/np.

class State:
    def __init__(self, x, y, z):
        self.x = x
        self.y = y
        self.z = z

    def equals(self, that):
        return self.x == that.x and self.y == that.y and self.z == that.z

    def add(self, that):
        return State(self.x + that.x, self.y + that.y, self.z + that.z)

    def assign(self, that):
        self.x = that.x
        self.y = that.y
        self.z = that.z

    def __str__(self):
        return "(" + str(self.x) + ", " + str(self.y) + ", " + str(self.z) + ")"

def update(initial, r):
    if (r < 1.0/6):
        return initial.add(State(1, 0, 0))
    elif (r >= 1.0/6 and r < 1.0/3):
        return initial.add(State(-1, 0, 0))
    elif (r >= 1.0/3 and r < 1.0/2):
        return initial.add(State(0, 1, 0))
    elif (r >= 1.0/2 and r < 2.0/3):
        return initial.add(State(0, -1, 0))
    elif (r >= 2.0/3 and r < 5.0/6):
        return initial.add(State(0, 0, 1))
    else:
        return initial.add(State(0, 0, -1))

def random_walk_3d():
    import random
    upper = 1000
    origin = State(0, 0, 0)
    initial = State(0, 0, 0)
    counter = 0
    while counter < upper:
        counter += 1
        r = random.uniform(0, 1)
        updated = update(initial, r)
        if (updated.equals(origin)):
            return True
        initial.assign(updated)
    return False

def monte_carlo_simulation(times):
    return_times = 0
    for i in range(times):
        returned = random_walk_3d()
        if returned:
            return_times += 1
    return float(return_times)/float(times)

def get_probability(times):
    return_frequencies = []
    for i in range(times):
        f = monte_carlo_simulation(20)
        return_frequencies.append(f)
        print "i = " + str(i+1) + ", total = " + str(times) + ", frequency = " +
    mean = sum(return_frequencies)/float(len(return_frequencies))
    variance = sum(map(lambda ele: (ele - mean)**2, return_frequencies))/float(le
    sigma = np.sqrt(variance)
    print "mean = " + str(mean) + ", sigma = " + str(sigma)
    return mean, sigma
```

数值积分得到小鸟的返回概率为0.328, 而蒙特卡洛模拟得到的返回概率为0.3316. 两者相差不大。因此，喝醉的小鸟有大概2/3的概率再也回不来了。

最后讲一些听说的关于这个定理的八卦。当年Polya有一个学生，这个学生的女朋友来看这个学生。Polya和他的学生以及他学生的女朋友一起到附近的公园里散步。到了公园之后，Polya自己一个人走，他的学生和学生的女朋友一起走。他们在公园里随机地散步，但是让他们尴尬的是，他们总是在公园的某个地方相遇。Polya有点担心他的学生会误以为他在跟踪他俩，因此他回去就研究了两个随机游走的粒子在任意维格点上随机游走的问题，并且证明了，如果格点的维数小于等于二，那么这两个粒子必然会反复相遇。这样，Polya就洗清了自己的“罪名”。这个问题的另一个等价表述是，一个粒子从二维格点上的某点出发做随机游走，那么该粒子重回出发点的概率是一，也就是这个粒子回到初始点的次数趋近于无穷大。因此，如果你和你的朋友在一个巨大的广场上走失了，不必惊慌，因为只要你们两个中有一个人开始找对方，那么经过足够长的时间后，你们两个必然会重逢。定义可以无限次返回的随机游走为recurrent，不能无限次返回的随机游走为transient，那么Polya的定理就是说，维数小于等于二的随机游走是recurrent，而维数大于二的随机游走是transient.

备注

根据这篇文章pnas.org/content/pnas/7...， 这个积分

$$Q=\frac{1}{\pi^3}\int_0^\pi d\theta\int_0^\pi d\phi\int_0^\pi d\psi\frac{1}{1-\frac{1}{3}\left(\cos\theta+\cos\phi+\cos\psi\right)}$$

有精确解，结果为

$$Q=\frac{\sqrt{6}}{32\pi^3}\Gamma\left(\frac{1}{24}\right)\Gamma\left(\frac{5}{24}\right)\Gamma\left(\frac{7}{24}\right)\Gamma\left(\frac{11}{24}\right)=1.5163860591519784$$

代入 $p=1-Q^{-1}$ 得到概率的精确解为 0.34053732955099936.

我只能说，数学家们真厉害。

编辑于 2020-05-28

47 条评论

Jane Shu 2019-08-30
这个“为证自己的清白”简直牛逼大发了
44

勃尔豪斯 回复 Jane Shu 2019-08-30
作为一个jeek，我能理解他的心情
2

dreday 回复 Jane Shu 2019-09-04
有点nerd
赞

展开其他 1 条回复

九三工人 2019-08-30
所以喝醉了住在一楼的话一定可以回家，住在二楼往上几乎永远回不了家?
12

Jason 回复 九三工人 2019-08-30
恩，那么如果住在重庆就别喝醉了
37

Alan Liang 回复 九三工人 2019-08-30
说的好像您能从1号楼的3楼直接走到2号楼的3楼似的
21

查看全部 12 条回复

临渊听泉 2019-08-30
这积分真是该死的甜美
7

jerna 2019-08-30
酒醒了不就能回去了吗
5

H4XeO6 2019-09-03
这积分竟然能写出精确值，你们搞数学的都这么可怕的吗[飙泪笑]
4

乘客嘎嘎 2019-08-29
那四维呢?
4

tommaxmim 回复 乘客嘎嘎 2019-08-30
0.1左右
14

迷失枫叶 2019-09-05
喝醉的麻雀都被炸着吃了
1

胡仲易 2019-08-30
recurrent还分为expectation of return time是无穷的还是有限的。
1

澄格 2019-08-30
你们更厉害
1

猫尸 2020-10-23
答主能不能解答一下在广场上找人的时候是两个人同时找对方更容易找到还是一人不动另一人找更容易找到?
赞

Saaadi 2019-10-06
为什么小鸟会喝醉?
赞

爱喝汽水的文文 2019-10-03
硬核。
赞

纳尼 2019-09-29
我是谁，我在哪，我怎么了
赞

Ken G 2019-09-23
非常浪漫
赞

松明 2019-09-23
精确解这个厉害了啊
赞

MasterISBE 2019-09-11
其实我一直对概率有这个困惑。。那怎么证明我某一只喝醉的小鸟，它是属于那个1/3, 还是2/3?
赞

张近微 2019-09-10
这题目好像一句诗啊
赞

No Name 2019-09-10
等一下？这能说明只要跑的够多够久和同一个人的艳遇必定会重复吗（不考虑飞机和地铁这种“三维”）
赞

白毛小鼠偷烛花 2019-09-09
我被有趣的题目骗了进来[捂脸]
赞