14 随机行走返回原点的概率

许传奇 PB16021546

1 题目

数值研究d（$d=1,2,3$）维空间中随机行走返回原点的概率P_d，讨论它随步数N的变化关系$P_d(N)$，能否定义相关的指数值？

2 原理与算法

2.1 原理

2.1.1 返回原点的概率

不同维数下返回原点的概率可以求出理论值。

设总步数为N。不失一般性，可以设$d=1,2,3$时，粒子分别在x轴、xy平面、xyz三维空间进行随机行走。每次随机行走的步数都为1个单位长度，且只会沿x、y、z轴正方向或负方向进行随机行走，且概率相等。沿x、y、z轴负方向、正方向行走的步数分别为x_-,x_+,y_-,y_+,z_-,z_+，返回原点时，满足的条件应该是：

$$x_-=x_+,\ y_-=y_+,\ z_-=z_+ \tag{1}$$

1. 当$d=1$时，约束条件和返回原点的条件为：

$$\begin{cases} x_-+x_+=N \\ x_-=x_+ \end{cases} \quad \Rightarrow \quad x_-=x_+=\frac{N}{2} \tag{2}$$

因此，返回原点的概率为：

$$P_d=\frac{N!}{\left[(\frac{N}{2})!\right]^2}(\frac{1}{2})^N \tag{3}$$

2. 当$d=2$时，约束条件和返回原点的条件为：

$$\begin{cases} x_-+x_++y_-+y_+=N \\ x_-=x_+ \\ y_-=y_+ \end{cases} \quad \Rightarrow \quad x_++y_+=\frac{N}{2} \tag{4}$$

因此，返回原点的概率为：

$$P_d=\sum_{x_+=0}^{\frac{N}{2}}\frac{N!}{\left[x_+!(\frac{N}{2}-x_+)!\right]^2}(\frac{1}{4})^N \tag{5}$$

3. 当$d=3$时，约束条件和返回原点的条件为：

$$\begin{cases} x_- + x_+ + y_- + y_+ + z_- + z_+ = N \\ x_- = x_+ \\ y_- = y_+ \\ z_- = z_+ \end{cases} \Rightarrow \quad x_+ + y_+ + z_+ = \frac{N}{2} \tag{6}$$

因此，返回原点的概率为：

$$P_d = \sum_{x_+=0}^{\frac{N}{2}} \sum_{y_+=0}^{\frac{N}{2}-x_+} \frac{N!}{\left[x_+! \cdot y_+! \cdot (\frac{N}{2} - x_+ - y_+)!\right]^2} (\frac{1}{6})^N \tag{7}$$

2.1.2 指数值的求解

由于返回原点的概率过于复杂，难以求解理论值。但在步数N非常大的时候，可以求得近似值。

对于阶乘，存在Sterling公式，即：

$$\lim_{N\to\infty} \frac{N!}{\sqrt{2\pi N}(\frac{N}{e})^N} = 1 \tag{8}$$

对于$d=1$时，当步数N很大时，有：

$$\begin{aligned} P_d &= \frac{N!}{\left[(\frac{N}{2})!\right]^2} (\frac{1}{2})^N \\ &\approx \frac{\sqrt{2\pi N}(\frac{N}{e})^N}{\left[\sqrt{2\pi \frac{N}{2}}(\frac{N}{2e})^{\frac{N}{2}}\right]^2} (\frac{1}{2})^N \\ &= \sqrt{\frac{4}{\pi}} N^{-\frac{1}{2}} \end{aligned} \tag{9}$$

因此，$d=1$时，指数为$-\frac{1}{2}$。

对于$d=2, d=3$的情况，由于需要求和，无法再得到比较好的近似。

不过，当N非常大时，认为在其他方向上回到原点的概率近似也与$N^{-\frac{1}{2}}$成正比，同时认为概率满足近似上的独立，因此返回原点的概率近似满足关系：$P_d \propto P_x P_y P_z$，P_x、P_y、P_z分别为在x、y、z方向上回到原点的概率。x、y、z之间不独立的关系影响的主要是$P_d = kN^\nu$中的系数k，而对ν的大小在N很大的时候几乎没有影响。

因此对应的指数为$-\frac{1}{2}d$，即当$d=2$时，指数为-1，当$d=3$时，指数为$-\frac{3}{2}$。

2.2 算法

2.2.1 返回原点的概率P_d随步数N的变换关系$P_d(N)$

对于随机行走问题，采用C语言自带的随机数生成函数rand()进行随机数生成。

$d=1,2,3$时，随机行走坐标变化的条件如下：

1. 当$d=1$时：

$$\begin{cases} x^{n+1} = x^n - 1 \quad rand()\%2 == 0 \\ x^{n+1} = x^n + 1 \quad rand()\%2 == 1 \end{cases} \tag{10}$$

2. 当$d=2$时:

$$\begin{cases} x^{n+1}=x^n-1 & rand()\%4==0 \\ x^{n+1}=x^n+1 & rand()\%4==1 \\ y^{n+1}=y^n-1 & rand()\%4==2 \\ y^{n+1}=y^n+1 & rand()\%4==3 \end{cases} \tag{11}$$

3. 当$d=3$时:

$$\begin{cases} x^{n+1}=x^n-1 & rand()\%6==0 \\ x^{n+1}=x^n+1 & rand()\%6==1 \\ y^{n+1}=y^n-1 & rand()\%6==2 \\ y^{n+1}=y^n+1 & rand()\%6==3 \\ z^{n+1}=z^n-1 & rand()\%6==4 \\ z^{n+1}=z^n+1 & rand()\%6==5 \end{cases} \tag{12}$$

设置三个不同的循环：

第一个循环，设置不同的随机行走步数；第二个循环，设置总个数为num_particles个不同的粒子；第三个循环，粒子进行随机行走。

用变量count计数返回到原点的粒子个数。在第二个循环中，起始时设置count为0。当第三个循环结束时，即粒子随机行走结束时，判断是否在原点，即$d=1:x=0$; $d=2:x=0,y=0$; $d=3:x=0,y=0,z=0$是否成立，如果成立，则：count=count+1；否则继续进行下一个粒子的随机行走。当所有粒子都随机行走完毕，输出：count/num_particles，即为返回原点的粒子的概率。

通过第一个循环，就能求出不同步数下返回原点的概率。

不过，因为返回原点时，步数必须为偶数。若步数为奇数，不可能满足返回原点的条件，即返回原点的概率为0。因此步数的起始值和步长都设置成偶数，可以减少计算时间。

将遍历到的步数N与对应的返回原点的概率P_d作出图像，就是返回原点的概率P_d随步数N的变换关系$P_d(N)$。

本次作业中，为了数据表达的方便，不考虑步数为奇数时的结果，同时将返回原点的概率P_d随步数N的关系$P_d(N)$设成连续的函数。但应该注意到步数只能为正整数，且只有步数为偶数时才有非平凡的返回原点的概率P_d

2.2.2 指数值

假设返回原点的概率近似满足$P_d=kN^\nu$，则对其求对数，有：

$$\ln P_d=\nu\ln N+\ln k \tag{13}$$

将得到的数据N和对应的P_d求对数后进行一次拟合，如果拟合的关系比较好的话，则可以认为返回原点的概率近似满足指数关系，指数就是一次曲线的斜率。

3 源文件使用说明

编译并运行“14_2RandomWalk.cpp”，将弹出命令行，按要求输入粒子总数num_particles、随机行走次数num_steps，程序将运行。

输入数据后，命令行上会显示相应数据。其中，第一列是步数N，第二列是$d = 1$时返回原点的概率，第三列是$d = 2$时返回原点的概率，第四列是$d = 3$时返回原点的概率。

程序运行并将数据输出到"data.txt"文件中。其中，第一列是步数N，第二列是$d = 1$时返回原点的概率，第三列是$d = 2$时返回原点的概率，第四列是$d = 3$时返回原点的概率。

编译并运行"plot.py"，就可以画出相应图像。

4 计算结果及具体分析

4.1 三个维度下返回原点的概率

如下图所示，是最大步数为1000时三个维度下返回原点的概率：

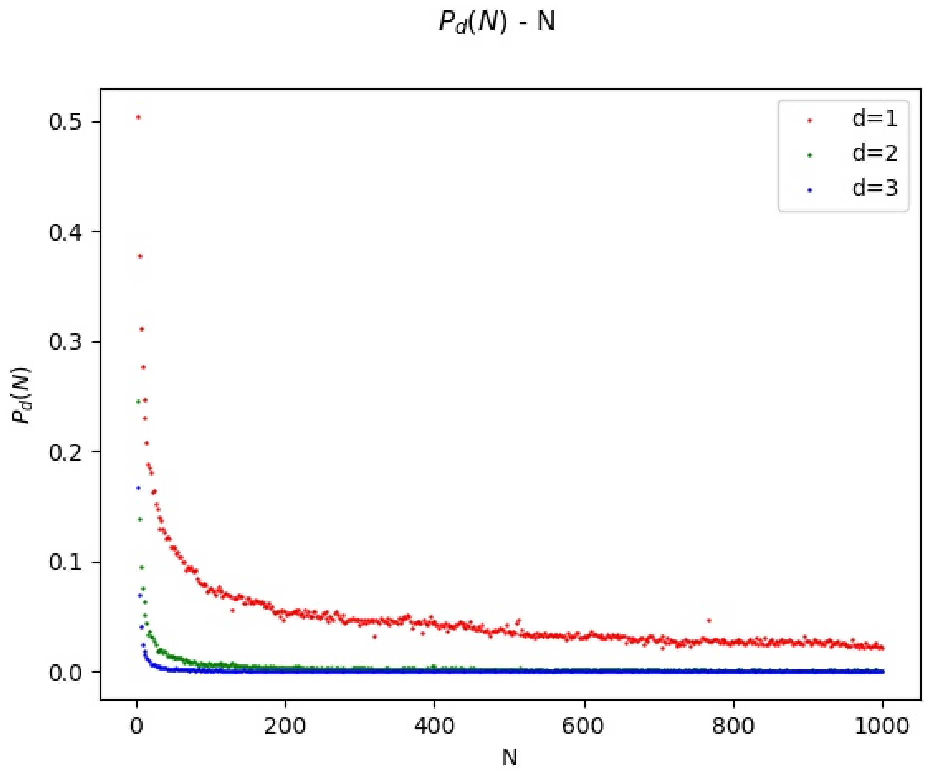

图 1: 三个维度下的返回原点的概率

可见当$d = 2, d = 3$时，概率很快衰减到几乎为0。因此只用关注比较小的最大步数时的情况。

重新设置最大步数为100，如下所示

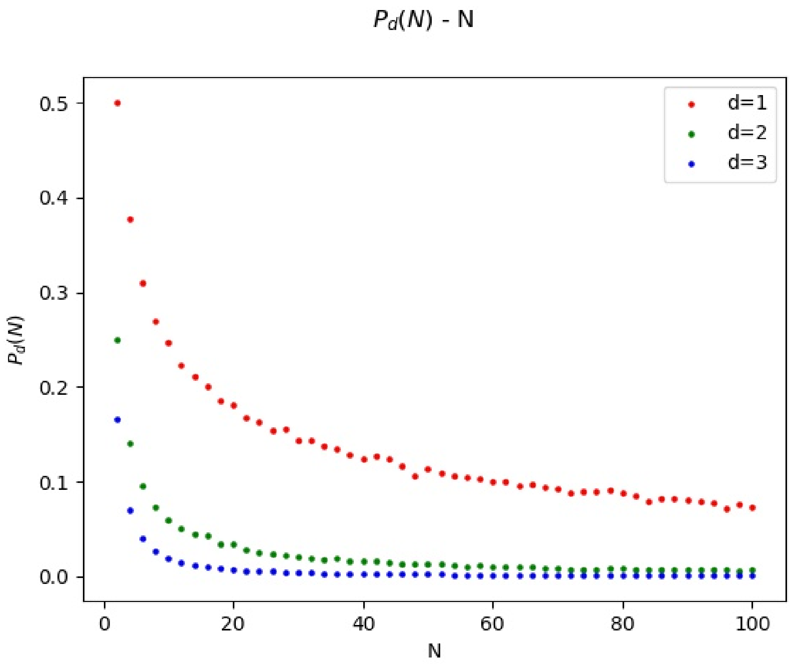

图 2: 三个维度下的返回原点的概率

4.2 不同维度下与理论曲线的对比图

考虑不同维度下与理论曲线的对比图，如下图所示：

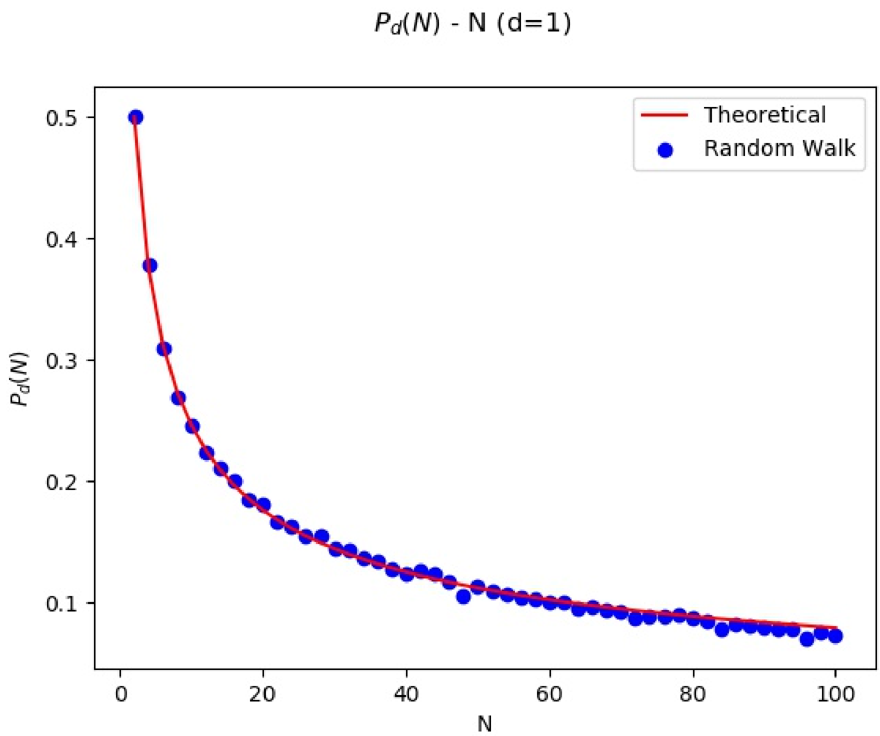

图 3: $d = 1$时与理论曲线的对比图

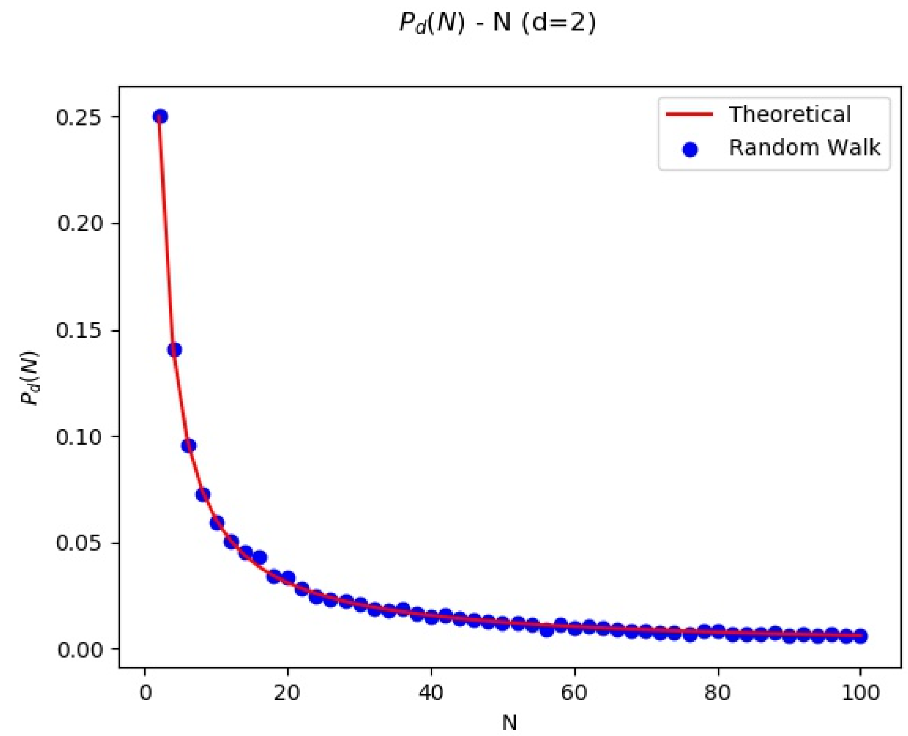

图 4: $d = 2$时与理论曲线的对比图

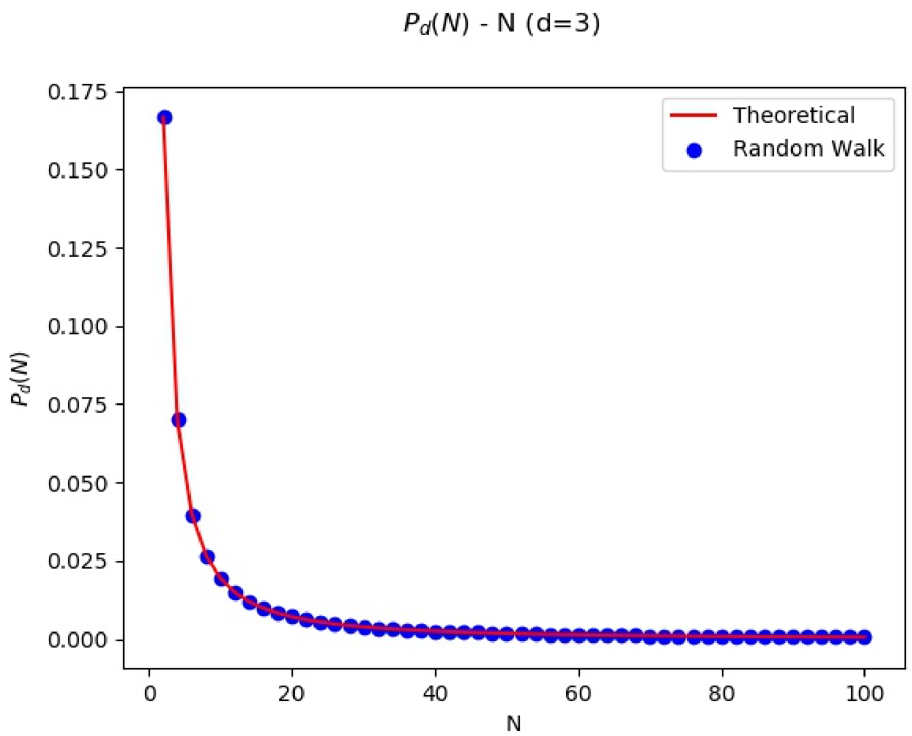

图 5: $d = 3$时与理论曲线的对比图

可见三种维度下与理论曲线都非常符合，证明程序编写无误。

4.3 对数图像线性拟合

考虑不同维度下对数图像线性拟合的图像与拟合直线的方程：

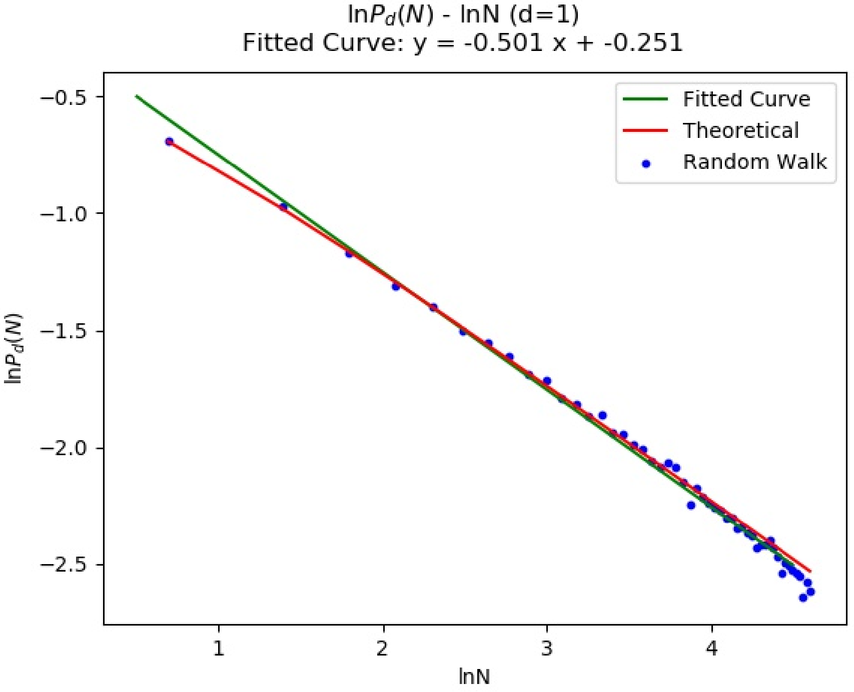

图 6: $d = 1$时对数图像线性拟合

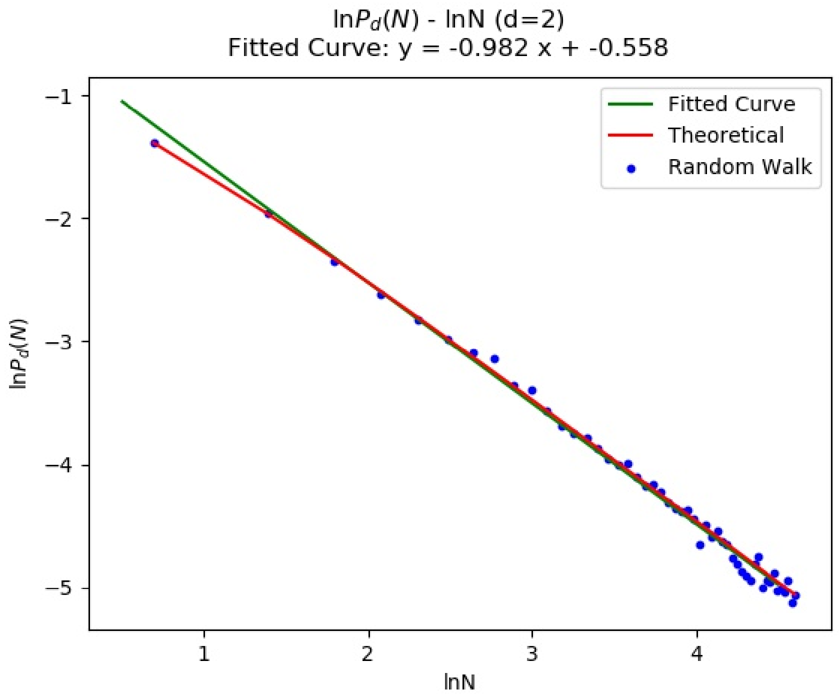

图 7: $d = 2$时对数图像线性拟合

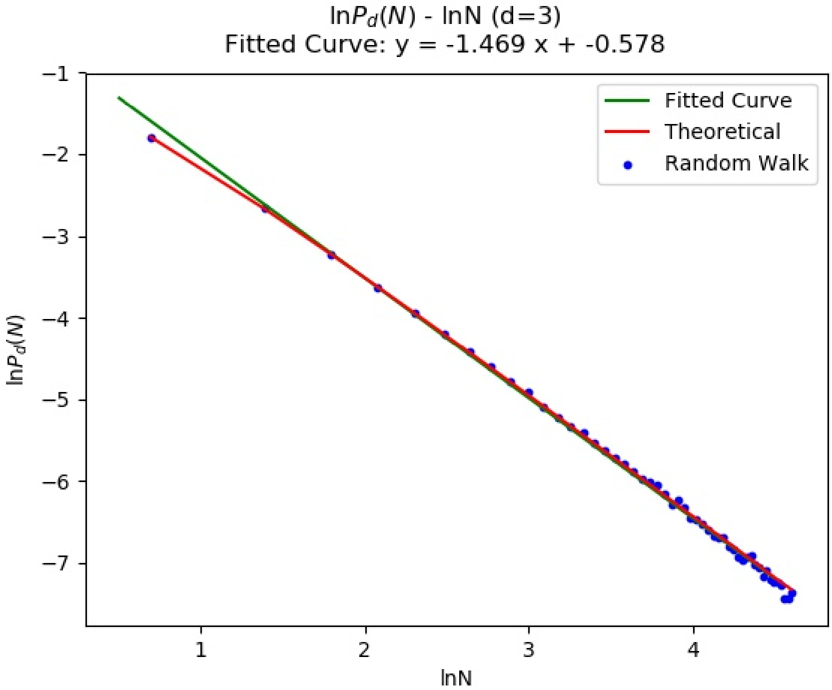

图 8: $d=3$时对数图像线性拟合

5 讨论

5.1 原理的讨论

本题只考虑了等概率等距离的随机行走问题。对于不是等概率等距离的问题，就会十分复杂。运用概率论的知识，依然可以求出理论上返回原点的概率，但将无法定义相关的指数值。

例如：$d=1$时等距离的随机行走，返回原点的概率为：

$$P_d = \frac{N!}{[(\frac{N}{2})!]^2} p^{\frac{N}{2}} (1-p)^{\frac{N}{2}} \tag{14}$$

这是可以在理论上得到的解析值。

但对于指数的计算，因为计算N很大时的近似值为：

$$\begin{aligned} P_d &= \frac{N!}{[(\frac{N}{2})!]^2} p^{\frac{N}{2}} (1-p)^{\frac{N}{2}} \\ &\approx \frac{\sqrt{2\pi N}(\frac{N}{e})^N}{\left[\sqrt{2\pi\frac{N}{2}}(\frac{N}{2e})^{\frac{N}{2}}\right]^2} p^{\frac{N}{2}} (1-p)^{\frac{N}{2}} \\ &= \sqrt{\frac{4}{\pi}} \left[\frac{p(1-p)}{4}\right]^{\frac{N}{2}} N^{-\frac{1}{2}} \end{aligned} \tag{15}$$

求对数后如下：

$$\ln P_d = -\frac{1}{2}\ln N + \frac{\ln\frac{p(1-p)}{4}}{2} N + \ln\sqrt{\frac{4}{\pi}} \tag{16}$$

不是单纯的与$\ln N$的一次函数的形式，因此不能定义相关的指数值。

5.2 算法的改进

在编程前已经进行了分析，得到步数为奇数时返回原点的概率为0。因此设置的步数和步长都是偶数。当上限一定时，这样可以大约减少一半的程序运行时间。虽然不是指数级的减少，但在时间较长时也有比较大的效果。

由于符合统计规律，因此需要设置较大的粒子数，不然会有较大的误差。特别对于$d = 3$的情况，由于本身衰减就很快，如果设置的粒子数很小，很可能得到的返回原点的概率都是0。因此需要设置很大的粒子数的值来保证得到的结果比较精确。

5.3 结果的讨论

5.3.1 返回原点的概率P_d随步数N的变换关系$P_d(N)$

从图中可以看出，随着步数的增加，回到原点的概率逐渐减小。这也符合二项分布的结论，因为回到原点的概率仅仅是二项分布中的一项，随着二项分布个数的增加，这个概率将会越来越小。

此外，可以看出相同步数N下，随着维数d的增大，返回原点的概率P_d将会减小。直觉上来分析，增加维数，需要在增加的维数上也使其回到原点，因此将会使返回原点的概率减小。

5.3.2 指数的求解

从图中可以看出，得到的点确实在一次直线的附近，拟合的结果与原理部分叙述的理论上的推导结果$-\frac{1}{2}, -1, -\frac{3}{2}$基本一致。即可以认为N足够大时，维数d的随机行走返回原点的概率近似为：

$$P_d \propto N^{-\frac{d}{2}} \tag{17}$$

但曲线的阶距与理论值的差别很大。主要是因为N最大也只有100，Sterling公式要求$N \to \infty$，因此阶距与理论值相差很大。但指数值却与Sterling公式符合的较好，主要是因为N较小时，对Sterling公式的影响主要是系数的值的影响。