

# 偏微分方程数值解 + 第二次上机作业

2100012131 蒋鹏

2025 年 11 月 6 日

## 目录

<b>1 问题描述</b>	<b>1</b>
<b>2 算法设计</b>	<b>1</b>
2.1 网格离散	1
2.2 算子离散及边界条件处理	3
2.3 数值算法	3
<b>3 数值结果</b>	<b>3</b>
3.1 数值算例	3
3.2 数值实验及结果	3
3.3 简要分析	4
<b>4 后续改进</b>	<b>4</b>

## 1 问题描述

在指定区域上  $\Omega$  求解 Robin 边值问题的 Laplace 方程

$$\begin{cases} -\Delta u = f, & x \in \Omega \\ \alpha u + \beta \frac{\partial u}{\partial n} = g, & x \in \partial\Omega \end{cases} \quad (1)$$

其中区域  $\Omega$  如图1所示。当  $\alpha = 0$  得到 Neumann 边值问题，当  $\beta = 0$  得到 Dirichlet 边值问题，所以我们设计该方程的对应有限差分格式，同时满足三类问题的求解。

## 2 算法设计

### 2.1 网格离散

在  $x$  和  $y$  方向以相同尺度  $h$  进行均匀剖分。令  $n = 1/h$ , 自下而上, 从左至右, 对 Domain 内的点及边界点进行编号,  $0 \leq i \leq 4n, 0 \leq j \leq j\_num[i]$ 。所有点可分为三类: 内点, 边界点及临边界点, 角点。



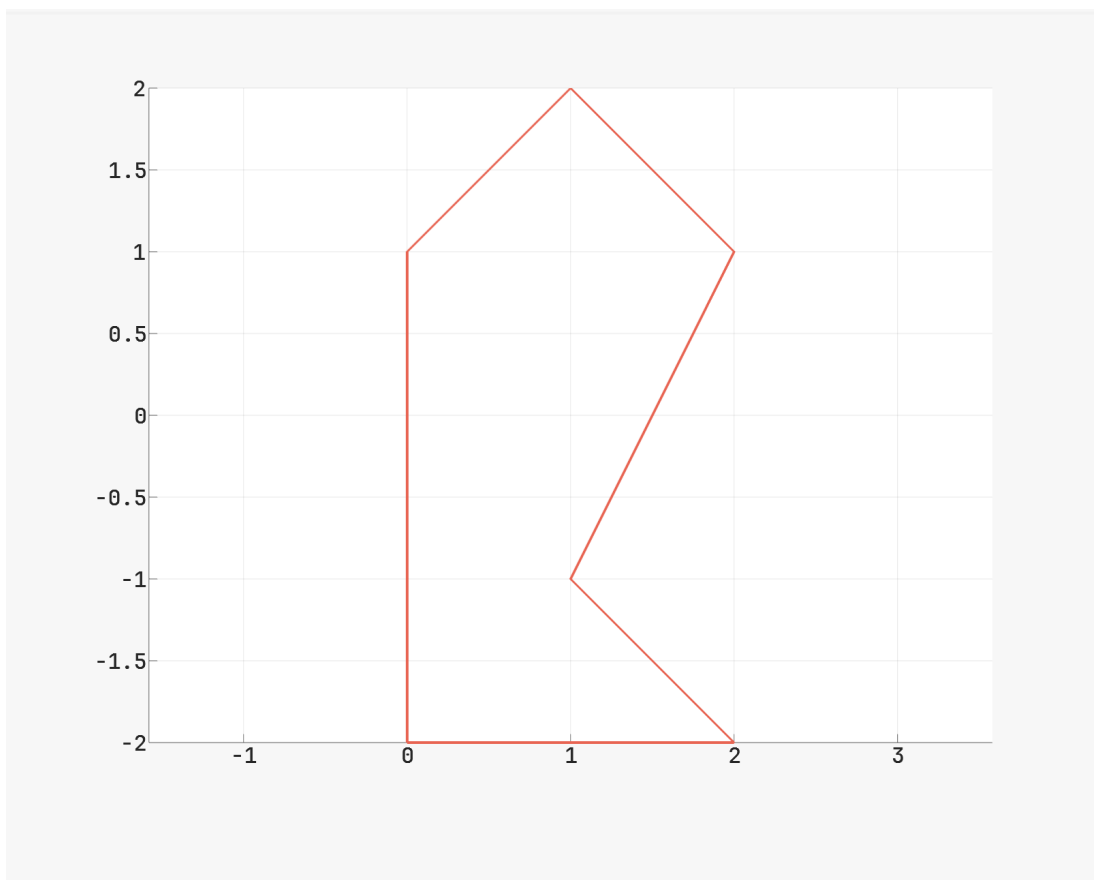


图 1: Domain



边界条件	参数取值	$h$	GMRES 迭代次数	$\ e_h\ _\infty$
Dirichlet	$\alpha=1, \beta=0$	0.125	16	3.1e-3
		0.0625	48	1.6e-3
		0.03125	172	8.0e-4
		0.015625	619	4.0e-4
		0.0078125	2367	2.0e-4
Neumann	$\alpha=0, \beta=1$	0.125	101	3.7e-2
		0.0625	502	1.9e-2
		0.03125	1169	9.7e-3
		0.015625	5983	5.0e-3
		0.0078125	31606	2.6e-3
Robin	$\alpha=1, \beta=1$	0.125	42	1.7e-2
		0.0625	104	8.1e-3
		0.03125	321	4.0e-3
		0.015625	1831	2.0e-3
		0.0078125	8553	1.0e-3

表 1: 数值结果展示

## 2.2 算子离散及边界条件处理

对于不同点进行不同格式离散:

对于内点, 采取五点差分格式离散 Laplace 算子。

对于边界点, 采取前向或者后向差分格式离散梯度, 进而离散方向导数。

对于临边界点, 即该点  $(x, y)$  非内点且不位于边界, 则在边界上最近点为  $(x^*, y^*)$ , 以点  $(x, y)$  的差分近似  $(x^*, y^*)$  的梯度, 进而离散方向导数。

对于角点, 赋予 Dirichlet 边界条件, 即规定角点处取值为真实解。

## 2.3 数值算法

此线性方程组较为复杂, 我们采取 GMRES(m) 迭代法进行求解。为了保证收敛性, 先进行 Gauss-Seidel 迭代约 100 步, 找到较为理想的初值, 随后进行 GMRES 求解。

# 3 数值结果

## 3.1 数值算例

采取给定的数值算例, 真实解  $u(x, y) = \frac{\sin(\pi x) \cos(2\pi y)}{5\pi^2}$ , 进而源项  $f(x, y) = \sin(\pi x) \cos(2\pi y)$ 。

## 3.2 数值实验及结果

对 Dirichlet、Neumann、Robin 三类边界条件均进行数值实验, 所得结果位于文档 output.txt, 列表如 I 所示。



### 3.3 简要分析

- 求解耗时、迭代次数: Dirichlet < Robin < Neumann. 原因是不同边值条件导致离散矩阵的条件数差异较大。
- 从表中可以清晰看出误差的下降趋势, 且该数值格式具有一阶精度, 即

$$\|e_h\|_\infty = \mathcal{O}(h) \quad (2)$$

- 修改单次迭代上限  $m$ , 可以减少 GMRES 迭代次数, 但单次迭代耗时增加, 总 CPU 时间应当基本不变。

## 4 后续改进

为了加速求解速率, 改进版将增加不完全 LU 分解 ILU 做预条件子, 实现 ILU-GMRES(m) 求解器。