目录

ADI 方法数值求解二维平面传热问题

202011010101 物理 2001 孙陶庵

2023 年 4 月 10 日

1 问题

求以下 PDE 的数值解：$\frac{\partial T}{\partial t} = \kappa(\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2})$ 其中初始条件区域内的温度为 $0°C$ 板子长宽皆为 $25cm$ 边界条件上下左右四边分别有恒定温度：$100°C, 0°C, 75°C, 50°C$ matlab 用 ADI 方法求温度分布随时间的变化画出 3D 图形

2 ADI 方法

1. 将二维平面分割成多个小方格。在每个小方格中，u、x 和 y 分别离散化为 U(i,j)、X(i,j) 和 Y(i,j)，其中 i 和 j 分别表示 x 和 y 的离散坐标。

2. 使用 ADI 方法对每个小方格进行求解。即将二维热传导方程分解为两个一维问题。在每个方向上，可以使用隐式方法进行时间步进，并使用三对角矩阵算法来解决空间离散化问题。每次迭代后，将得到一个新的温度场 U(i,j)。根据所得到的温度场，计算需要的热量分布、温度梯度等。这里求解三对角矩阵是用的高斯消元法，简单来说就构造增广矩阵进行行变换后将线性方程组转化为一个上三角矩阵，进而通过回带求解得到方程组的解。以下是推导过程：

考虑二维热传导方程：

$$\frac{\partial u}{\partial t} = k\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right)$$

其中，u 是温度，t 是时间，k 是热传导系数，x 和 y 是空间坐标。

ADI 方法求解方程，需要将其分解为两个一维问题，一个在 x 方向，一个在 y 方向。具体地，我们可以将时间步长 Δt 分成两个部分，即 $\Delta t/2$，然后在 x 方向应用隐式 Euler 方法，得到一个中间温度场 $u_{i,j}^{n+1/2}$，然后在 y 方向应用隐式 Euler 方法，得到下一个时间步长的温度场 $u_{i,j}^{n+1}$。

1. 在 x 方向上，将 $u_{i,j}^{n}$ 的值用 $u_{i,j}^{n+1/2}$ 表示，即

$$u_{i,j}^{n} \approx u_{i,j}^{n+1/2}$$

2. 将 (2) 代入二维热传导方程中，得到

$$\frac{\partial u_{i,j}^{n+1/2}}{\partial t} = k\left(\frac{\partial^2 u_{i,j}^{n+1/2}}{\partial x^2} + \frac{\partial^2 u_{i,j}^{n}}{\partial y^2}\right)$$

3. 在 x 方向上，应用隐式 Euler 方法，得到

$$u_{i,j}^{n+1/2} - u_{i,j}^{n} = \frac{\Delta t}{2} k \frac{\partial^2 u_{i,j}^{n+1/2}}{\partial x^2}$$

4. 在 y 方向上，将 $u_{i,j}^{n+1/2}$ 的值用 $u_{i,j+1}^{n+1/2}$ 表示，即

$$u_{i,j}^{n+1/2} \approx u_{i,j+1}^{n+1/2}$$

5. 将 (4) 代入 (2) 中，得到

$$\frac{\partial u_{i,j+1}^{n+1/2}}{\partial t} = k\left(\frac{\partial^2 u_{i,j}^{n+1/2}}{\partial x^2} + \frac{\partial^2 u_{i,j+1}^{n+1/2}}{\partial y^2}\right)$$

6. 在 y 方向上，应用隐式 Euler 方法，得到

$$u_{i,j}^{n+1} - u_{i,j+1}^{n+1/2} = \frac{\Delta t}{2} k \frac{\partial^2 u_{i,j+1}^{n+1/2}}{\partial y^2}$$

7. 将 (3) 和 (6) 相加，得到

$$u_{i,j}^{n+1} - u_{i,j}^{n} = \frac{\Delta t}{2} k \left(\frac{\partial^2 u_{i,j}^{n+1/2}}{\partial x^2} + \frac{\partial^2 u_{i,j+1}^{n+1/2}}{\partial y^2}\right)$$

8. 将 $u_{i,j}^{n+1/2}$ 和 $u_{i,j+1}^{n+1/2}$ 的值用 $u_{i,j+1/2}^{n+1}$ 表示，即

$$u_{i,j}^{n+1/2} \approx u_{i,j+1/2}^{n+1}$$

$$u_{i,j+1}^{n+1/2} \approx u_{i,j+1/2}^{n+1}$$

9. 将 (8) 代入 (7) 中，得到

$$u_{i,j}^{n+1} - u_{i,j}^{n} = \frac{\Delta t}{2} k \left(\frac{\partial^2 u_{i,j+1/2}^{n+1}}{\partial x^2} + \frac{\partial^2 u_{i,j+1/2}^{n+1}}{\partial y^2} \right)$$

10. 最终得到 ADI 方法的迭代公式：

$$u_{i,j}^{n+1} = u_{i,j}^{n} + \frac{\Delta t}{2} k \left(\frac{u_{i+1,j}^{n+1} - 2u_{i,j}^{n+1} + u_{i-1,j}^{n+1}}{\Delta x^2} + \frac{u_{i,j+1}^{n+1} - 2u_{i,j}^{n+1} + u_{i,j-1}^{n+1}}{\Delta y^2} \right)$$

其中，$u_{i,j}^{n+1}$ 表示下一个时间步长的温度场，$u_{i,j}^{n}$ 表示当前时间步长的温度场，Δx 和 Δy 分别是 x 和 y 方向的网格间距。该公式是一个中心差分格式，可以用于数值计算二维平面传热问题。

1. 在 x 方向走半个时间步

$$\frac{u_{i,j}^{k+1/2} - u_{i,j}^{k}}{\Delta t/2} = \kappa \left(\frac{u_{i+1,j}^{k+1/2} - 2u_{i,j}^{k+1/2} + u_{i-1,j}^{k+1/2}}{(\Delta x)^2} + \frac{u_{i,j+1}^{k} - 2u_{i,j}^{k} + u_{i,j-1}^{k}}{(\Delta y)^2} \right)$$

2. 在 y 方向走半个时间步

$$\frac{u_{i,j}^{k+1} - u_{i,j}^{k+1/2}}{\Delta t/2} = \kappa \left(\frac{u_{i+1,j}^{k+1/2} - 2u_{i,j}^{k+1/2} + u_{i-1,j}^{k+1/2}}{(\Delta x)^2} + \frac{u_{i,j+1}^{k+1} - 2u_{i,j}^{k+1} + u_{i,j}^{k+1}}{(\Delta y)^2} \right)$$

取 $\Delta x = \Delta y, r = \dfrac{\kappa \Delta t}{(\Delta x)^2}$ 则方程变为：

$$-ru_{i-1}^{k+1/2} + 2(1+r)u_{i,j}^{k+1/2} - ru_{i+1}^{k+1/2} = ru_{i,j-1}^{k} + 2(1-r)u_{i,j}^{k} + ru_{j+1}^{k}$$

$$-ru_{i,j-1}^{k+1} + 2(1+r)u_{i,j}^{k+1} - ru_{i,j+1}^{k+1} = ru_{i-1}^{k+1/2} + 2(1-r)u_{i,j}^{k+1/2} + ru_{i+1}^{k+1/2}$$

3 代码部分

```
L = 25;
W = 25;
T = 1;
Nx = 21;
Ny = 21;
Nt = 1000;
dx = L / (Nx - 1);
dy = W / (Ny - 1);
dt = T / Nt;
kappa = 0.835;
u0 = zeros(Nx, Ny);
u0(ceil(Nx/2), ceil(Ny/2)) = 0;
u0(:, 1) = 75;
u0(:, end) = 50;
u0(1, :) = 100;
u0(end, :) = 0;

%zhushi
ax = dt * kappa / (2 * dx^2);
ay = dt * kappa / (2 * dy^2);
bx = 1 + 2 * ax;
by = 1 + 2 * ay;
cx = 1 - 2 * ax;
cy = 1 - 2 * ay;

for k = 1:Nt
    % alphazhushi
    for i = 2:Nx-1
        a = zeros(Ny-2, Ny-2);
```

```
30          b = zeros(Ny-2, 1);
31          % zhushialpha
32          for j = 2:Ny-1
33              a(j-1, j-1) = bx;
34              if j > 2
35                  a(j-1, j-2) = cx * ax;
36              end
37              if j < Ny-1
38                  a(j-1, j) = cx * ax;
39              end
40              b(j-1) = cy * u0(i-1, j) + (1 - 2*cy) * u0(i, j) + cy * u0(i+1,
    j);
41          end
42          % aaa1122
43          u1(i, 2:Ny-1) = GaussElimination(a, b);
44      end
45      % ylpha
46      for j = 2:Ny-1
47          a = zeros(Nx-2, Nx-2);
48          b = zeros(Nx-3, 1);
49          % zhushialpha
50          for i = 2:Nx-1
51              a(i-1, i-1) = by;
52              if i > 2
53                  a(i-1, i-2) = cy * ay;
54              end
55              if i < Nx-1
56                  a(i-1, i) = cy * ay;
57              end
```

```
58              b(i-1) = cx * u1(i, j-1) + (1 - 2*cx) * u1(i, j) + cx * u1(i, j
    +1);
59          end
60          % aaa1122
61          u0(2:Nx-1, j) = Thomas(a, b);
62      end
63  end
64
65  [X, Y] = meshgrid(0:dx:L, 0:dy:W);
66  surf(X, Y, u0');
67  xlabel('x');
68  ylabel('y');
69  zlabel('Temperature');
70
71
72
73  function x = GaussElimination(A, b)
74
75  n = size(A, 1);
76  m = size(b, 2);
77
78  if n ~= m
79      error('Error: The size of A and b do not match.');
80  end
81
82  % aa55688
83  for i = 2:n
84      factor = A(i, i-1) / A(i-1, i-1);
85      A(i, i-1) = 0;
```

```
    A(i, i) = A(i, i) - factor * A(i-1, i);
    b(i) = b(i) - factor * b(i-1);
end

% pa89
x(n) = b(n) / A(n, n);
for i = n-1:-1:1
    x(i) = (b(i) - A(i, i+1) * x(i+1)) / A(i, i);
end

x = x';
end
```

注释部分：

zhushi: 定义 AD 方程的系数矩阵; alphazhushi: 按 x 方向迭代; zhushialpha: 构建三对角矩阵; aaa1122: 解三对角矩阵方程; ylpha: 按 y 方向迭代; aa55688: 前向消元; pa89: 回代求解;

最后形成这张图片

[H]

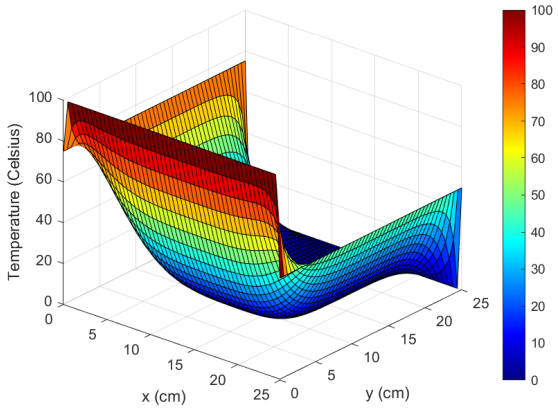

图 1: 结果

4 t=1s 时的温度分布

备注：这是最一开始的代码，原本以为只要求 $t = 1s$ 时的温度分布，后面才发现要全部的，所以又重新写了一份在上面

```
L = 0.25;
dx = 0.01; dy = 0.01;
dt = 0.01;
kappa = 0.835;
T_ini = zeros(L/dx, L/dy);

T_ini(1, :) = 100;  % up
T_ini(end, :) = 0;  % down
T_ini(:, 1) = 75;  % left
T_ini(:, end) = 50;  % right


t_start = 0;
t_end = 1;
t_range = t_start:dt:t_end;

T = T_ini;
for t = t_range
    % zhushialpha
    a = kappa * dt / dx^2;
    %ax = ay = r
    r = a;
    [m, n] = size(T);
    A = zeros(m*n, m*n);
    b = zeros(m*n, 1);
```

```
    for i = 1:m
        for j = 1:n
            idalpha = (i-1)*n + j;
            if i == 1 || i == m || j == 1 || j == n
                A(idalpha, idalpha) = 1;
                b(idalpha) = T(i, j);
            else
                A(idalpha, idalpha-n) = -r;
                A(idalpha, idalpha-1) = -1;
                A(idalpha, idalpha) = 2*(1+r);
                A(idalpha, idalpha+1) = -1;
                A(idalpha, idalpha+n) = -1/r;
                b(idalpha) = T(i-1, j)*r + T(i, j-1) + T(i, j+1) + T(i+1, j
   )/r;
            end
        end
    end

    %Temp. Distribution
    T_new = reshape(A\b, m, n);
    T = T_new;
end



imagesc([0 L], [0 L], T);
set(gca, 'YDir', 'reverse')
colormap(jet);
colorbar;
title('Temperature Distribution at t=1');
```

```
xlabel('x (m)');
ylabel('y (m)');
```

最后形成这张图片

[H]

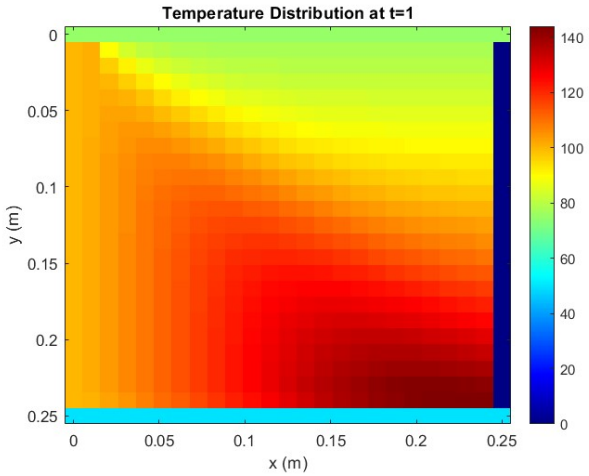

图 2: 结果

5 特别

在代码后面加上特定内容可以把要求的图做成多张图片，再透过 python 代码整合成 GIF（附件），可以得到温度随时间的变化。但是我这里代码无法继续加了（不会做），所以用 FTCS 方法实现

```
    import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
import imageio
import os

L = 25
n = 50
dx = L / (n - 1)
```

```
kappa = 0.835
sigma = 0.25
dt = sigma * dx**2 / kappa
T = np.zeros((n, n))
T[0, :] = 100
T[-1, :] = 0
T[:, 0] = 75
T[:, -1] = 50



total_frames = 15



frames = []
frame_count = 0
for j in range(int(6000)):
    for i in range(1, n-1):
        for k in range(1, n-1):
            T[i, k] += sigma * kappa * (T[i+1, k] - 2*T[i, k] + T[i-1, k] +
    T[i, k+1] - 2*T[i, k] + T[i, k-1])

    if j % 50 == 0 and frame_count < total_frames:
        frame_count += 1
        x = np.linspace(0, L, n)
        y = np.linspace(0, L, n)
        X, Y = np.meshgrid(x, y)

        fig = plt.figure()
        ax = plt.axes(projection='3d')
```

```
        ax.plot_surface(X, Y, T, cmap='viridis')
        ax.set_xlabel('x (cm)')
        ax.set_ylabel('y (cm)')
        ax.set_zlabel('Temperature (Celsius)')

        fig.canvas.draw()
        img = np.frombuffer(fig.canvas.tostring_rgb(), dtype=np.uint8)
        img = img.reshape(fig.canvas.get_width_height()[::-1] + (3,))
        frames.append(img)

        plt.close()

        if frame_count >= total_frames:
            break

imageio.mimsave('temperature.gif', frames, fps=3)
file_path = os.path.join("C:/Users/toby1/Desktop", "temperature.gif")
imageio.mimsave(file_path,frames, fps=10)
```