南 京 邮 电 大 学

实 验 报 告

课程名称： 计算物理实践

专　　业： 应用物理学

学　　号： B16080611

姓　　名： 王松乐

完成日期： 2019 年 6 月

目　录

第一章　简单物理实验的模拟及实验数据处理 3

1.1 行驻波的动态演示 3

1.2 单摆运动演示 6

1.3 最小二乘法拟合 7

第二章　方程组的数值解法 9

2.1 牛顿法迭代 9

2.2 两种近似求解方法 10

2.2.1 Jacobi 方法 10

2.2.2 Gauss-Seidel 方法 11

第三章　静电场问题 12

3.1 接地金属槽 12

3.2 同轴金属槽 14

第四章　热传导方程 16

4.1 一维热传导 16

4.2 二维热传导 18

第五章 蒙特卡洛法 20

5.1 蒙特卡洛的原理 20

5.2 蒙特卡洛的应用 20

结束语 23

致谢 23

参考文献 24

附录 25

第一章 简单物理实验的模拟及实验数据处理

1.1 行驻波的动态演示

- **矩形波导的原理**

通常将由金属材料制成的、矩形截面的、内充空气的规则金属波导称为矩形波导，我们设矩形波导的宽边尺寸为 a，窄边尺寸为 b，并建立如图 1 所示的坐标：

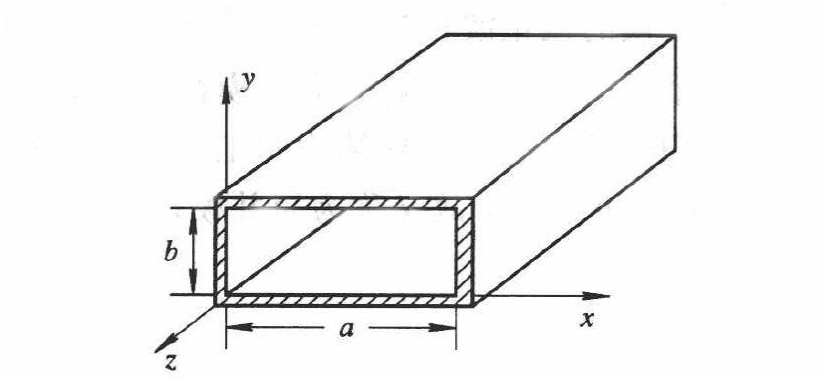

图 1. 矩形波导示意图

我们可以根据麦克斯韦方程，解出无源场电场和磁场应该满足的方程：

$$\begin{cases} \nabla^2 \vec{E} + k^2 \vec{E} = 0 \\ \nabla^2 \vec{H} + k^2 \vec{H} = 0 \end{cases}$$

通过代入边界条件解方程，我们可以得到 TE 波场分量的表达式：

$$\begin{cases} E_x = \sum_{m=0}^{\infty}\sum_{n=0}^{\infty} \frac{j\omega\mu}{k_c^2}\frac{n\pi}{b} H_{mn} \cos\left(\frac{m\pi}{a}x\right)\sin\left(\frac{n\pi}{b}y\right)\mathrm{e}^{-\mathrm{j}\beta z} \\ E_y = \sum_{m=0}^{\infty}\sum_{n=0}^{\infty} \frac{-j\omega\mu}{k_c^2}\frac{m\pi}{a} H_{mn} \sin\left(\frac{m\pi}{a}x\right)\cos\left(\frac{n\pi}{b}y\right)\mathrm{e}^{-\mathrm{j}\beta z} \\ E_z = 0 \\ H_z = \sum_{m=0}^{\infty}\sum_{n=0}^{\infty} \frac{\mathrm{j}\beta}{k_c^2}\frac{m\pi}{a} H_{mn} \sin\left(\frac{m\pi}{a}x\right)\cos\left(\frac{n\pi}{b}y\right)\mathrm{e}^{-\mathrm{j}\beta z} \\ H_y = \sum_{m=0}^{\infty}\sum_{n=0}^{\infty} \frac{\mathrm{j}\beta}{k_c^2}\frac{n\pi}{b} H_{mn} \cos\left(\frac{m\pi}{a}x\right)\sin\left(\frac{n\pi}{b}y\right)\mathrm{e}^{-\mathrm{j}\beta z} \end{cases}$$

通过类似的方式我们不难解得 TM 的方程。

- **行驻波理论**

根据课本电磁场沿不同方向的波的结构是不同的，分别满足两种不同形式的正弦波运动，下列是正弦波的两种不同的运动形式：

$$\begin{cases} u = \sin(\omega t \pm \beta x) \\ u = \sin(\omega t)\cos(\beta x) \end{cases}$$

其中第一种运动方式称为行波，第二种运动方式称为驻波。

- **TE_{10} 模的动态演示**

下面我们将以 *TE_{10}* 为例动态展示矩形波导中的行波与驻波状态，根据教材 *P_{46}*，我们可以代入m、n、k_c三个参数，得到 *TE_{10}* 模的各场分量表达式满足如下关系（忽略前面常数的条件下，我们定性进行讨论）：

$$\begin{cases} E_y \propto \sin\left(\frac{\pi}{a}x\right)\cos\left(\omega t - \beta z - \frac{\pi}{2}\right) \\ H_x \propto \sin\left(\frac{\pi}{a}x\right)\cos\left(\omega t - \beta z + \frac{\pi}{2}\right) \\ H_z \propto \cos\left(\frac{\pi}{a}x\right)\cos(\omega t - \beta z) \\ E_x = E_z = H_y = 0 \end{cases}$$

通过上面解与行驻波进行对比，我们不难发现，电磁场沿着 x 方向是满足驻波形式的解，电磁场沿着 z 方向满足行波形式的解。

为了验证我们我们的理论，根据上述表达式，我们可以通过 Matlab 对电场和磁场进行表示。由于我们的结果与时间有关，所以，这里我们截取连续一段时间的图片作为文档演示：

- **沿 z 轴方向（行波演示截图）**

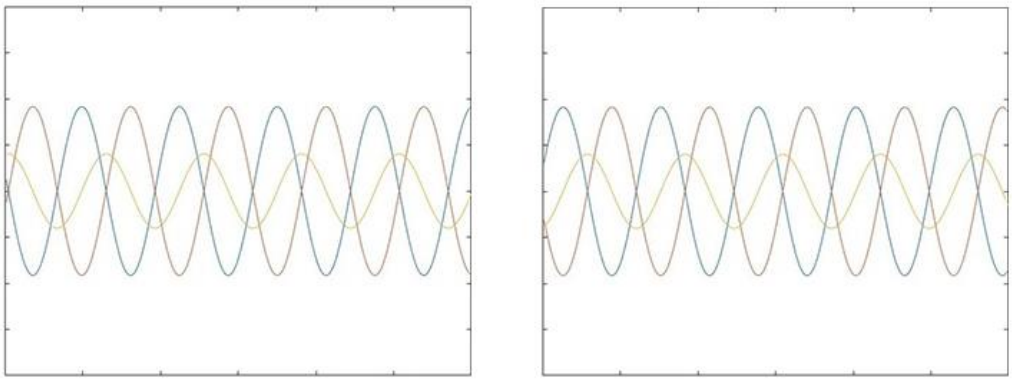

图 2.　t_1（左）、t_2（右）时刻电磁场的 Ey、Hx、Hz 随着 z 轴的变化的波形

我们可以发现电磁场各分量的波形沿着 z 方向正向传播，波形不变，这满足我们理论中的行波特点，为了使得波形更具有空间特征，我们将 H 的空间变化图进行了动态演示，该图是 Hx-Hz 平面内分量沿着 z 轴的变化图：

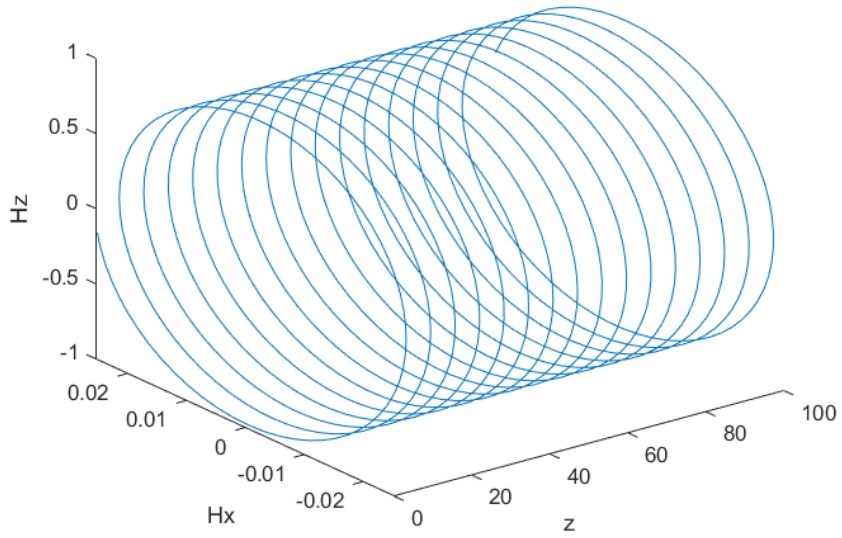

图 3. 行波的 H 的空间动态演示截图

- **沿 x 轴方向（驻波演示截图）**

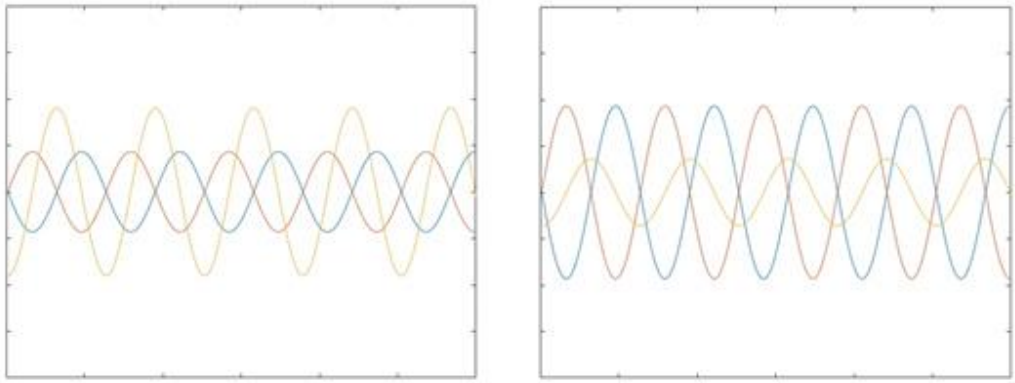

图 4. t1（左）、t2（右）时刻电磁场的 Ey、Hx、Hz 随着 x 轴的变化的波形

我们可以发现波谷和波峰在 x 方向上的分布是不变的，只是波的幅值随着时间变化，这是满足驻波特点的，同样的我们也对驻波的实际空间分量进行了动态演示，演示结果如下：

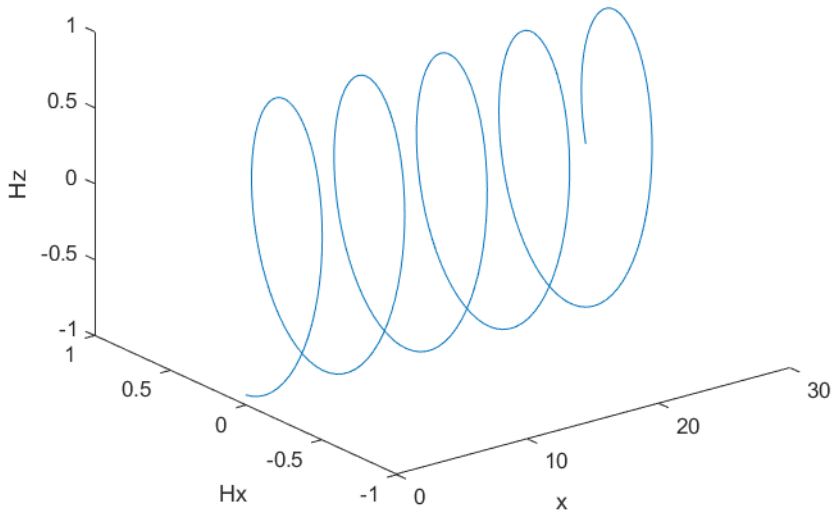

图 5. 驻波的空间分量动态演示

1.2 单摆运动演示

单摆是一种能够往复摆动的装置，将无重量细杆或不可伸长的细绳悬于重力场内一点，另一端固定一个重球即可得到。

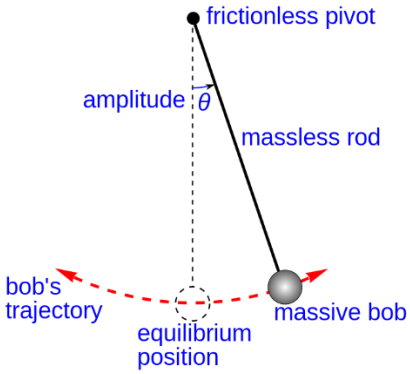

图 6. 单摆运动的基本原理

在这一部分，我们演示的是理想的平面单摆，通过力矩和角动量定理我们不难推导得到单摆方程(在摆动角度较小的条件下满足：$\sin\theta \approx \theta$)

$$\frac{d^2\theta}{dt^2} + \frac{g}{l}\theta = 0$$

通过二阶常系数线性齐次微分方程的理论，我们可以得到单摆的解为：

$$\theta = A\cos\left(\sqrt{\frac{g}{l}}t + \varphi\right)$$

该解属于近似解，通常当我们满足$\theta < 5°$,即可满足误差在千分之以下，由此我们可以计算得到小球坐标随着时间变化的函数，我们用 Matlab 进行编码演示得到如下结果：

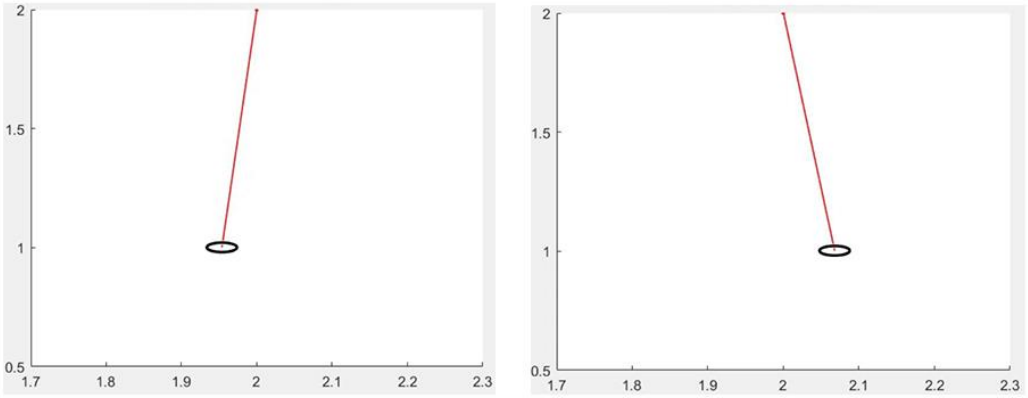

图 7 单摆运动演示动画截图

1.3 最小二乘法拟合

- **求解原理**

最小二乘法（又称最小平方法）是一种数学优化技术。它通过最小化误差的平方和寻找数据的最佳函数匹配，通常用于寻找满足方程的最优参数，基本原理如下：

在我们研究两个变量(x, y)之间的相互关系时，通常可以得到N个成对的实验数据(x_1, y_1)、(x_2, y_2)、(x_3, y_3)、……、$(x_n$、$y_n)$。我们可以将这些数据描绘在 x-y 直角坐标系中，若发现这些点在一条直线附近，可以用直线方程去描述这些点的分布：

$$y'(x') = a_0 x' + b_0$$

为了建立这条方程，我们就需要确定 a0 和 b0，应用最小二乘法原理我们利用实验值与理论值的离差的平方和最小作为最优判据，即满足下面等式：

$$\min \quad k = \sum_i^n \left(y_i - y'(x_i)\right)^2$$

我们可以发现由于 k 是最小值，所以 k 对 a0、b0 的偏导数应该为零，这样我们就可以得到两个方程组:

$$\begin{cases} \dfrac{dk}{da_0} = \sum_i^n -2 * (y_i - a_0 x_i - b_0) * x_i = 0 \\ \dfrac{dk}{db_0} = \sum_i^n -2 * (y_i - a_0 x_i - b_0) = 0 \end{cases}$$

由此我们可以解得到:

$$\begin{cases} b_0 = \sum_i^n \dfrac{(y_i - a_0 x_i)}{n} \\ a_0 = \dfrac{n\sum_i^n x_i y_i - \sum_i^n x_i \sum_i^n y_i}{n\sum_i^n x_i^2 - (\sum_i^n x_i)^2} \end{cases}$$

根据题目已知条件，我们得到下面五组数据：

(1,0.2339) (2,0.3812) (3,0.5759) (4,0.8153) (5,0.9742)

代入上面的计算公式，我们不难计算得到方程的解和曲线：

$$y = 0.1915x \ + 0.02169$$

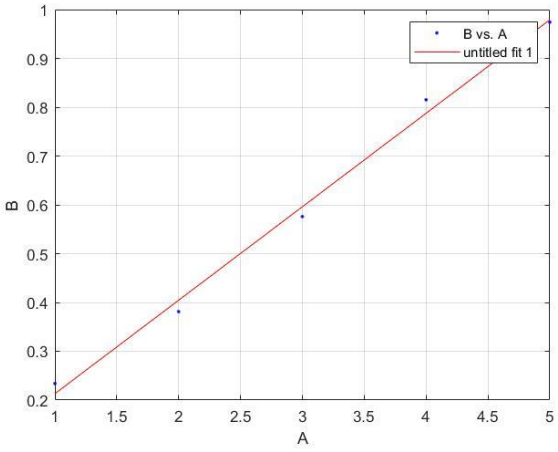

图 8 最小二乘拟合结果曲线

除了上面的方法我们同样也可以代入 Curve Fitting toolbox 进行求解，得到的结果与上面几乎一致：

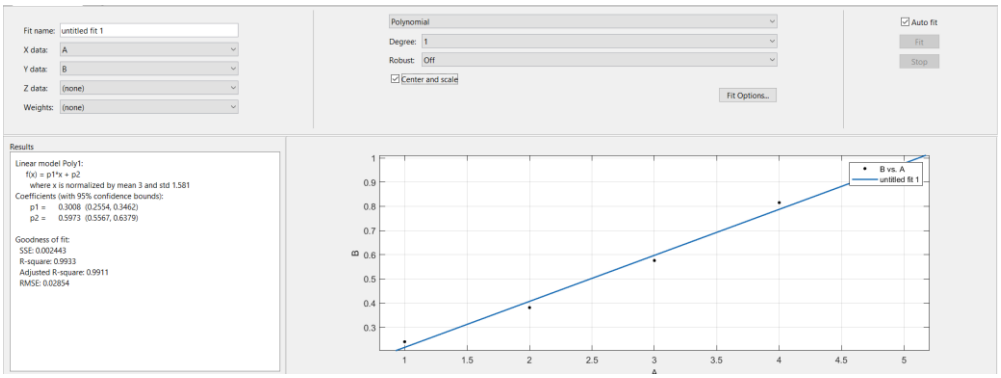

图 9 cftool 工具箱拟合结果

第二章 方程组的数值解法

2.1 牛顿法迭代

牛顿法（英语：Newton's method）又称为牛顿-拉弗森方法（英语：Newton-Raphson method），它是一种在实数域和复数域上近似求解方程的方法。方法使用函数 f(x) 的泰勒级数的前面几项来寻找方程 f(y) = 0 的根，基本原理如下：

假设有一个非线性函数 f(x) = 0, 在x_0处通过级数展开，取其线性部分，作为非线性方程的近似方程，则有:

$$f(x_0) + (x - x_0)f'(x_0) = 0$$

设$f'(x_0) \neq 0$，则其解为:

$$x = x_1 = x_0 - \frac{f(x_0)}{f'(x_0)}$$

因为这是利用泰勒公式的一阶展开，$f(x) = f(x_0) + (x - x_0)f'(x_0)$ 并不是完全相等而是近似相等，这里的 x_1 还不能让 f(x) = 0, 只是说更接近我们要的解了，所以我们将 f(x) 在 x_1 处泰勒展开，重复上述过程，进行 n 次，代入牛顿迭代公式

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

通过迭代，这个式子必然在 f(x)=0 的时候收敛，我们可以通过设置精度来结束迭代。

- **仿真参数**

仿真函数：$xe^x - 1 = f(x)$

初始值为 1，精度设为 0.000001

迭代次数上限 100

- **仿真结果**

迭代次数=18 , x=0.5671438

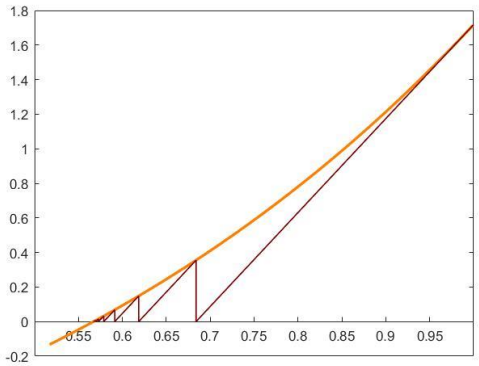

图 10. 牛顿法迭代仿真结果图

结果注意：由于 matlab 自身的精度显示有限，我们的小数点精确度超过 6 位已无法再进一步判别，但是根据 double 的精度上限，我们计算机内部实际

上是可以判断超过十位小数的迭代结果，但是在这里是不必要的，所以我们建议设置参数时精度不超过六位小数，该结果迭代速度较快，满足收敛要求，所以仿真结果是准确的。

2.2 两种近似求解方法

2.2.1 Jacobi 方法

- **基本原理**

对于一个具有普遍形式的线性方程组

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n = b_1 \\ a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n = b_2 \\ \cdots \\ a_{n1}x_1 + a_{n2}x_2 + \ldots + a_{nn}x_n = b_n \end{cases}$$

通过将 x 的某一项移项，将方程组进行变形，我们可以得到 Jacobi 迭代公式的一般形式：

$$\begin{cases} x^{(0)} = \left(x_1^{(0)}, x_2^{(0)}, \ldots, x_n^{(0)}\right)^T \\ x_i^{(k+1)} = \frac{1}{a_{ii}}\left(b_i - \sum_{i=1}^{n} a_{ij}x_j^{(k)}\right) \end{cases}$$

我们可以将方程组即为 Ax=b，其中我们可将 A 分裂成 D、L、U，分别为下三角矩阵、对角矩阵、上三角矩阵，通过矩阵表示我们可以得到矩阵迭代公式：

$$x^{(k+1)} = -D^{-1}(L+U)x^{(k)} + D^{-1}b$$

- **迭代结果**

针对第二部分的第三问，我们将下列参数代入 Jacobi 方法中进行计算：

$$A = \begin{bmatrix} 10 & -1 & 0 \\ -1 & 10 & -2 \\ 0 & -2 & 10 \end{bmatrix} \quad B = [9 \quad 7 \quad 6]^T$$

设定精度为 10^{-4},初值为 1，代入雅克比迭代公式，迭代 6 次得到结果:

$$\mathrm{x} = [0.9958 \quad 0.9579 \quad 0.7916]^{\mathrm{T}}$$

2.2.2 Gauss-Seidel 方法

- **基本原理**

对于同样形式的一般的线性方程组，我们可以通过下面的公式进行迭代：

$$x_m^{k+1} = \frac{1}{a_{mm}}\left(b_m - \sum_{j=1}^{m-1} a_{mj}\cdot x_j^{k+1} - \sum_{j=m+1}^{n} a_{mj}\cdot x_j^{k}\right), \quad 1 \le m \le n$$

在求 k+1 步近似值时，我们利用第 m 个方程求解第 m 个未知量。在求解过程中，所有已解出的 k+1 步元素都被直接使用，重复上述的求解过程，可以得到一个线性方程组解的近似值数列，设置误差值，如果方程最终有解，则该方法最终会收敛在一个稳定值，该方法要求主对角元素非零，满足要求。

- **迭代结果与比较**

Gauss-Seidel 通过迭代 9 次，得到的结果是：

$$\mathrm{X} = [0.9958 \quad 0.9579 \quad 0.7916]^{\mathrm{T}}$$

这与之前使用 Jacobi 迭代法求得的结果完全一致，两种方法在解决线性方程组都是有效的，只是适用范围不太一样，这种近似方法通常都是用于加快解算大型方程组的运算。

在我们区分两者哪个更优的情况下，主要关注发散和收敛的特性。Jacobi 迭代和 Gauss-Seidel 迭代可能同时发散，也可能同时收敛，但是一个快，一个慢；也可能一个收敛、一个发散，它们的收敛范围并不重合，但两者同时收敛时，很多情况下 Gauss-Seidel 方法要比 Jacobi 收敛要快，尤其对于大型矩阵运算来说，在我们这一题中由于运算纬度较小，没有显著的区别。

第三章 静电场问题

3.1 接地金属槽

设一个长直接接地金属槽，如图所示，其侧壁和地面电位为零，顶盖电位为 $\varphi = 100 \cdot \sin\pi x$，求槽内电位分布。

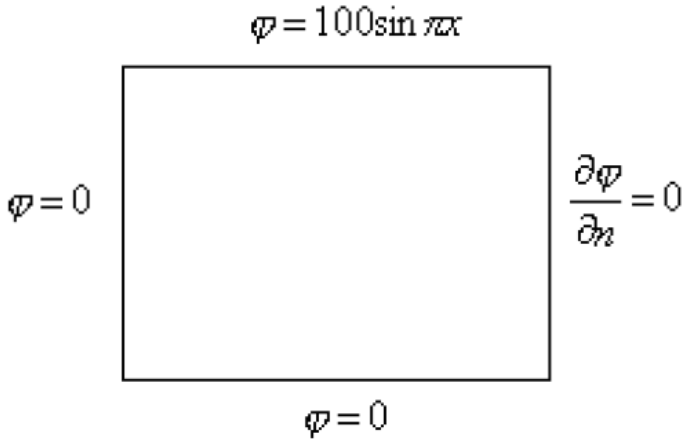

图 11. 接地金属槽边界示意图

- **基本原理**

对于金属槽而言，其电位函数满足无源的拉普拉斯方程，即满足：

$$\nabla^2 \varphi = 0$$

由图描述得到，我们不难得到左边界和下边界的电位应该满足 Dirichlet 边界条件始终为 0，右边界满足 Neuman 边界条件电位导数为 0，表达成如下形式：

$$\left.\frac{\partial \varphi}{\partial y}\right|_{y=y_1} = 0, \varphi|_{y=y_0} = 0$$
$$\varphi|_{x=x_0} = 0, \varphi|_{x=x_1} = 100\sin \pi x$$

这题属于混合型边值问题，只要用 Partial Differential Equation Toolbox(PDE)就可以快速得到数值解结果，我们也同样可以调用 PDE 相关的函数进行仿真，详细代码见附录，我们这里利用图形化界面进行仿真演示。

- **仿真演示**

打开 Matlab 的偏微分方程工具箱（输入命令：pdetool），打开界面后，进行如下设置：

1. 网格设置（设置 x、y 在合适范围即可，划分网格清晰）
2. 绘制模型：选择 Draw 菜单的 Rectangle/Square ，在中央区域绘制如图所示的矩形（具体尺寸不影响）

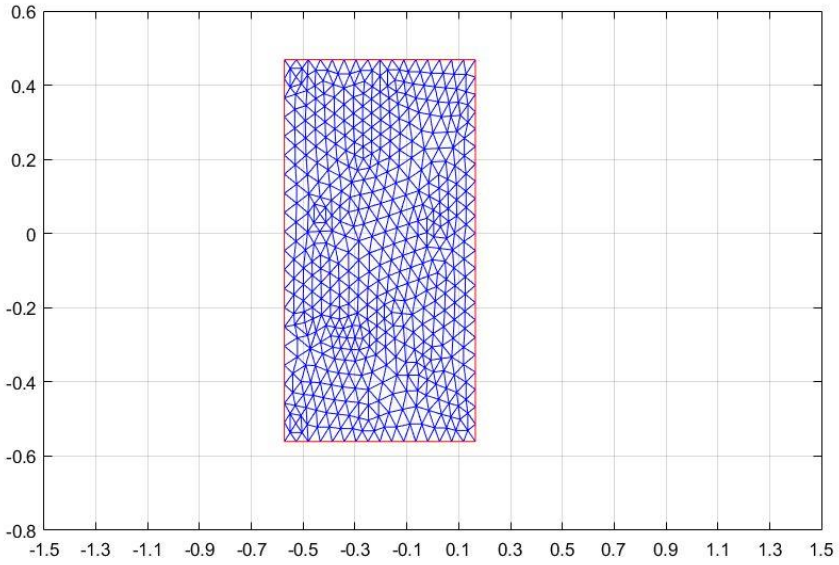

图 12. 接地金属槽仿真模型图

3. 模型类型选择：在应用模式中选择 Electrostatics 模式
4. 输入边界条件：进入 Boundary Mode，输入：
   a) 左边界：Dirichlet 条件：h=1，r=0;
   b) 右边界：Neuman 条件：g=0，q=0;
   c) 上边界：Dirichlet 条件：h=1，r=100*sin(pi*x)
   d) 下边界：Dirichlet 条件：h=1，r=0;
5. 方程参数设定：选择 PDE Spacification：
   a) 设置 Epsilon=1
   b) 设置 rho=0
6. 图形解析：生成网格，适当加密之后，通过 Plot 菜单中的参数设置，选择合适的显示参数，然后 Plot，就可以得到如下结果：

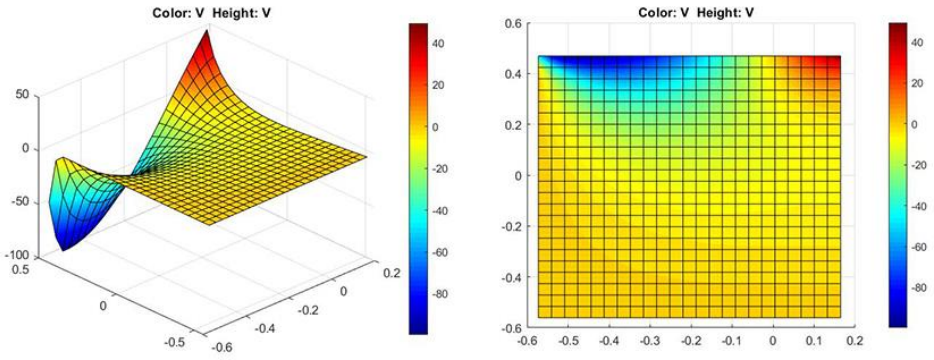

图 11. 接地金属槽数值解三维图（左）及顶部投影图（右）

3.2 同轴金属槽

设两个同轴矩形金属槽如图 12 所示，外金属槽电位为 0，内金属电位为 100V，求槽内电位分布，并绘出电位分布图。

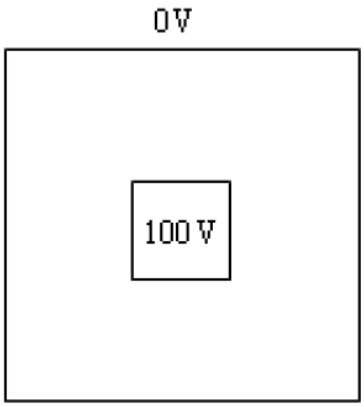

图 13. 同轴金属槽边界示意图

- **原理**

该题的基本原理与之前的题是一致，电磁场依然满足无源条件的拉普拉斯方程，只是边界条件有所不同：

$$\varphi|_{\text{Outter}} = 0, \varphi|_{\text{inner}} = 100$$

这是不规则边界的第一类边界问题，我们仍然可以通过 PDE 工具箱对其进行求解。

- **仿真步骤**

打开 Matlab 的偏微分方程工具箱（输入命令：pdetool），打开界面后，进行如下设置：

1. 网格设置（设置 x、y 在合适范围即可，划分网格清晰）
2. 绘制模型：选择 Draw 菜单的 Rectangle/Square ，在中央区域绘制如图所示的矩形（需要保证两个矩形中心重合，最好画成方形），绘制完成后在上面 Set Formula 栏中输入 R1-R2。

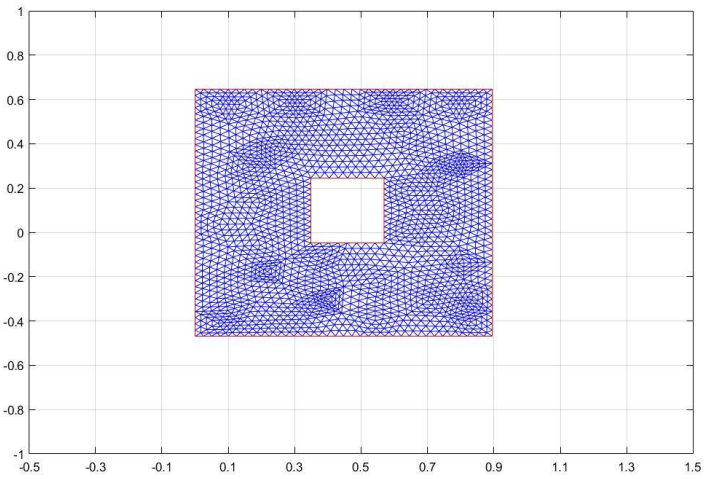

图 14. 同轴金属槽仿真模型图

3. 模型类型选择：在应用模式中选择 Electrostatics 模式
4. 输入边界条件：进入 Boundary Mode，输入：
   a) 内边界：Dirichlet 条件：h=1，r=100;（四条边界四次操作）
   b) 外边界：Dirichlet 条件：h=1，r=0; （四条边界四次操作）
5. 方程参数设定：选择 PDE Spacification：
   a) 设置 Epsilon=1
   b) 设置 rho=0
6. 图形解析：生成网格，适当加密之后，通过 Plot 菜单中的参数设置，选择合适的显示参数，然后 Plot，就可以得到如下结果：

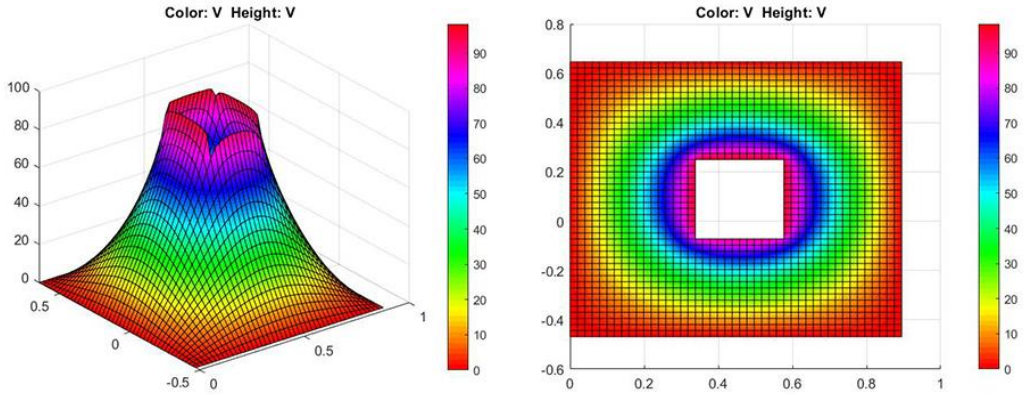

图 15. 同轴金属槽中的电势分布

第四章 热传导方程

4.1 一维热传导

求热传导方程混合问题的数值解，其中 N，h，k 等参数自取。

$$\begin{cases} u_t = u_{xx} & 0 < x < 1, t > 0 \\ u(x,0) = x^2 & 0 \le x \le 1 \\ u(0,t) = 0, u(1,t) = 1 & t \ge 0 \end{cases}$$

- **原理**

这是典型的一维热传导问题，所给方程中，给定了两处边界条件为 Dirichlet 条件，并且给定了初始状态满足的方程，其中温度只与 x 和 t 有关，我们可以通过分离变量求出解析解，由于本题要求数值解，我们可以通过 PDE toolbox 工具箱进行求解。

- **仿真求解步骤**

打开 Matlab 的偏微分方程工具箱（输入命令：pdetool），打开界面后，进行如下设置：

1. 网格设置（设置 x、y 在合适范围即可，划分网格清晰）
2. 绘制模型：选择 Draw 菜单的 Rectangle/Square ，矩形务必绘制在 x∈(0,1)范围内，y 轴范围不影响模型仿真结果，我们通过 Dirichlet 条件可以将二维情况降到一维结果。

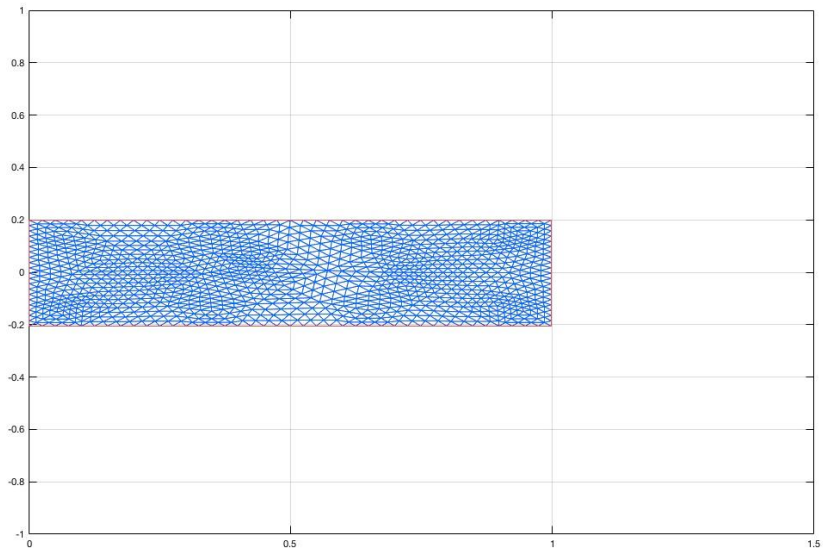

图 16. 同轴金属槽仿真模型图

3. 模型类型选择：在应用模式中选择 Heat Transfer 模式
4. 输入边界条件：进入 Boundary Mode，输入：
   a) 左边界：Dirichlet 条件：h=1，r=0;
   b) 右边界：Dirichlet 条件：h=1，r=1;
   c) 上下边界：Neumann 条件：g=0，q=0;
5. 方程参数设定：选择 PDE Spacification：
   a) 选择 Parabolic（抛物线形方程）
   b) 设置 Rho=C=k=1
   c) 设置 Q=h=0
6. 设置 Solve Parameters 中 $u(t_0)$=x.^2
7. 图形解析：生成网格，适当加密之后，通过 Plot 菜单中的参数设置，选择合适的显示参数，然后 Plot，就可以得到如下结果：

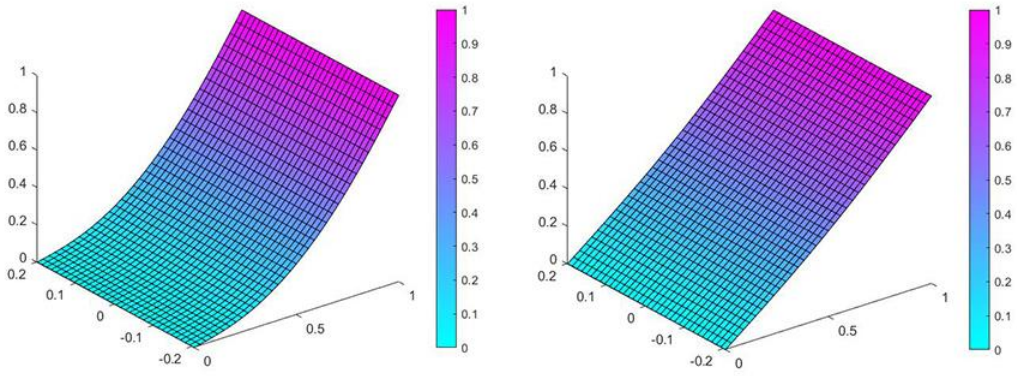

图 17. 一维热传导热分布初始状态（左）和结束状态（右）

4.2 二维热传导

求有限空间内的热传导问题，求如下方程的数值解，边界条件如教材所述

$$\begin{cases} u_{tt} = u_{xx} + u_{yy} \\ u(x, y, 0) = xy \end{cases}$$

- **原理**

对于题目表述的边界条件我们没有找到明确的表述，我们采取书中的一个例子给出的边界条件，我们认为 y=y0,y1 边界上的温度始终为零，为恒温源,对应 Dirichlet 条件为零；x=x0，x1 边界满足绝热条件即 Neuman 条件，导数为 0。如下给出数学表述：

$$\begin{cases} \left.\dfrac{\partial \varphi}{\partial x}\right|_{x=x_0,x_1} = 0 \\ \left.\varphi\right|_{y=y_0,y_1} = 0 \end{cases}$$

- **仿真过程简述**

打开 Matlab 的偏微分方程工具箱（输入命令：pdetool），打开界面后，进行如下设置：

1. 网格设置（设置 x、y 在合适范围即可，划分网格清晰）
2. 绘制模型：选择 Draw 菜单的 Rectangle/Square,在中央区域绘制如图所示的矩形（具体尺寸不影响）

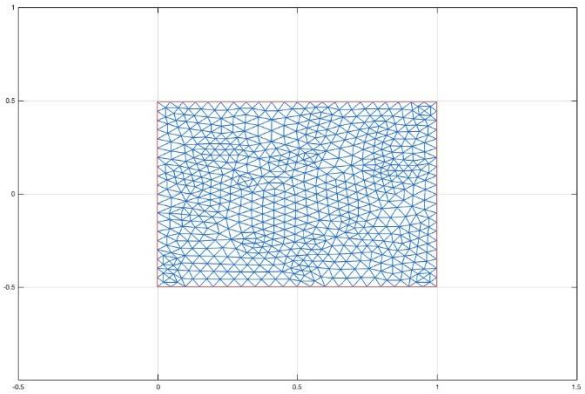

图 18. 二维热传导模型示意图

3. 模型类型选择：在应用模式中选择 Heat Transfer 模式
4. 输入边界条件：进入 Boundary Mode，输入：
   a) 左右边界：Neumann 条件：g=0，q=0;

b) 上下边界：Dirichlet 条件：h=1，r=0;

5. 方程参数设定：选择 PDE Spacification：
   c) 选择 Parabolic（抛物线形方程）
   d) 设置 Rho=C=k=1
   e) 设置 Q=h=0
6. 设置 Solve Parameters 中 $u(t_0)$=x.*y
7. 图形解析：生成网格，适当加密之后，通过 Plot 菜单中的参数设置，选择合适的显示参数，然后 Plot，就可以得到如下结果：

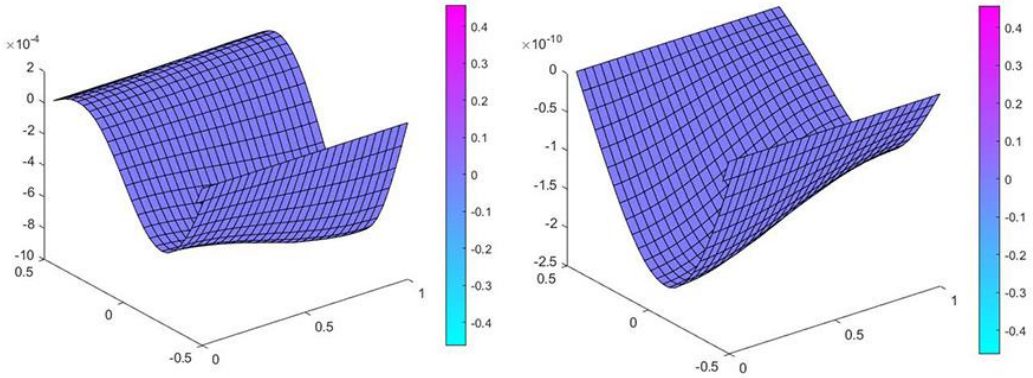

图 19. 0.15s（左）和 1.5s 时刻热分布情况

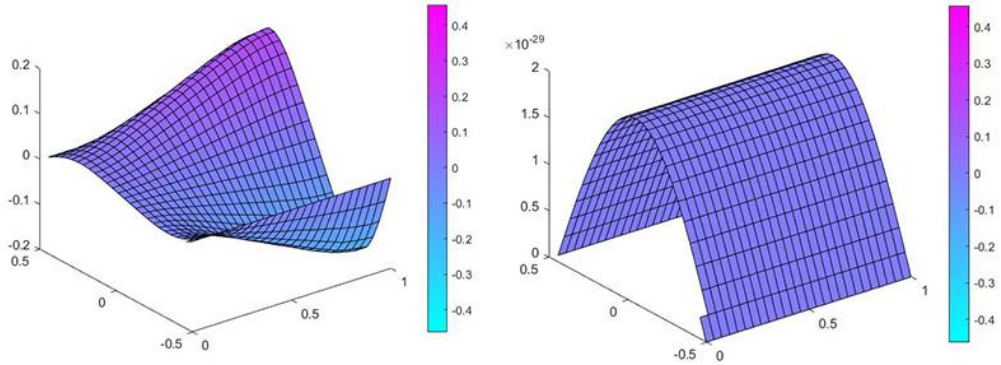

图 20. 初始时刻和 500s 时刻热分布情况

第五章 蒙特卡洛法

5.1 蒙特卡洛的原理

蒙特卡罗方法（Monte Carlo method），也称统计模拟方法，是 1940 年代中期由于科学技术的发展和电子计算机的发明，而提出的一种以概率统计理论为指导的数值计算方法。是指使用随机数（或更常见的伪随机数）来解决很多计算问题的方法。

通常蒙特卡罗方法可以粗略地分成两类：一类是所求解的问题本身具有内在的随机性，借助计算机的运算能力可以直接模拟这种随机的过程。

另一种类型是所求解问题可以转化为某种随机分布的特征数，比如随机事件出现的概率，或者随机变量的期望值。通过随机抽样的方法，以随机事件出现的频率估计其概率，或者以抽样的数字特征估算随机变量的数字特征，并将其作为问题的解。这种方法多用于求解复杂的多维积分问题。

在第二种类型中，蒙特卡洛通常都可以起到非常有效的降维效果，虽然收敛速度较慢，仍然具有十分高效的求解作用。

5.2 蒙特卡洛的应用

一球体的半径 R=0.5，球上有一个半径 r=0.3m 的圆柱形空洞，其轴线与球直径重合。试用 M-C 方法求实体的体积。

为了求解这个问题，我们首先要对图像有一个深入的认识，这里我们采用 3ds max 进行建模描述，如图所示：

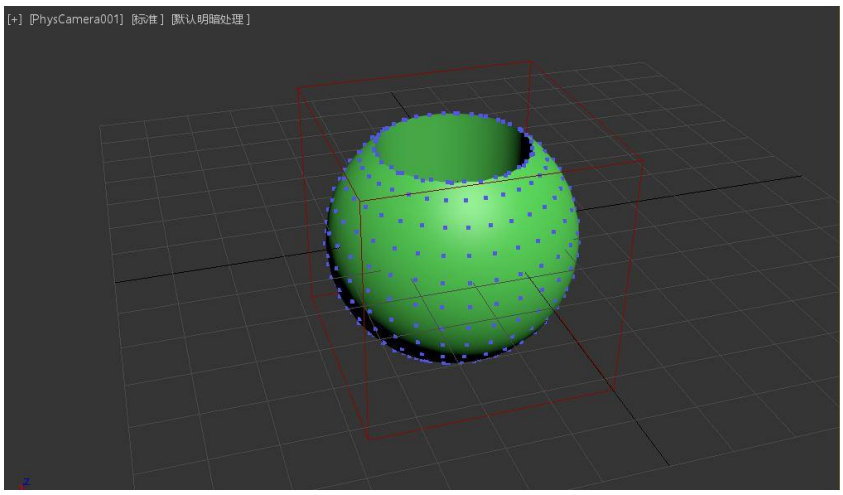

图 21. 3DS Max 图像建模图

在如图所示的立方体中，有一个被圆柱挖空的球体，我们可以发现，我们通过球和圆柱的积分差是可以求出这部分体积的，但是这样的积分比较复杂，而且圆柱和球的常用积分变量也不一样，转换起来比较麻烦，复杂度远远大于一般的三重积分问题，这里我们通过蒙特卡洛的思想，可以对这部分体积进行求解。

● 仿真过程

首先，我们不难得到和球体相切的正方体的体积为 1，假设空间中存在一点 a（x,y），在已知球心坐标（x0,y0）,和柱心坐标（x1，y1）的情况下不难求解得到 a 点距离球心、柱心的距离为 d1、d2，通过圆柱半径和圆心我们可以判断出该点是否处于实体部分。

那么我们不妨设想，立方空间中假设有均匀的 N 个点，那么如果实体的体积为 V，我们不难得到处于实体中的点的个数为 V*N。反过来想，如果我们能够实现均匀的撒点，并且判断处于实体的点的个数那么我们就可以求出体积 V.大致原理图可以由下图所示，

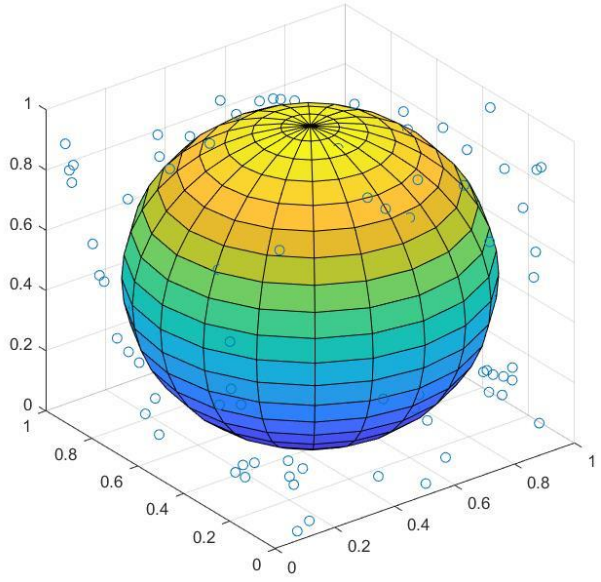

图 22. 蒙特卡洛撒点原理图（简化）

上图所示是没有挖去柱形的判断情况，但基本逻辑和算法是一样的，只是判断条件不同罢了，下面给出 M-C 算法的流程图。

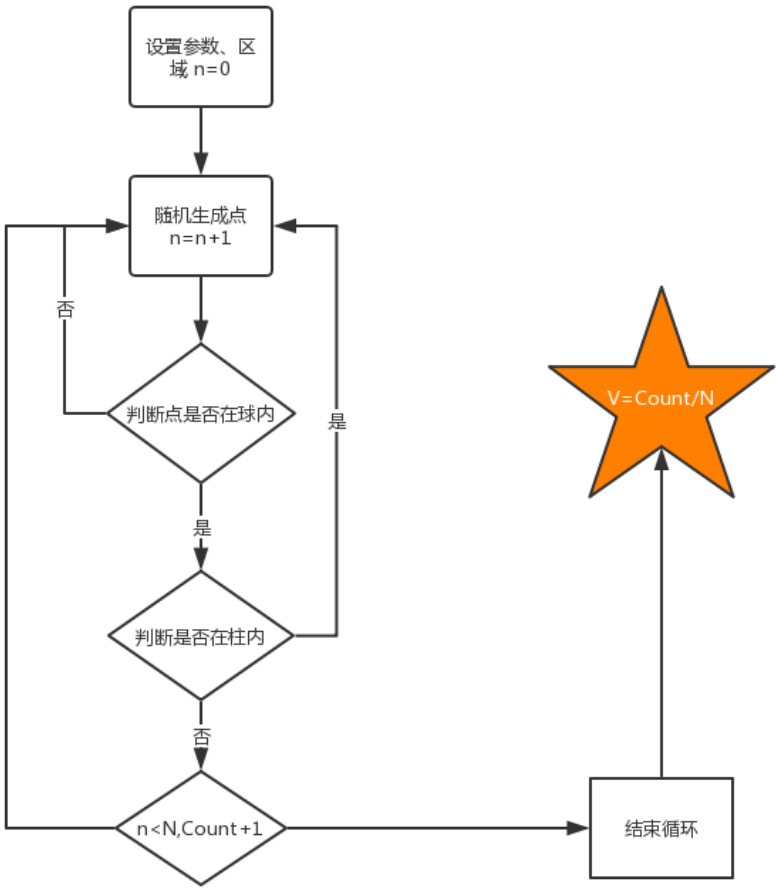

图 23. 蒙特卡洛算法流程图

蒙特卡洛的解的可靠性往往取决于 N 的大小，为了更加准确计算得到我们想要的体积结果，我们对不同的 N 进行了测试，结果如下：

表 1. 不同次数的蒙特卡洛仿真结果

Count	256	2677	26850	268435	2680430
N	1000	10000	100000	1000000	10000000
V（m^3）	0.2560	0.2677	0.2685	0.2684	0.2680

如表所示，我们十万次以上的蒙特卡洛随机的结果是相近的，所以我们可以认为体积为 0.2680~0.2685,最大误差为 0.18% << 1%，这样的近似结果基本是可以接受的，当然我们认为 0.2680 这个结果是更加可信的。

结束语

在这次计算物理实验中，打心底说是收获颇多，更多的是一种总结的感悟了。这次实验和撰写整个报告大致耗费我一天半的时间，提前在端午假期完成了两周实验周的内容。这高效快速的进度主要取决于之前在数模比赛中积累的经验，对 Matlab 的熟练掌握以及对模型的深入掌握，尤其是曲线拟合还有偏微分方程的求解这部分内容，都是之前在数模训练中都有比较多的接触了，本来以为会花费更多的时间的，但是现在感觉就是轻车熟路了。

当然除了对之前学习的巩固以外，我也同样学到了不少新东西，比如 matlab 的动态模型演示，之前从来没有想过还能制作动画模型，这次是学到了，而且还掌握了一些相关的命令比如，movie, sphere 等等，对 matlab 的掌握又更进一步了。在这次实验中，我应该算选题目比较多的了，但是对我来说其实难度不算很大，包括对一些辅助软件的掌握，比如 3DS MAX，流程图软件等等，这些都对我理解模型和求解提供了不少的帮助。数值分析的课程，我大概算是在大二期间，已经自学过半了，这次补充学习，也算是后续课程了。

这次其实感触最大的就是，自己基本上已经熟练掌握 PDE 工具箱了，说实话，之前看到这个工具箱是比较害怕的。因为，感觉数学物理方法这门课不太好学，而且掌握的也不好，但是没想到 PDE 工具箱以及相关的函数已经比较好的能够快捷地求解数值解了，现在也算是跨过了这道心理门槛。

说实话，我觉得这门实验课应该早点开，如果放在大二开的话可能对我做数模会有不少的帮助，现在学的感觉像是一个总结性的课程，我以后是想做物理的计算方面的科研工作，但是我感觉现在书本里写的更像是简单的物理实验仿真，而且更多的是数值分析方面的内容，更有点偏向应用数学方面的内容（个人感觉），可能我印象中的计算物理包含理论的部分会比较多的原因。

致谢

完成这次实验，我不得不感谢我的数模导师和数模队友，在他们的陪伴下，我才把这么多的相关的 matlab 和模型知识消化理解。这才有我今天能够如此快速地完成这次实验，撰写完成这份报告。当然提供实验教学和实验内容的老师，也同样给了我大量的帮助，在此致谢。

注：动态演示视频（以附件形式发至邮箱，也可在线查看）：

https://github.com/lemoxiao/Computational-Physics

参考文献

[1] 计算物理学，哈尔滨工业大学出版社，程锺贤编著
[2] 数值分析，华中科技大学出版社，李庆杨等编著
[3] 微波技术与天线，西安电子科技大学出版社，刘学观等编著
[4] 电磁场数值计算与 MATLAB 实现，华中科技大学出版社，何红雨等编著
[5] 数学物理方程的 Matlab 解法与可视化， 清华大学出版社， 彭芳麟编著
[6] MATLAB 数值分析与应用， 机械工业出版社，宋叶志等编著
[7] 偏微分方程的 matlab 解法， 武汉大学出版社， 路君安等编著
[8] 电磁场数值计算法与 Matlab 实现， 华中科技大学出版社， 何红雨编著

附录

1. 驻波动态演示

```
% Standing_wave.m
%Display parameter
t=0:0.03:10000;
x=0:0.1:50;
z=rand(1)*pi;
%Wave Parameter_voltage
Omega =3/4*pi;
Phi = rand(1)*2*pi;
beta=1;
a=pi;
%Omega is circular frequency;
%Phi is time factor

for i=1:length(t)
Ex=0;
Ey=sin(pi/a*x)*cos(Omega*t(i)-beta*z-pi/2);
Ez=0;
Hx=sin(pi/a*x)*cos(Omega*t(i)-beta*z+pi/2);
Hy=0;
Hz=cos(pi/a*x)*cos(Omega*t(i)-beta*z);
plot(Ey);hold on
plot(Hx);hold on
plot(Hz);hold on
xlim([0 300])
ylim([-2 2])
hold on
pause(0.001);
hold off
end
```

2. 行波动态演示

```
%Display parameter
t=0:0.05:10000;
z=0:0.1:50;
x=rand(1)*pi;
%Wave Parameter_voltage
```

```
Omega =3/4*pi;
Phi = rand(1)*2*pi;
beta=1;
a=1;
%Omega is circular frequency;
%Phi is time factor

for i=1:length(t)
Ex=0;
Ey=sin(pi/a*x)*cos(Omega*t(i)-beta*z-pi/2);
Ez=0;
Hx=sin(pi/a*x)*cos(Omega*t(i)-beta*z+pi/2);
Hy=0;
Hz=cos(pi/a*x)*cos(Omega*t(i)-beta*z);
plot(Ey);hold on
plot(Hx);hold on
plot(Hz);hold on
xlim([0 300])
ylim([-2 2])
hold on
pause(0.001);
hold off
end
```

3. 单摆动态演示

```
clear;clc;
%pendulum parameter
g=9.8;
L=1;
Phi=rand(1)*2*pi;
t=0:0.01:20
Theta_max = 4;
%theta must less than 5 degree
if Theta_max>5
    fprintf("theta overflow");
    return
end
A= 2*pi/360*Theta_max;
%theta equation

for i=1:length(t)
```

```
    axis([1.7 2.3 0.5  2]);
    theta = A*cos(sqrt(g/L)*t(i)+Phi);
    %Node coordinates
    x1 = 2;
    y1 = 2;
    %Spherical coordinates
    x2 = x1+sin(theta)*L;
    y2 = y1-cos(theta)*L;
    Node_circle= viscircles([x1 y1],0.001)
    wire = line([x1 x2],[y1
y2],'LineWidth',1,'Color','red');
    Spherical_circle = viscircles([x2
y2],0.02,'Color','black');
    pause(0.001);
    if i<length(t)
        delete(Node_circle);
        delete(wire);
        delete(Spherical_circle);
    end
end
```

4. 最小二乘法拟合

```
A=[ 1     2     3     4     5];
B=[    0.2339    0.3812    0.5759    0.8153
0.9742];
%% Fit: 'untitled fit 1'.
[xData, yData] = prepareCurveData( A, B );
% Set up fittype and options.
ft = fittype( 'poly1' );
% Fit model to data.
[fitresult, gof] = fit( xData, yData, ft );
% Plot fit with data.
figure( 'Name', 'untitled fit 1' );
h = plot( fitresult, xData, yData );
legend( h, 'B vs. A', 'untitled fit 1',
'Location', 'NorthEast' );
% Label axes
xlabel A
ylabel B
grid on
```

5. Jacobi 迭代

```
clear;clc
%function [x,k]=Fjacobi(A,b,x0,tol)
%jacobi iteration
A=[10 -1 0;-1 10 -2; 0 -2 10];
b=[9 7 6]';
x0=[1 1 1]';
tol=1e-4;
% tol is errir tolerance ,x0 is initial value
maxl=300;       %max iterations
D=diag(diag(A));
L=-tril(A,-1);
U=-triu(A,1);
B=D\(L+U);
f=D\b;
x=B*x0+f;
k=1;

while norm(x-x0)>=tol
    x0=x;
    x=B*x+f;
    k=k+1;
    if(k>=maxl)
        disp('Unconvergence');
        return;
    end
    %[k x']
end

% check A\b
```

6. G_S 迭代

```
function x=Gauss_Seidel(A,b)
[m,n]=size(A);
x0=zeros(n,1);
x=zeros(n,1);
k=0;
for i=1:n
    x(i)=b(i);
    for j=1:i-1
        x(i)=x(i)-A(i,j)*x(j);
```

```
    end
    for j=i+1:n
        x(i)=x(i)-A(i,j)*x0(j);
    end
    x(i)=x(i)/A(i,i);
end
x
while norm(x-x0)>1e-10
    k=k+1;
    x0=x;
    for i=1:n
        x(i)=b(i);
        for j=1:i-1
            x(i)=x(i)-A(i,j)*x(j);
        end
        for j=i+1:n
            x(i)=x(i)-A(i,j)*x0(j);
        end
        x(i)=x(i)/A(i,i);
    end
    x
end
k
```

7. 接地金属槽

```
% This script is written and read by pdetool and
should NOT be edited.
% There are two recommended alternatives:
 % 1) Export the required variables from pdetool
and create a MATLAB script
 %    to perform operations on these.
 % 2) Define the problem completely using a MATLAB
script. See
 %
http://www.mathworks.com/help/pde/examples/index.h
tml for examples
 %    of this approach.
function pdemodel
[pde_fig,ax]=pdeinit;
pdetool('appl_cb',5);
set(ax,'DataAspectRatio',[1 1 1]);
set(ax,'PlotBoxAspectRatio',[1.5 1 1]);
```

```
set(ax,'XLim',[-1.5 1.5]);
set(ax,'YLim',[-1 1]);
set(ax,'XTick',[ -1.5,...
 -1.3,...
 -1.1000000000000001,...
 -0.89999999999999991,...
 -0.69999999999999996,...
 -0.5,...
 -0.29999999999999982,...
 -0.099999999999999867,...
 0.099999999999999867,...
 0.29999999999999982,...
 0.5,...
 0.69999999999999996,...
 0.89999999999999991,...
 1.1000000000000001,...
 1.3,...
 1.5,...
]);
set(ax,'YTickMode','auto');
pdetool('gridon','on');

% Geometry description:
pderect([-0.57236842105263164 0.16447368421052655
0.46874999999999978 -0.56085526315789513],'R1');
set(findobj(get(pde_fig,'Children'),'Tag','PDEEval
'),'String','R1')

% Boundary conditions:
pdetool('changemode',0)
pdesetbd(4,...
'dir',...
1,...
'1',...
'0')
pdesetbd(3,...
'dir',...
1,...
'1',...
'0')
pdesetbd(2,...
'neu',...
1,...
'0',...
```

```
'0')
pdesetbd(1,...
'dir',...
1,...
'1',...
'100*sin(pi*x)')

% Mesh generation:
setappdata(pde_fig,'Hgrad',1.3);
setappdata(pde_fig,'refinemethod','regular');
setappdata(pde_fig,'jiggle',char('on','mean',''));
setappdata(pde_fig,'MesherVersion','preR2013a');
pdetool('initmesh')
pdetool('refine')

% PDE coefficients:
pdeseteq(1,...
'1.0',...
'0.0',...
'0',...
'1.0',...
'0:10',...
'0.0',...
'0.0',...
'[0 100]')
setappdata(pde_fig,'currparam',...
['1.0';...
'0  '])

% Solve parameters:
setappdata(pde_fig,'solveparam',...
char('0','1458','10','pdeadworst',...
'0.5','longest','0','1E-4','','fixed','Inf'))

% Plotflags and user data strings:
setappdata(pde_fig,'plotflags',[1 1 1 1 1 1 7 1 1
1 0 1 1 0 1 0 0 1]);
setappdata(pde_fig,'colstring','');
setappdata(pde_fig,'arrowstring','');
setappdata(pde_fig,'deformstring','');
setappdata(pde_fig,'heightstring','');

% Solve PDE:
pdetool('solve')
```

8. 同轴金属槽

```
   % This script is written and read by pdetool
and should NOT be edited.
% There are two recommended alternatives:
 % 1) Export the required variables from pdetool
and create a MATLAB script
 %    to perform operations on these.
 % 2) Define the problem completely using a MATLAB
script. See
 %
http://www.mathworks.com/help/pde/examples/index.h
tml for examples
 %    of this approach.
function pdemodel
[pde_fig,ax]=pdeinit;
pdetool('appl_cb',5);
set(ax,'DataAspectRatio',[1 1 1]);
set(ax,'PlotBoxAspectRatio',[1.5 1 1]);
set(ax,'XLim',[-1.5 1.5]);
set(ax,'YLim',[-1 1]);
set(ax,'XTick',[ -1.5,...
 -1.3,...
 -1.1000000000000001,...
 -0.89999999999999991,...
 -0.69999999999999996,...
 -0.5,...
 -0.29999999999999982,...
 -0.099999999999999867,...
 0.099999999999999867,...
 0.29999999999999982,...
 0.5,...
 0.69999999999999996,...
 0.89999999999999991,...
 1.1000000000000001,...
 1.3,...
 1.5,...
]);
set(ax,'YTickMode','auto');
pdetool('gridon','on');

% Geometry description:
```

```
pderect([2.2204460492503131e-16
0.89473684210526327 0.64638157894736792 -
0.46875000000000067],'R1');
pderect([0.34868421052631615 0.56907894736842146
0.24506578947368429 -0.047697368421052433],'R2');
set(findobj(get(pde_fig,'Children'),'Tag','PDEEval
'),'String','R1-R2')

% Boundary conditions:
pdetool('changemode',0)
pdesetbd(8,...
'dir',...
1,...
'1',...
'100')
pdesetbd(7,...
'dir',...
1,...
'1',...
'0')
pdesetbd(6,...
'dir',...
1,...
'1',...
'0')
pdesetbd(5,...
'dir',...
1,...
'1',...
'100')
pdesetbd(4,...
'dir',...
1,...
'1',...
'100')
pdesetbd(3,...
'dir',...
1,...
'1',...
'100')
pdesetbd(2,...
'dir',...
1,...
'1',...
```

```
'0')
pdesetbd(1,...
'dir',...
1,...
'1',...
'0')

% Mesh generation:
setappdata(pde_fig,'Hgrad',1.3);
setappdata(pde_fig,'refinemethod','regular');
setappdata(pde_fig,'jiggle',char('on','mean',''));
setappdata(pde_fig,'MesherVersion','preR2013a');
pdetool('initmesh')
pdetool('refine')
pdetool('refine')

% PDE coefficients:
pdeseteq(1,...
'1.0',...
'0.0',...
'0',...
'1.0',...
'0:10',...
'0.0',...
'0.0',...
'[0 100]')
setappdata(pde_fig,'currparam',...
['1.0';...
'0  '])

% Solve parameters:
setappdata(pde_fig,'solveparam',...
char('0','5904','10','pdeadworst',...
'0.5','longest','0','1E-4','','fixed','Inf'))

% Plotflags and user data strings:
setappdata(pde_fig,'plotflags',[1 1 1 1 1 1 8 1 1
1 0 1 1 0 1 0 0 1]);
setappdata(pde_fig,'colstring','');
setappdata(pde_fig,'arrowstring','');
setappdata(pde_fig,'deformstring','');
setappdata(pde_fig,'heightstring','');

% Solve PDE:
```

```
pdetool('solve')
```

9. 一维热传导混合边界

```
% This script is written and read by pdetool and
should NOT be edited.
% There are two recommended alternatives:
 % 1) Export the required variables from pdetool
and create a MATLAB script
 %    to perform operations on these.
 % 2) Define the problem completely using a MATLAB
script. See
 %
http://www.mathworks.com/help/pde/examples/index.h
tml for examples
 %    of this approach.
function pdemodel
[pde_fig,ax]=pdeinit;
pdetool('appl_cb',9);
set(ax,'DataAspectRatio',[1 2 1]);
set(ax,'PlotBoxAspectRatio',[1 0.66666666666666663
1.3333333333333333]);
set(ax,'XLim',[0 1.5]);
set(ax,'YLim',[-1 1]);
set(ax,'XTick',[ 0,...
 0.5,...
 1,...
 1.5,...
]);
set(ax,'YTickMode','auto');
pdetool('gridon','on');

% Geometry description:
pderect([0.99835526315789469 1.6653345369377348e-
16 -0.20559210526315819
0.19901315789473673],'R1');
set(findobj(get(pde_fig,'Children'),'Tag','PDEEval
'),'String','R1')

% Boundary conditions:
pdetool('changemode',0)
pdesetbd(4,...
'dir',...
1,...
```

```
'1',...
'1')
pdesetbd(3,...
'neu',...
1,...
'0',...
'0')
pdesetbd(2,...
'dir',...
1,...
'1',...
'0')
pdesetbd(1,...
'neu',...
1,...
'0',...
'0')

% Mesh generation:
setappdata(pde_fig,'Hgrad',1.3);
setappdata(pde_fig,'refinemethod','regular');
setappdata(pde_fig,'jiggle',char('on','mean',''));
setappdata(pde_fig,'MesherVersion','preR2013a');
pdetool('initmesh')
pdetool('refine')
pdetool('refine')

% PDE coefficients:
pdeseteq(2,...
'1.0',...
'0',...
'(0)+(0).*(0.0)',...
'(1.0).*(1.0)',...
'0:0.01:0.2',...
'x.^2',...
'0.0',...
'[0 100]')
setappdata(pde_fig,'currparam',...
['1.0';...
'1.0';...
'1.0';...
'0  ';...
'0  ';...
'0.0'])
```

```
% Solve parameters:
setappdata(pde_fig,'solveparam',...
char('0','3456','10','pdeadworst',...
'0.5','longest','0','1E-4','','fixed','Inf'))

% Plotflags and user data strings:
setappdata(pde_fig,'plotflags',[1 1 1 1 1 1 1 1 1
1 1 21 1 0 1 0 0 1]);
setappdata(pde_fig,'colstring','');
setappdata(pde_fig,'arrowstring','');
setappdata(pde_fig,'deformstring','');
setappdata(pde_fig,'heightstring','');

% Solve PDE:
pdetool('solve')
```

10. 二维热简单混合边界

```
% This script is written and read by pdetool and
should NOT be edited.
% There are two recommended alternatives:
 % 1) Export the required variables from pdetool
and create a MATLAB script
 %    to perform operations on these.
 % 2) Define the problem completely using a MATLAB
script. See
 %
http://www.mathworks.com/help/pde/examples/index.h
tml for examples
 %    of this approach.
function pdemodel
[pde_fig,ax]=pdeinit;
pdetool('appl_cb',9);
set(ax,'DataAspectRatio',[1 1.5 1]);
set(ax,'PlotBoxAspectRatio',[1 0.66666666666666663
1]);
set(ax,'XLim',[-0.5 1.5]);
set(ax,'YLim',[-1 1]);
set(ax,'XTick',[ -0.5,...
 0,...
 0.5,...
 1,...
```

```
 1.5,...
 2,...
 2.5,...
 3,...
]);
set(ax,'YTick',[ -0.5,...
 0,...
 0.5,...
 1,...
 1.5,...
 2,...
 2.5,...
 3,...
]);
pdetool('gridon','on');

% Geometry description:
pderect([-0.0021929824561403577
0.99780701754386025 0.49177631578947345 -
0.49835526315789469],'R1');
set(findobj(get(pde_fig,'Children'),'Tag','PDEEval
'),'String','R1')

% Boundary conditions:
pdetool('changemode',0)
pdesetbd(4,...
'neu',...
1,...
'0',...
'0')
pdesetbd(3,...
'dir',...
1,...
'1',...
'0')
pdesetbd(2,...
'neu',...
1,...
'0',...
'0')
pdesetbd(1,...
'dir',...
1,...
'1',...
```

```
'0')

% Mesh generation:
setappdata(pde_fig,'Hgrad',1.3);
setappdata(pde_fig,'refinemethod','regular');
setappdata(pde_fig,'jiggle',char('on','mean',''));
setappdata(pde_fig,'MesherVersion','preR2013a');
pdetool('initmesh')
pdetool('refine')

% PDE coefficients:
pdeseteq(2,...
'1.0',...
'0',...
'(0)+(0).*(0.0)',...
'(1.0).*(1.0)',...
'0:0.1:2000',...
'x.*y',...
'0.0',...
'[0 100]')
setappdata(pde_fig,'currparam',...
['1.0';...
'1.0';...
'1.0';...
'0  ';...
'0  ';...
'0.0'])

% Solve parameters:
setappdata(pde_fig,'solveparam',...
char('0','1962','10','pdeadworst',...
'0.5','longest','0','1E-4','','fixed','Inf'))

% Plotflags and user data strings:
setappdata(pde_fig,'plotflags',[1 1 1 1 1 1 1 1 1
1 1 20001 1 0 1 0 0 1]);
setappdata(pde_fig,'colstring','');
setappdata(pde_fig,'arrowstring','');
setappdata(pde_fig,'deformstring','');
setappdata(pde_fig,'heightstring','');

% Solve PDE:
pdetool('solve')
```

11. 蒙特卡洛方法

```
clear;clc
n=100;
x(n)=0;
y(n)=0;
z(n)=0;
count=0;
for i=1:n
    x(i)=rand(1);
    y(i)=rand(1);
    z(i)=rand(1);
    d1(i)=sqrt((x(i)-0.5)^2+(y(i)-0.5)^2+(z(i)-
0.5)^2);
    d2(i)=sqrt((x(i)-0.5)^2+(y(i)-0.5)^2)
    if d1(i)<0.5&d2(i)>0.3
        count=count+1;
    end
end

save n_100.mat count


clear;clc
n=500;
x(n)=0;
y(n)=0;
z(n)=0;
count=0;
for i=1:n
    x(i)=rand(1);
    y(i)=rand(1);
    z(i)=rand(1);
    d1(i)=sqrt((x(i)-0.5)^2+(y(i)-0.5)^2+(z(i)-
0.5)^2);
    d2(i)=sqrt((x(i)-0.5)^2+(y(i)-0.5)^2)
    if d1(i)<0.5&d2(i)>0.3
        count=count+1;
    end
end

save n_500.mat count
clear;clc
```

```
n=5000;
x(n)=0;
y(n)=0;
z(n)=0;
count=0;
for i=1:n
    x(i)=rand(1);
    y(i)=rand(1);
    z(i)=rand(1);
    d1(i)=sqrt((x(i)-0.5)^2+(y(i)-0.5)^2+(z(i)-
0.5)^2);
    d2(i)=sqrt((x(i)-0.5)^2+(y(i)-0.5)^2)
    if d1(i)<0.5&d2(i)>0.3
        count=count+1;
    end
end

save n_5000.mat count

clear;clc
n=20000;
x(n)=0;
y(n)=0;
z(n)=0;
count=0;
for i=1:n
    x(i)=rand(1);
    y(i)=rand(1);
    z(i)=rand(1);
    d1(i)=sqrt((x(i)-0.5)^2+(y(i)-0.5)^2+(z(i)-
0.5)^2);
    d2(i)=sqrt((x(i)-0.5)^2+(y(i)-0.5)^2)
    if d1(i)<0.5&d2(i)>0.3
        count=count+1;
    end
end

save n_20000.mat count
```