实验报告：一维热方程的数值解法

MindFlow 实验员

2026 年 1 月 19 日

目录

第 1 章 实验目的

TODO (待办)

待补充：增加实验的工程意义和应用场景描述

本实验的目标：

1. 理解抛物型 PDE 的差分格式原理
2. 实现显式欧拉格式 (FTCS) 与隐式格式 (Backward Euler)
3. 验证数值解的稳定性条件与收敛阶
4. 比较不同格式的计算效率

第 2 章 问题描述

求解如下一维热传导方程初边值问题：

$$\frac{\partial u}{\partial t} = \alpha \frac{\partial^2 u}{\partial x^2}, \quad 0 < x < L, \quad t > 0 \tag{0.2-1}$$

$$u(x,0) = \sin(\pi x/L) \quad \text{(初始条件)} \tag{0.2-2}$$

$$u(0,t) = u(L,t) = 0 \quad \text{(Dirichlet 边界条件)} \tag{0.2-3}$$

定义 0.2.1 (精确解)

利用分离变量法可得精确解：

$$u(x,t) = e^{-\alpha(\pi/L)^2 t}\sin(\pi x/L)$$

这是一个指数衰减的正弦波，衰减率由热扩散系数 α 决定。

FIXME (需修正)

第 2.1 节的推导过程需要补充完整的分离变量法步骤

第 3 章

数值方法

0.3.1 显式格式 (FTCS)

麦克斯韦方程组

$\nabla \cdot \vec{E} = \frac{\rho}{\varepsilon_0}, \quad \nabla \cdot \vec{B} = 0$

$\nabla \times \vec{E} = -\frac{\partial \vec{B}}{\partial t}, \quad \nabla \times \vec{B} = \mu_0 \vec{J} + \mu_0 \varepsilon_0 \frac{\partial \vec{E}}{\partial t}$

对时间采用向前差分，空间采用中心差分：

$$\frac{u_i^{n+1} - u_i^n}{\Delta t} = \alpha \frac{u_{i+1}^n - 2u_i^n + u_{i-1}^n}{\Delta x^2}$$

令网格比 $r = \frac{\alpha \Delta t}{\Delta x^2}$，迭代公式为：

$$u_i^{n+1} = r u_{i+1}^n + (1 - 2r) u_i^n + r u_{i-1}^n$$

【警告】稳定性条件

FTCS 格式是条件稳定的，Von Neumann 稳定性分析表明必须满足：

$$r = \frac{\alpha \Delta t}{\Delta x^2} \leq \frac{1}{2}$$

否则数值解将产生震荡并发散。这严格限制了时间步长的选取。

0.3.2　隐式格式 (Backward Euler)

> **NOTE (备注)**
> 隐式格式无条件稳定，但需要求解线性方程组，计算量更大

对时间采用向后差分：

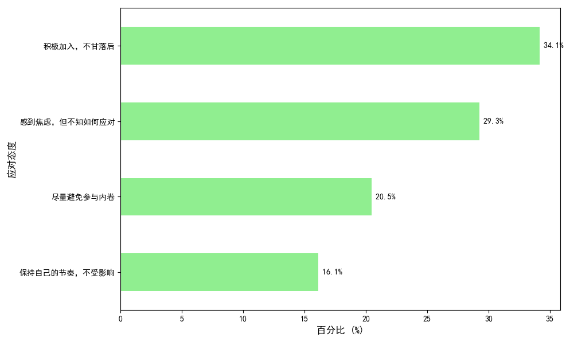
$$\frac{u_i^{n+1} - u_i^n}{\Delta t} = \alpha \frac{u_{i+1}^{n+1} - 2u_i^{n+1} + u_{i-1}^{n+1}}{\Delta x^2}$$

整理得三对角方程组 $\boldsymbol{A}\boldsymbol{u}^{n+1} = \boldsymbol{u}^n$，可用 Thomas 算法高效求解。

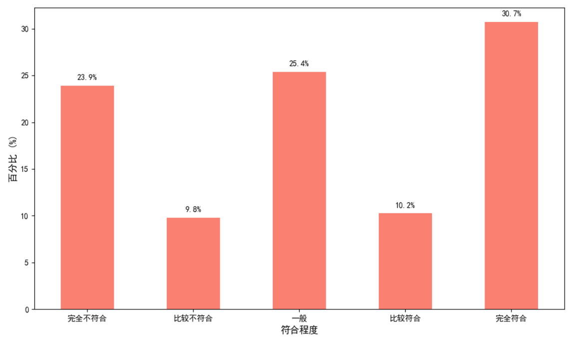
第 4 章　实验结果

0.4.1　稳定性验证

选取参数 $L = 1, \alpha = 0.01, \Delta x = 0.1$，测试不同网格比的效果。



(a) $r = 0.4$ (稳定)



(b) $r = 0.6$ (发散)

图 0.4-1　不同网格比下的数值解

0.4.2 误差分析

定理 0.4.1 (截断误差)

FTCS 格式的局部截断误差为 $O(\Delta t) + O(\Delta x^2)$，即时间一阶、空间二阶精度。

下表记录了不同时空步长下的最大误差 $E_\infty = \max_i |u_i^N - u_{exact}(x_i)|$：

表 0.4-1　误差收敛表

Δx	Δt	E_∞	收敛阶
0.100	0.0010	1.23e-3	-
0.050	0.00025	3.05e-4	2.01
0.025	0.00006	7.61e-5	2.00
0.0125	0.000015	1.90e-5	2.00

【技巧】误差分析技巧

收敛阶可用公式 $p = \log_2(E_h/E_{h/2})$ 计算。若结果接近 2，说明空间二阶精度得到验证。

49

第 5 章 结论与讨论

50

【结论】实验结论

1. **FTCS 格式**：实现简单、计算高效，但受 $r \leq 0.5$ 稳定性条件限制
2. **误差验证**：实验结果验证了 $O(\Delta x^2 + \Delta t)$ 的理论截断误差阶
3. **发散现象**：在网格比 $r > 0.5$ 时确实观察到数值发散，与理论分析一致
4. **隐式格式**：无条件稳定，适用于刚性问题，但计算成本更高

52

TODO (待办)

51 后续工作：实现 Crank-Nicolson 格式 (时间二阶精度) 并进行对比实验

53

54

第 6 章 附录：核心代码

55

代码 0.6-1：FTCS 格式实现

```
import numpy as np

def heat_ftcs(L, T, nx, nt, alpha):
```

56

```python
    """
    使用 FTCS 格式求解一维热方程

    Parameters:
        L: 空间域长度
        T: 时间域长度
        nx: 空间网格点数
        nt: 时间步数
        alpha: 热扩散系数

    Returns:
        u: 最终时刻的数值解
    """
    dx = L / (nx - 1)
    dt = T / (nt - 1)
    r = alpha * dt / dx**2

    # 稳定性检查
    if r > 0.5:
        print(f"Warning: r={r:.3f} > 0.5, 格式不稳定!")

    # 初始条件
    x = np.linspace(0, L, nx)
    u = np.sin(np.pi * x / L)

    # 时间推进
    for n in range(1, nt):
        u_new = u.copy()
        u_new[1:-1] = r*u[2:] + (1-2*r)*u[1:-1] + r*u[:-2]
        u = u_new

    return u
```

57

代码 0.6-2：误差计算与可视化

```python
import matplotlib.pyplot as plt

def compute_error(u_num, u_exact):
    """计算最大范数误差"""
    return np.max(np.abs(u_num - u_exact))

# 参数设置
L, T, alpha = 1.0, 0.1, 0.01
nx, nt = 101, 1001

# 数值解
u_num = heat_ftcs(L, T, nx, nt, alpha)

# 精确解
x = np.linspace(0, L, nx)
u_exact = np.exp(-alpha * (np.pi/L)**2 * T) * np.sin(np.pi * x / L)

# 误差
error = compute_error(u_num, u_exact)
print(f"Max error: {error:.6e}")
```

58