有限差分法解波动方程

ZJ

2021 年 12 月 17 日

1　问题描述

在给定初值条件的情况下，通过数值方法近似求解波动方程:

$$\frac{\partial^2 u}{\partial t^2} = \Delta u = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 u}{\partial z^2}$$

在下文中，我们将 $u(x, y, z, t)$ 简记为 $u(\mathbf{x}, t)$.

1.1　问题区域

该问题的区域限定在一个立方体中:

$$\Omega = [0 \leqslant x \leqslant L_x] \times [0 \leqslant y \leqslant L_y] \times [0 \leqslant z \leqslant L_z]$$

1.2　初值条件

首先是时间上的初值条件：

$$\left.u\right|_{t=0} = \varphi(x, y, z)$$
$$\left.\frac{\partial u}{\partial t}\right|_{t=0} = 0$$

然后是空间上的初值条件，可以有两种情况：第一类齐次边界条件，也就是在边界上都为 0，

$$u(0, y, z, t) = 0, \quad u(L_x, y, z, t) = 0$$
$$u(x, 0, z, t) = 0, \quad u(x, L_y, z, t) = 0$$
$$u(x, y, 0, t) = 0, \quad u(x, y, L_z, t) = 0$$

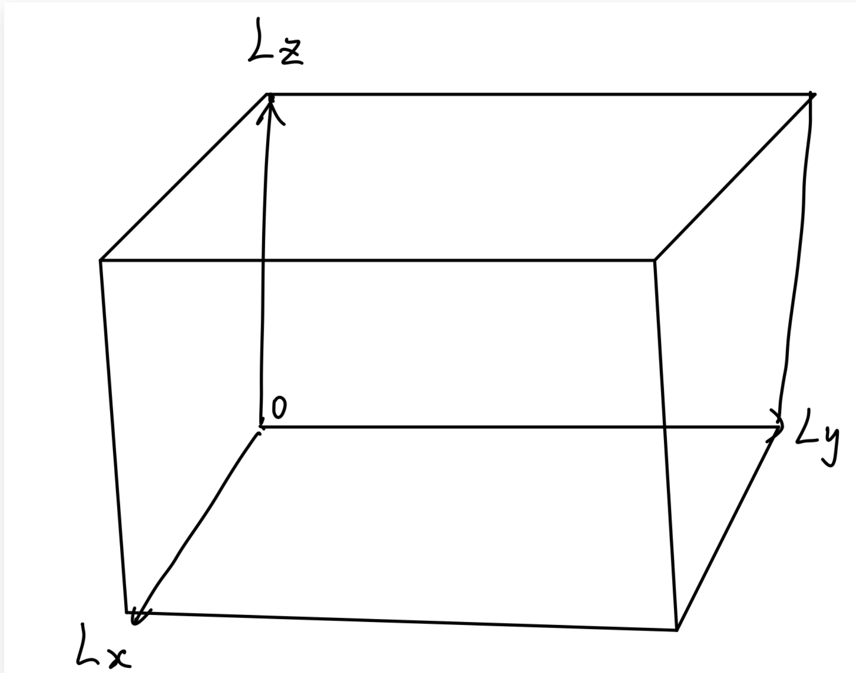

图 1: 问题区域

或周期性边界条件，

$$
\begin{aligned}
u(0, y, z, t) = u\left(L_x, y, z, t\right), \quad u_x(0, y, z, t) = u_x\left(L_x, y, z, t\right) \\
u(x, 0, z, t) = u\left(x, L_y, z, t\right), \quad u_y(x, 0, z, t) = u_y\left(x, L_y, z, t\right) \\
u(x, y, 0, t) = u\left(x, y, L_z, t\right), \quad u_z(x, y, 0, t) = u_z\left(x, y, L_z, t\right)
\end{aligned}
$$

2 数值求解

2.1 差分法

所谓差分就是用数值的方法来近似计算梯度:

$$
\begin{aligned}
\frac{\partial f(x)}{\partial x} &= \lim_{\Delta x \rightarrow 0} \frac{f(x+\Delta x)-f(x)}{\Delta x} \\
&\approx \frac{f(x+\Delta x)-f(x)}{\Delta x}
\end{aligned}
$$

$$
\begin{aligned}
\frac{\partial^2 f}{\partial x^2} &= \lim_{\Delta x \to 0} \frac{f'(x+\Delta x) - f'(x)}{\Delta x} \\
&\approx \frac{\frac{f(x+\Delta x)-f(x)}{\Delta x} - \frac{f(x)-f(x-\Delta x)}{\Delta x}}{\Delta x} \\
&= \frac{f(x+\Delta x) + f(x-\Delta x) - 2f(x)}{\Delta x^2}
\end{aligned}
$$

2.2 网格划分

我们将立方体区域 Ω 划分为 $N \times N \times N$ 的立方体网格，每个网格的尺寸为 $h_x \times h_y \times h_z$. 我们以每个网格的顶点作为计算节点，如图2 所示.

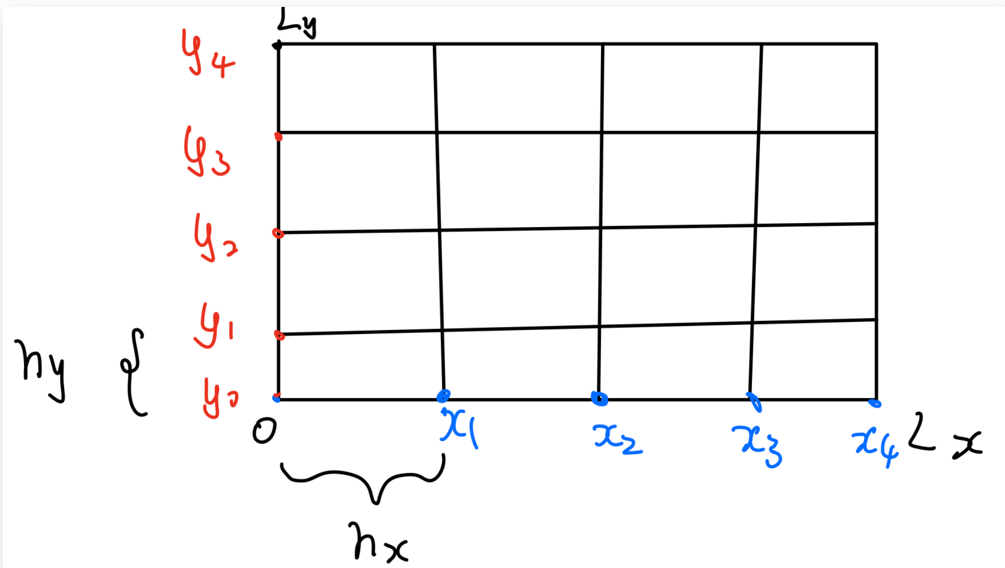

图 2: 以 $N = 4$ 的二维情况为例

$$
\bar{\omega}_h = \{(x_i = ih_x, y_j = jh_y, z_k = kh_z)\,, i, j, k = 0, 1, \ldots, N, h_x N = L_x, h_y N = L_y, h_z N = L_z\}
$$

$$
\omega_\tau = \{t_n = n\tau, n = 0, 1, \ldots, K, \tau K = T\}\,.
$$

上面的公式中，$\bar{\omega}_h$ 表示网格区域的划分，ω_τ 表示时间上的划分，每个时间节点之间的跨度为 τ.

2.3 差分模拟

我们要通过数值近似模拟波动方程：

$$\frac{\partial^2 u}{\partial t^2} = \Delta u = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 u}{\partial z^2}$$

首先将函数 $u(x, y, z, t)$ 在节点 (x_i, y_j, z_k, t_n) 处取值，记为：

$$u(x_i, y_j, z_k, t_n) = u_{ijk}^n$$

这里的下标 i, j, k 表示在空间上的 (x_i, y_j, z_k) 处取值，上标 n 表示此时的时间为 t_n.

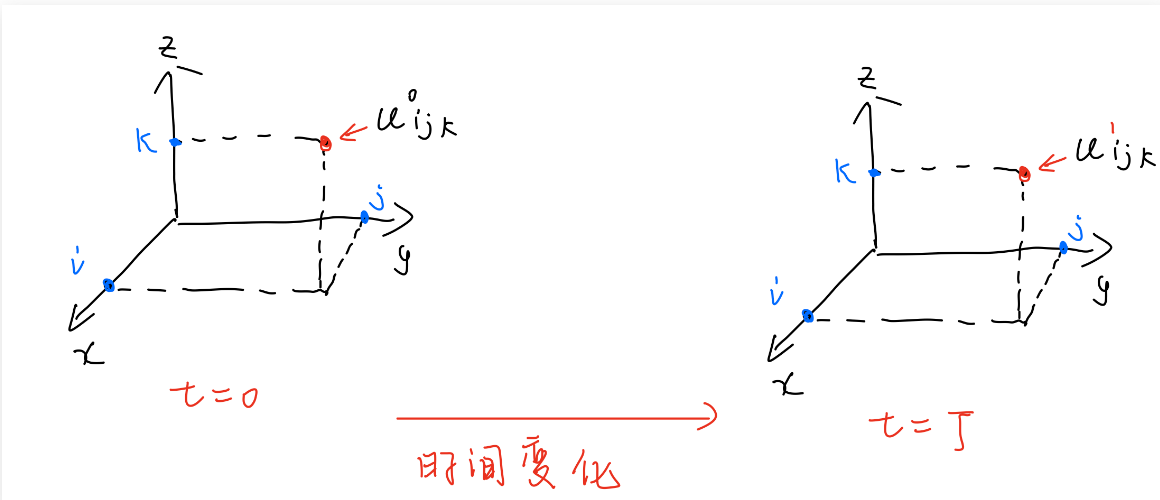

图 3: u_{ijk}^n

2.3.1 等式左侧的模拟

我们先对等式左侧进行差分模拟：

$$\frac{\partial^2 u(\mathbf{x}, t)}{\partial t^2} \approx \frac{u(\mathbf{x}, t + \Delta t) + u(\mathbf{x}, t - \Delta t) - 2u(\mathbf{x}, t)}{\Delta t^2}$$

体现在网格上就是：

$$\frac{\partial^2 u(\mathbf{x}, t_n)}{\partial t^2} \approx \frac{u(\mathbf{x}, t_{n+1}) + u(\mathbf{x}, t_{n-1}) - 2u(\mathbf{x}, t_n)}{\tau^2}$$

写成 PDF 中对应的形式即为：

$$\left.\frac{\partial^2 u(\mathbf{x}, t)}{\partial t^2}\right|_{u_{ijk}^n} \approx \frac{u_{ijk}^{n+1} - 2u_{ijk}^n + u_{ijk}^{n-1}}{\tau^2}$$

2.3.2 等式右侧的模拟

接下来是对等式右侧 $\Delta u=\frac{\partial^2 u}{\partial x^2}+\frac{\partial^2 u}{\partial y^2}+\frac{\partial^2 u}{\partial z^2}$ 的模拟，我们首先模拟 $\frac{\partial^2 u}{\partial x^2}$：

$$\frac{\partial^2 u(x,y,z,t)}{\partial x^2}\approx\frac{u(x+\Delta x,y,z,t)+u(x-\Delta x,y,z,t)-2u(x,y,z,t)}{\Delta x^2}$$

即:

$$\left.\frac{\partial^2 u}{\partial x^2}\right|_{u_{ijk}^n}\approx\frac{u_{i+1,j,k}^n-2u_{i,j,k}^n+u_{i-1,j,k}^n}{h_x^2}$$

类似的，我们可以求出 u 关于 y 和 z 的二阶导，整合后得到:

$$\Delta u_{ijk}^n\approx\frac{u_{i+1,j,k}^n-2u_{i,j,k}^n+u_{i-1,j,k}^n}{h_x^2}+\frac{u_{i,j+1,k}^n-2u_{i,j,k}^n+u_{i,j-1,k}^n}{h_y^2}+\frac{u_{i,j,k+1}^n-2u_{i,j,k}^n+u_{i,j,k-1}^n}{h_z^2}$$

2.4 迭代求解

我们通过波动方程的离散形式:

$$\left.\frac{\partial^2 u(\mathbf{x},t)}{\partial t^2}\right|_{u_{ijk}^n}=\Delta u_{ijk}^n$$

获得了方程组:

$$\frac{u_{ijk}^{n+1}-2u_{ijk}^n+u_{ijk}^{n-1}}{\tau^2}=\frac{u_{i+1,j,k}^n-2u_{i,j,k}^n+u_{i-1,j,k}^n}{h_x^2}+\frac{u_{i,j+1,k}^n-2u_{i,j,k}^n+u_{i,j-1,k}^n}{h_y^2}+\frac{u_{i,j,k+1}^n-2u_{i,j,k}^n+u_{i,j,k-1}^n}{h_z^2}$$

接下来我们对该方程组进行恒定变形，得到:

$$\begin{aligned}u_{ijk}^{n+1}&=2\left(1-\frac{\tau^2}{h_x^2}-\frac{\tau^2}{h_y^2}-\frac{\tau^2}{h_z^2}\right)u_{ijk}^n-u_{ijk}^{n-1}+\frac{\tau^2}{h_x^2}\left(u_{i+1,j,k}^n+u_{i-1,j,k}^n\right)\\&\quad+\frac{\tau^2}{h_y^2}\left(u_{i,j+1,k}^n+u_{i,j-1,k}^n\right)+\frac{\tau^2}{h_z^2}\left(u_{i,j,k+1}^n+u_{i,j,k-1}^n\right)\\&=2\left(1-r_x-r_y-r_z\right)u_{ijk}^n-u_{ijk}^{n-1}+r_x\left(u_{i+1,j,k}^n+u_{i-1,j,k}^n\right)\end{aligned}$$

$$\begin{aligned}u_{ijk}^{n+1}&=2\left(1-\frac{\tau^2}{h_x^2}-\frac{\tau^2}{h_y^2}-\frac{\tau^2}{h_z^2}\right)u_{ijk}^n-u_{ijk}^{n-1}+\frac{\tau^2}{h_x^2}\left(u_{i+1,j,k}^n+u_{i-1,j,k}^n\right)\\&\quad+\frac{\tau^2}{h_y^2}\left(u_{i,j+1,k}^n+u_{i,j-1,k}^n\right)+\frac{\tau^2}{h_z^2}\left(u_{i,j,k+1}^n+u_{i,j,k-1}^n\right)\\&=2\left(1-r_x-r_y-r_z\right)u_{ijk}^n-u_{ijk}^{n-1}+r_x\left(u_{i+1,j,k}^n+u_{i-1,j,k}^n\right)\\&\quad+r_y\left(u_{i,j+1,k}^n+u_{i,j-1,k}^n\right)+r_z\left(u_{i,j,k+1}^n+u_{i,j,k-1}^n\right)\end{aligned}$$

这里将 $\frac{\tau^2}{h_x^2}$ 记为 r_x，使得看起来更简洁。那么我们的迭代策略也很明显了:

$$u_{ijk}^{n+1}=2\left(1-r_x-r_y-r_z\right)u_{ijk}^n-u_{ijk}^{n-1}+r_x\left(u_{i+1,j,k}^n+u_{i-1,j,k}^n\right)+r_y\left(u_{i,j+1,k}^n+u_{i,j-1,k}^n\right)+r_z\left(u_{i,j,k+1}^n+u_{i,j,k-1}^n\right)$$

2.5 具体流程

1. 通过 $t=0$ 时的初值条件计算 $u_{ijk}^0=\varphi(x_i,y_j,z_k)$.

2. 要计算 u_{ijk}^1，如果通过上面的方程来计算的话，也就是 $n=0$ 的情况，此时需要 u^0 和 u^{-1}，但我们不知道 u^{-1} 的值，这时就需要进行一点修改，原方程为:

$$\frac{u^1-2u^0+u^{-1}}{\tau^2}=\Delta u^0$$

   我们可以假设 $u^1-u^0=u^0-u^{-1}$，这样就得到了:

$$\begin{aligned} u_{ijk}^1 &= u_{ijk}^0+\frac{\tau^2}{2}\Delta u_{ijk}^0 \\ &= u_{ijk}^0+\frac{\tau^2}{2}\Delta\varphi(x_i,y_j,z_k) \end{aligned}$$

3. 接下来要计算 $u_{ijk}^1, u_{ijk}^2, \ldots$，就可直接用迭代方程求解即可:

$$u_{ijk}^{n+1}=2\left(1-r_x-r_y-r_z\right)u_{ijk}^n-u_{ijk}^{n-1}+r_x\left(u_{i+1,j,k}^n+u_{i-1,j,k}^n\right)+r_y\left(u_{i,j+1,k}^n+u_{i,j-1,k}^n\right)+r_z\left(u_{i,j,k+1}^n+u_{i,j,k-1}^n\right)$$

3 实验

接下来用 Python 对二维情况进行模拟，这里使用二维情况是为了方便可视化，二维和三维的区别不大。

这里我们选择表格中的第一个函数（二维情况）进行模拟:

$$u(x,y,t)=\sin\left(\frac{\pi}{L_x}x\right)\cdot\sin\left(\frac{\pi}{L_y}y\right)\cdot\cos\left(a_t\cdot t\right),\quad a_t=\pi\sqrt{\frac{1}{L_x^2}+\frac{1}{L_y^2}}$$

那么我们的初值函数为:

$$\varphi(x,y)=\sin\left(\frac{\pi}{L_x}x\right)\cdot\sin\left(\frac{\pi}{L_y}y\right)$$

第一步是计算 u^0 的情况:

$$u(x_i,y_j,t=0)=\varphi(x_i,y_j)$$

第二步是计算 u^1，使用公式:

$$u^1=u^0+\frac{\tau^2}{2}\Delta\varphi(x_i,y_j)$$

这里初值函数的 Laplace 算子为:

$$\Delta\varphi(x,y) = -\left(\left(\frac{\pi}{L_x}\right)^2 + \left(\frac{\pi}{L_y}\right)^2\right)\sin\left(\frac{\pi}{L_x}x\right)\cdot\sin\left(\frac{\pi}{L_y}y\right)$$

第三步为迭代更新:

$$u_{ij}^{n+1} = 2\left(1 - r_x - r_y\right)u_{ij}^{n} - u_{ij}^{n-1} + r_x\left(u_{i+1,j}^{n} + u_{i-1,j}^{n}\right)$$

这里我们有一点要注意:我们更新 u_{ij}^{n+1} 时需要用到上下左右中 5 的顶点的值来计算差分,但边界上的 u 值可以直接根据边界条件算出,不需要通过差分模拟计算,这也就避免了边界点处无法计算差分的问题。

在输入条件为 $L_x = L_y = 2, T = 2, N = 30, K = 200$ 的情况下,我们得到的模拟解大致长这样:

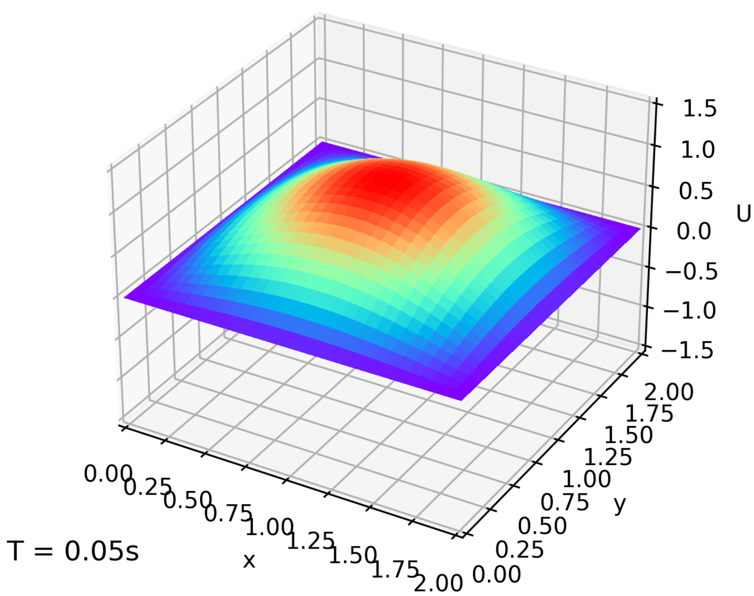

图 4: $t = 0.05$ 时刻的模拟解

误差函数随迭代次数的变化如图 5所示:

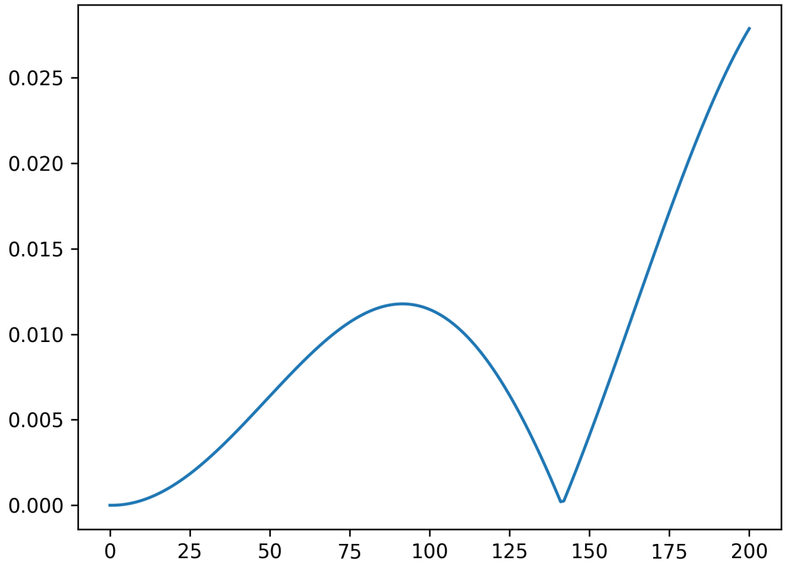

图 5: 误差函数

最后提醒一点，不同的参数设置可能会导致模拟不收敛，我们要保证步长比小于 1:

$$\frac{\tau}{\frac{1}{2}\sqrt{h_x^2+h_y^2}} \le 1$$

这可以通过观察 $r_x \le 1/4$ 来保证。