

分类号： O241.5

学校代码 10335



博士 学位 论文



论文题目 两类非线性矩阵方程的数值算法研究

姓 吴乙荣

指导教师 黄正达 教授、王何宇 副教授

学科(业) 计算数学

所 系 数学系

提交时间 2014 年 4

Heyu Wang

Numerical Methods for Two Classes of Nonlinear Matrix Equations

by

Yirong Wu

Supervisor: Zhengda Huang

A Dissertation Submitted to Zhejiang University in Partial Fulfilment for the
Degree of Doctor of Philosophy in Computational Mathematics

Department of Mathematics, Zhejiang University
Hangzhou, Zhejiang, 310027
P. R. China

April, 2016

摘 要

关键词： Newton 法； Halley 法；

Abstract

Keywords: Newton's Method; Halley's Method;

目 录

表 格

插 图

第 1 章 绪论

1.1 研究背景

1.2 迭代法介绍

- 一般的 Newton 法 (初始点 x_0 给定):

$$F'(x_k)\Delta x_k = -F(x_k), \quad x_{k+1} = x_k + \Delta x_k, \quad k = 0, 1, \dots$$

- 简化 Newton 法: 这种变形的 Newton 法将每步计算 $F'(x_k)$ 改为固定的 $F'(x_0)$ (初始点 x_0 给定):

$$F'(x_0)\Delta x_k = -F(x_k), \quad x_{k+1} = x_k + \Delta x_k, \quad k = 0, 1, \dots$$

这样, 每步只需要计算 n 个分量函数, 但这种迭代法只有线性收敛.

- Newton 类法: 当所要处理的方程组维数较大时, 很难求解 Newton 方程 (??) 的精确解 Δx_k , 进而无法得到精确的 Jacobi 矩阵 $F'(x_{k+1})$. 为了能够应用 Newton 法较好的处理这种情况, 得到了如下的一种变形 Newton 法 (初始点 x_0 给定):

$$M(x_k)\Delta x_k = -F(x_k), \quad x_{k+1} = x_k + \Delta x_k, \quad k = 0, 1, \dots, \quad (1.1)$$

其中 $M(x_k)$ 是近似于 $F'(x_k)$ 的矩阵.

- 非精确 Newton 法: 同样考虑大规模方程组情形, 相比于 Newton 类法用一个近似的 Jacobi 矩阵来代替 $F'(x_k)$, 若考虑在求解 Newton 方程 (??) 时, 不去求其精确解而只需要求满足某种条件的近似解来作为迭代修正, 这样便得到如下的变形 Newton 法 (初始点 x_0 给定):

$$F'(x_k)\Delta x_k = -F(x_k) + r_k, \quad x_{k+1} = x_k + \Delta x_k, \quad k = 0, 1, \dots, \quad (1.2)$$

其中 $r_k \in \mathbb{Y}$ 一般应满足 $\|r_k\|/\|F(x_k)\| \leq \eta_k$, $k = 0, 1, \dots$, $\{\eta_k\}$ 满足 $0 < \eta_k < 1$, 可能与 x_k 有关, 为控制序列, 用来控制求方程 (??) 的解的精确程度. 显然令 $\eta_k \equiv 0$ 时得到一般的 Newton 法.

- 拟 Newton 法: Newton 法的主要缺点之一是每步都要计算导数 $F'(x)$ 的值, 当分量函数较复杂时计算很不方便, 拟 Newton 法是针对这一缺点提出的算法, 其核心是用通过计算函数值来代替导数以避免求导:

$$J_k \Delta x_k = -F(x_k), \quad J_{k+1} = J_k + \Delta J_k, \quad k = 0, 1, \dots,$$

其中 J_k 是近似于 $F'(x_k)$ 的矩阵. 不同的 ΔJ_k 选择, 可以得到不同的迭代法. 例如, 当取 $\Delta J_k = F(x_{k+1})\Delta x_k^T / (\Delta x_k^T \Delta x_k)$, 则得到 Broyden 法.

- Gauss-Newton 法: 这种变形的 Newton 法主要应用于求解非线性最小二乘(约束/无约束)问题, 迭代格式为:

$$\|F'(x_k)\Delta x_k + F(x_k)\| = \min, \quad x_{k+1} = x_k + \Delta x_k, \quad k = 0, 1, \dots \quad (1.3)$$

- 1) Kantorovich 型收敛理论: 理论上, Newton 法收敛性的一个最重要收敛结果是被称为 Newton-Kantorovich 半局部收敛定理 [?]. 该定理在理论和应用上都是相当重要的, 它是解方程算法现代研究的起点. 大量的收敛结果都是基于所谓的 Kantorovich 型条件而得到的, 例如, [? ? ? ? ? ? ? ? ? ?].
- 2) Smale 点估计理论: 该理论是由 Smale 于 1986 年提出, 由 α -理论和 γ -理论组成. 在 α -理论中, 假设 F 在初始点 x_0 是解析的, 给出了基于如下三个不变量的收敛判据 [?]:

$$\begin{cases} \alpha(F, x_0) = \beta(F, x_0)\gamma(F, x_0), \\ \beta(F, x_0) = \|F'(x_0)^{-1}F(x_0)\|, \\ \gamma(F, x_0) = \sup_{k \geq 2} \left\| \frac{1}{k!} F'(x_0)^{-1} F^{(k)}(x_0)^{-1} \right\|^{\frac{1}{k-1}}. \end{cases}$$

而 γ -理论则研究了算子 F 在解析条件下的局部收敛性. 定理 ?? 称为 γ -定理. 王兴华等人改进并完善了 Smale 点估计理论(见文献 [?] 及其所列文献). 值得指出的是, 王兴华引入了 γ 条件并在此基础上系统建立了 Smale 原先在解析条件下的全部结果(见文献 [?] 及其所列文献).

定义 1.1 (Q 收敛阶). 设序列 $\{x_k\}$ 收敛到 x^* . 如果存在 $q \geq 1$ 及常数 $c \geq 0$ 和 $N \geq 0$ 使得当 $k \geq N$ 时有 $\|x^* - x_{k+1}\| \leq c\|x^* - x_k\|^q$, 则称序列 $\{x_n\}$ 具有 Q 收

敛阶至少为 q . 特别地, 当 $q = 2$ 时称为(至少) Q 平方收敛, $q = 3$ 时称为(至少) Q 立方收敛.

定义 1.2 (R 收敛阶). 设序列 $\{x_k\}$ 收敛到 x^* . 如果存在 $\tau > 1$ 及常数 $c \in (0, \infty)$ 和 $\theta \in (0, 1)$ 使得对所有 $n \in \mathbb{N}$ 有 $\|x_k - x^*\| \leq c\theta^{\tau k}$, 则称 $\{x_k\}$ 具有 R 收敛阶至少为 τ .

1.3 矩阵函数

定义 1.3. 对于任意函数 f , 如果

$$f^{(j)}(\lambda_\ell), \quad j = 0, \dots, n_\ell - 1, \quad \ell = 1, \dots, r$$

的值都存在, 则称 f 在 A 的谱上有定义.

定义 1.4. 设函数 f 在矩阵 $A \in \mathbb{C}^{n \times n}$ 的谱上有定义, 且有 Jordan 典范形 (??), 则有

$$f(A) := Zf(J)Z^{-1} = Z\text{diag}(f(J_k))Z^{-1}, \quad (1.4)$$

其中

$$f(J_k) = \begin{bmatrix} f(\lambda_k) & f'(\lambda_k) & \cdots & \frac{f^{(m_k-1)}(\lambda_k)}{(m_k-1)!} \\ & f(\lambda_k) & \ddots & \vdots \\ & & \ddots & f'(\lambda_k) \\ & & & f(\lambda_k) \end{bmatrix}.$$

定理 1.1 ([?, 定理 1.13]). 设函数 f 在矩阵 $A \in \mathbb{C}^{n \times n}$ 的谱上有定义, 则有如下性质:

- (i) $f(A)$ 与 A 可交换, 即 $f(A)A = Af(A)$;
- (ii) $f(A^T) = f(A)^T$;
- (iii) $f(XAX^{-1}) = Xf(A)X^{-1}$;
- (iv) $f(A)$ 的全部特征值分别为 $f(\lambda_\ell)$, 其中 $\lambda_\ell, \ell = 1, \dots, n$ 为 A 的特征值;
- (v) 如果 X 与 A 可交换, 那么 X 与 $f(A)$ 可交换.

定理 1.2 ([? , 定理 1.15]). 设函数 f, g 在矩阵 $A \in \mathbb{C}^{n \times n}$ 的谱上有定义.

- (i) 若 $h(t) = f(t) + g(t)$, 则 $h(A) = f(A) + g(A)$;
- (ii) 若 $h(t) = f(t)g(t)$, 则 $h(A) = f(A)g(A)$.

1.3.1 矩阵平方根

定理 1.3 ([?]). 矩阵 $A \in \mathbb{C}^{n \times n}$ 存在一个平方根的充要条件是如下定义的递增的整数数列 d_1, d_2, \dots 没有两项都是相同的奇数:

$$d_\ell = \dim(\text{null}(A^\ell)) - \dim(\text{null}(A^{\ell-1})), \quad \ell = 1, 2, \dots$$

定理 1.4 (矩阵平方根的分类, [?]). 设非奇异矩阵 $A \in \mathbb{C}^{n \times n}$ 的 *Jordan* 典范形由 (??) 给出, 其所有不同的特征值数为 r . 如果 $r \leq s$, 那么 A 有 2^r 准平方根, 由如下式子给出:

$$X_j = Z \text{diag}(L_1^{(j_1)}, L_2^{(j_2)}, \dots, L_s^{j_s}) Z^{-1}, \quad j = 1, 2, \dots, 2^r,$$

其中 $j_k = 1$ 或 2 , 当 $\lambda_\ell = \lambda_k$ 时 $j_\ell = j_k$, $k = 1, 2, \dots, s$. 特别地, 如果 $r < s$, 那么 A 存在非准平方根, 其形式为

$$X_j(U) = Z U \text{diag}(L_1^{(j_1)}, L_2^{(j_2)}, \dots, L_s^{j_s}) U^{-1} Z^{-1}, \quad j = 2^r + 1, \dots, 2^s,$$

其中 $j_k = 1$ 或 2 , U 为任意的与 J 可交换的非奇异矩阵, 且对于每一个 j , 存在 ℓ 和 k , 使得当 $j_\ell \neq j_k$ 时 $\lambda_\ell = \lambda_k$.

定理 1.5 ([?]).

$$\begin{aligned} u_{ii}^2 &= t_{ii}, \quad i = 1, 2, \dots, n, \\ (u_{ii} + u_{jj})u_{ij} &= t_{ij} - \sum_{k=i+1}^{j-1} u_{ik}u_{kj}, \quad j > i. \end{aligned}$$

于是, 有如下计算非奇异矩阵平方根的算法, 该算法由 Björck & Hammarling 于 [?] 得到.

算法 1.1 计算矩阵平方根的 Schur 法 [?]

给定非奇异矩阵 $A \in \mathbb{C}^{n \times n}$, 本算法通过 Schur 分解来计算 A 的主平方根 $A^{1/2}$.

1. 计算矩阵 A 的 Schur 分解 $A = QRQ^*$;
2. 计算矩阵 U 各对角元素的主平方根 $u_{ii} = t_{ii}^{1/2}$, $i = 1, \dots, n$;
3. 依次计算矩阵 U 的非对角元:

$$u_{ij} = \frac{t_{ij} - \sum_{k=i+1}^{j-1} u_{ik}u_{kj}}{u_{ii} + u_{jj}}, \quad j = 2, 3, \dots, n, \quad i = j-1, j-2, \dots, 1;$$

4. 计算 $X = Q U Q^*$.

引理 1.1 ([?, 引理 6.8]). 假设 Newton 法 (??) 的初始点 X_0 与矩阵 A 可交换, 且所产生的序列 $\{X_k\}$ 是有定义的. 则对于所有的 $k \geq 1$, X_k 与 A 都是可交换的, 且此时 Newton 法的迭代格式为:

$$X_{k+1} = \frac{1}{2}(X_k + X_k^{-1}A). \quad (1.5)$$

$$\begin{cases} X_{k+1} = \frac{1}{2}(X_k + Y_k^{-1}), & X_0 = A, \\ Y_{k+1} = \frac{1}{2}(Y_k + X_k^{-1}), & Y_0 = I. \end{cases} \quad (1.6)$$

1.3.2 矩阵 p 次根

定理 1.6 (矩阵 p 次根的存在性, [?]). 定义如下的整数列 $\{d_k\}$:

$$d_k = \dim(\text{null}(A^k)) - \dim(\text{null}(A^{k-1})), \quad k = 1, 2, \dots.$$

矩阵 $A \in \mathbb{C}^{n \times n}$ 存在 p 次根的充要条件是对于任一整数 $\nu \geq 0$, 数列 $\{d_k\}$ 中至多只有一个元素处于 $p\nu$ 与 $p(\nu + 1)$ 之间.

定理 1.7 (矩阵 p 次根的分类 [?]).

定理 1.8 ([?, 定理 7.2]).

算法 1.2 计算矩阵平方根的实 Schur 法 [?]

给定矩阵 $A \in \mathbb{R}^{n \times n}$, 其所有特征值都不属于 $\mathbb{R}^- := (-\infty, 0]$. 本算法通过实 Schur 分解来计算 A 的主平方根 $A^{1/2}$.

1. 计算矩阵 A 的实 Schur 分解 $A = QRQ^T$, 其中 R 是 $m \times m$ 的块矩阵.
2. 计算矩阵 U 各对角块的主平方根: 当 $R_{ii} = [r_{ij}]_{1 \times 1}$ 时, $U_{ii} = R_{ii}^{1/2}$; 当 $R_{ii} = [r_{ij}]_{2 \times 2}$ 时,

$$U_{ii} = \begin{bmatrix} \alpha + \frac{1}{4\alpha}(r_{11} - r_{22}) & \frac{1}{2\alpha}r_{12} \\ \frac{1}{2\alpha}r_{21} & \alpha - \frac{1}{4\alpha}(r_{11} - r_{22}) \end{bmatrix},$$

其中

$$\alpha = \begin{cases} \left(\frac{|\theta| + (\theta^2 + \mu^2)^{1/2}}{2}\right)^{1/2}, & \theta \geq 0, \\ \frac{\mu}{2\left(\frac{|\theta| + (\theta^2 + \mu^2)^{1/2}}{2}\right)^{1/2}}, & \theta < 0, \end{cases}$$

$$\theta = \frac{r_{11} + r_{12}}{2}, \quad \mu = \frac{(-(r_{11} - r_{22})^2 - 4r_{21}r_{22})^{1/2}}{2}.$$

3. 依次通过计算如下的方程而得到矩阵 U 的非对角块 U_{ij} :

$$U_{ii}U_{ij} + U_{ij}U_{jj} = R_{ij} - \sum_{k=i+1}^{j-1} U_{ik}U_{kj}, \quad j = 2, 3, \dots, m, \quad i = j-1, j-2, \dots, 1.$$

4. 计算 $X = QUQ^T$.
-

定理 1.9 ([?]). 给定矩阵 $A \in \mathbb{C}^{n \times n}$, 设其谱为 $\sigma(A)$. 若 $\sigma(A) \subset \mathcal{E}_1$, 其中

$$\mathcal{E}_1 := \{z \in \mathbb{C} : \operatorname{Re} z > 0, |z| \leq 1\}, \quad (1.7)$$

则以 $X_0 = I$ 为初始点的 Newton 法 (??) 所产生的矩阵序列 $\{X_k\}$ 收敛于矩阵 A 的主 p 次根 $A^{1/p}$.

定理 1.10 ([?]). 给定矩阵 $A \in \mathbb{C}^{n \times n}$, 设其谱为 $\sigma(A)$ 。若 $\sigma(A) \subset \mathcal{E}_2$, 其中

$$\mathcal{E}_2 := \left\{ z \in \mathbb{C} : |z| \leq 2, |\arg(z)| < \frac{\pi}{4} \right\}, \quad (1.8)$$

则以 $X_0 = \mathbf{I}$ 为初始点的 Newton 法 (??) 所产生的矩阵序列 $\{X_k\}$ 收敛于矩阵 A 的主 p 次根 $A^{1/p}$.

$$\begin{cases} X_{k+1} = X_k \left(\frac{(p-1)\mathbf{I} + N_k}{p} \right), & X_0 = \mathbf{I}, \\ N_{k+1} = \left(\frac{(p-1)\mathbf{I} + N_k}{p} \right)^{-p} N_k, & N_0 = A. \end{cases} \quad (1.9)$$

算法 1.3 计算矩阵主 p 次根的 Schur-Newton 法 [?, 算法 3]

给定矩阵 $A \in \mathbb{R}^{n \times n}$, 其所有特征值都不属于 $\mathbb{R}^- := (-\infty, 0]$. 给定整数 $p \geq 2$, 则存在整数 $k_0 \geq 0$ 及奇数 q 使得 $p = 2^{k_0}q$. 本算法通过实 Schur 分解和对偶 Newton 法 (1.9) 来计算 A 的主 p 次根 $A^{1/p}$.

1. 计算矩阵 A 的实 Schur 分解 $A = QRQ^T$.
 2. 若 $q = 1$, 令 $k_1 = k_0$; 若 $q \neq 1$, 则选取 $k_1 \geq k_0$ 使得存在正数 s 使任意 A 的特征值 λ 满足
- $$s\lambda^{1/2^{k_1}} \in \left\{ z \in \mathbb{C} : \left| z - \frac{6}{5} \right| \leq \frac{3}{4} \right\}.$$
3. 通过算法 1.2 计算 $B = R^{1/2^{k_1}}$.
 4. 若 $q = 1$, 则令 $X = QBQ^T$; 若 $q \neq 1$, 则通过对偶 Newton 法 (1.9) 来计算 $C = (B/s)^{1/q}$ 并令 $X = Q(Cs^{1/q})^{2^{k_1-k_0}}Q^T$.
-

Guo & Higham [?] 给出了一种含参数的对偶 Newton 法:

$$\begin{cases} X_{k+1} = \left(\frac{(p+1)\mathbf{I} - N_k}{p} \right)^{-1} X_k, & X_0 = c\mathbf{I}, \\ N_{k+1} = \left(\frac{(p+1)\mathbf{I} - N_k}{p} \right)^p N_k, & N_0 = \frac{1}{c^p} A. \end{cases} \quad (1.10)$$

显然, 当 $N_k \rightarrow \mathbf{I}$ 时 $X_k \rightarrow A^{1/p}$ 。此外, 也给出了计算矩阵主 p 次根的算法 1.4:

算法 1.4 计算矩阵主 p 次根的含参数 Schur-Newton 法 [? , 算法 3.3]

给定矩阵 $A \in \mathbb{R}^{n \times n}$, 其所有特征值都不属于 $\mathbb{R}^- := (-\infty, 0]$. 给定整数 $p \geq 2$, 则存在整数 $k_0 \geq 0$ 及奇数 q 使得 $p = 2^{k_0}q$. 本算法通过实 Schur 分解和含参数 Newton 法 (1.10) 来计算 A 的主 p 次根 $A^{1/p}$.

1. 计算矩阵 A 的实 Schur 分解 $A = QRQ^T$.
2. 若 $q = 1$, 令 $k_1 = k_0$; 若 $q \neq 1$, 则选取 $k_1 \geq k_0$ 使得 $|\lambda_1/\lambda_n|^{1/2^{k_1}} \leq 2$, 其中 $\lambda_1, \dots, \lambda_n$ 为 A 的特征值且满足 $|\lambda_n| \leq \dots \leq |\lambda_1|$, 当 λ_ℓ 不全是实数时, 重新选取 k_1 使得对任意的 $\ell \in \{1, 2, \dots, n\}$ 都有
$$\arg(\lambda_\ell^{1/2^{k_1}}) \in \left(-\frac{\pi}{8}, \frac{\pi}{8}\right).$$
3. 通过算法 1.2 计算 $B = R^{1/2^{k_1}}$.
4. 若 $q = 1$, 则令 $X = QBQ^T$; 若 $q \neq 1$, 则先选取参数 c , 再通过含参数对偶 Newton 法 (1.10) 来计算 $C = B^{1/q}$ 并令 $X = QC^{2^{k_1-k_0}}Q^T$.

Halley 法是计算矩阵主 p 次根的另一种重要的迭代法。类似于 Newton 法, 若初始点 X_0 与矩阵 A 可交换, 则可得如下的计算矩阵主 p 次根的简化 Halley 法:

$$X_{k+1} = X_k ((p+1)X_k^p + (p-1)A)^{-1} ((p-1)X_k^p + (p+1)A), \quad AX_0 = X_0 A. \quad (1.11)$$

特别地, 初始点取为 $X_0 = I$ 是研究时主要考虑的情形, 如 [? ? ?]. 关于 Halley (1.11) 法的收敛性, Iannazzo [?] 得到了如下的结果:

定理 1.11 ([?]). 给定矩阵 $A \in \mathbb{C}^{n \times n}$, 设其谱为 $\sigma(A)$. 若 $\sigma(A) \subset \{z \in \mathbb{C} : \operatorname{Re} z > 0\}$, 则以 $X_0 = I$ 为初始点的 Halley 法 (1.11) 所产生的矩阵序列 $\{X_k\}$ 收敛于矩阵 A 的主 p 次根 $A^{1/p}$.

Guo 在 [?] 中进一步证明了当 $\sigma(A) \subset \{z \in \mathbb{C} : \operatorname{Re} z > 0\}$ 时 Halley 法 (1.11) 是三阶收敛的。应用在[?] 中对 Newton 法稳定性的分析方法可知, Halley 法 (1.11) 同样在数值计算中是不稳定的。故类似于 Newton 法, 可引入如下的

具有数值稳定的对偶形式的 Halley 法:

$$\begin{cases} X_{k+1} = X_k ((p+1)\mathbf{I} + (p-1)N_k)^{-1} ((p-1)\mathbf{I} + (p+1)N_k), & X_0 = \mathbf{I}, \\ N_{k+1} = N_k ((p+1)\mathbf{I} + (p-1)N_k)^{-1} ((p-1)\mathbf{I} + (p+1)N_k), & N_0 = A. \end{cases} \quad (1.12)$$

显然, 当 $N_k \rightarrow \mathbf{I}$ 时 $X_k \rightarrow A^{1/p}$ 。算法 1.5 给出了应用对偶 Halley 法 (1.12) 来计算 A 的主 p 次根 $A^{1/p}$ 。

算法 1.5 计算矩阵主 p 次根的 Schur-Halley 法 [? , 算法 4]

给定矩阵 $A \in \mathbb{R}^{n \times n}$, 其所有特征值都不属于 $\mathbb{R}^- := (-\infty, 0]$. 给定整数 $p \geq 2$, 则存在整数 $k_0 \geq 0$ 及奇数 q 使得 $p = 2^{k_0}q$. 本算法通过实 Schur 分解和对偶 Halley 法 (1.12) 来计算 A 的主 p 次根 $A^{1/p}$.

1. 计算矩阵 A 的实 Schur 分解 $A = QRQ^T$.
 2. 若 $q = 1$, 令 $k_1 = k_0$; 若 $q \neq 1$, 则选取 $k_1 \geq k_0$ 使得存在正数 s 使任意 A 的特征值 λ 满足
- $$s\lambda^{1/2^{k_1}} \in \left\{ z \in \mathbb{C} : \left| z - \frac{8}{5} \right| \leq 1 \right\}.$$
3. 通过算法 1.2 计算 $B = R^{1/2^{k_1}}$.
 4. 若 $q = 1$, 则令 $X = QBQ^T$; 若 $q \neq 1$, 则通过对偶 Halley 法 (1.12) 来计算 $C = (B/s)^{1/q}$ 并令 $X = Q(Cs^{1/q})^{2^{k_1-k_0}}Q^T$.
-

1.4 论文的组织

第 2 章 预备知识

我们把后文中用到的混合有限元方法和构建混合元所用的几何遗传树结构，以及移动网格方法，在本章中做简要描述和历史回顾。

2.1 混合有限元

混合有限元方法的专著可以参考F.Brezzi和M.Fortin [?]。二阶椭圆性问题的混合有限元方法就是基于Hellinger-Reissner变分原理的有限元方法。在流体中混合元方法是把速度和压力耦合在一起求解，因而精度得到提高，体现出相对于分开求解速度和压力有限元方法的优势。另外，对于不可压流体中的Galerkin逼近，只能采取混合有限元。在这一节中，我们以Stokes方程为例，来介绍混合元方法。Stokes方程可以看做是Navier-stokes的简化，去掉了非线性项。Stokes方程有两个需要求解的函数速度 \mathbf{u} 和压力 p 是耦合的，并且速度要满足质量守恒的条件($\nabla \cdot \mathbf{u} = 0$)。因此，需要我们用混合有限元方法来求解方程。为确保方程的解存在唯一，速度和压力所在的有限元空间必须满足inf-sup条件。如果有限元不满足inf-sup(LBB) 条件，例如在四边形单元中的 $Q_1 - Q_1, Q_1 - P_0$, 三角形单元中的 $P_1 - P_1, P_1 - P_0$ 元，还需要在压力空间上做稳定化。

下面我们以Stokes方程为例，给出它的混合变分形式。Stokes方程可以表示为：

$$\begin{aligned} -\Delta \mathbf{u} + \nabla p &= f \quad \text{在 } \Omega \text{ 内,} \\ \nabla \cdot \mathbf{u} &= 0 \quad \text{在 } \Omega \text{ 内,} \\ \mathbf{u} &= 0 \quad \text{在 } \partial\Omega \text{ 上,} \end{aligned} \tag{2.1}$$

令

$$a(\mathbf{u}, \mathbf{v}) = \int_{\Omega} \nabla \mathbf{u} \cdot \nabla \mathbf{v}, \tag{2.2}$$

$$b(\mathbf{u}, q) = - \int_{\Omega} \nabla \cdot \mathbf{u} q. \tag{2.3}$$

为简便起见，我们这里只考虑二维情形。对(3.1)利用格林公式，可以得到混合变分形式：寻找 $(\mathbf{u}, p) \in (H_0^1(\Omega))^2 \times L_0^2(\Omega)$ 使得

$$\begin{aligned} a(\mathbf{u}, \mathbf{v}) + b(\mathbf{v}, p) &= (f, v), \\ b(\mathbf{u}, q) &= 0. \end{aligned} \quad (2.4)$$

对 $\forall (\mathbf{v}, q) \in X \times P, X \in (H_0^1(\Omega))^2, P \in (L_0^2)$, $L_0^2 = \{q \in L^2(\Omega) | \int_{\omega} q dx = 0\}$ 。其中有限元空间 X, P 要满足inf-sup条件或者LBB条件，即

$$\beta \|q\|_P \leq \sup_{\mathbf{v} \in X} \frac{b(\mathbf{v}, q)}{\|\mathbf{v}\|_X}, \quad \forall q \in P. \quad (2.5)$$

这个条件的验证由很多专著中给出，例如([?], [?], [?])等。我们不加证明地给出如下定理：

定理 2.1. 问题(2.4)的解存在唯一。

在研究用混合元去离散Stokes问题时，我们构造的有限维子空间 $(X_h, P_h) \subset (X, P)$ 要满足离散的inf-sup条件：

$$\beta_0 \|q_h\|_{P_h} \leq \sup_{\mathbf{v}_h \in X_h} \frac{b(\mathbf{v}_h, q_h)}{\|\mathbf{v}_h\|_{X_h}}, \quad \forall q_h \in P_h. \quad (2.6)$$

其中 β_0 是一个跟网格尺度无关的常量。Mini元，RT元以及Taylor-Hood元都是满足离散的LBB条件的。我们得到(2.4)对应的离散形式：寻找 $(\mathbf{u}_h, p_h) \in X_h \times P_h$ 使得

$$\begin{aligned} a(\mathbf{u}_h, \mathbf{v}_h) + b(\mathbf{v}_h, p_h) &= (f_h, v_h), \\ b(\mathbf{u}_h, q_h) &= 0. \end{aligned} \quad (2.7)$$

求解(2.7)，得到的离散数值解 (\mathbf{u}_h, p_h) ，如果 (\mathbf{u}, p) 是弱形式(2.4)的解，则我们可以得到对应的误差估计：

$$\|\mathbf{u} - \mathbf{u}_h\|_1 + \|p - p_h\|_0 \leq C \left\{ \inf_{\mathbf{v}_h \in X_h} \|\mathbf{u} - \mathbf{v}_h\|_1 + \inf_{q_h \in P_h} \|p - q_h\|_0 \right\}. \quad (2.8)$$

我们倾向于使用满足LBB条件的混合元Taylor-Hood元，如 $P_2 - P_1, Q_2 - Q_1$ 元，即速度是二次元，压力是线性元。在实际的工程计算中，因为线性元简单，所以比较受欢迎。因此我们设想，如果把稳定的Taylor-Hood元中的速度二次元换成线性元，同时又要保证速度和压力满足inf-sup条件，这就

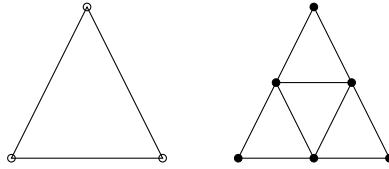


图 2.1: 左: 压力 p 单元, ○为 p 单元上的自由度; 右: 4个速度 v 单元, ●表示 v 单元上的自由度.

是 $P_1isoP_2P_1$ 的想法。如图2.1所示, 因为压力是线性元, 因此在一个三角形单元上有三个自由度, 分布在三角形的三个顶点上。同样地, 速度是二次元, 在这个三角形单元个有6个自由度, 分布在三角形的三个顶点和三边的中点上。如果这个三角形单元加密一次, 即连接三边中点, 因此得到四个小三角形。这样在每个小三角形上有分布着三个自由度。在[?]中证明了 $P_1isoP_2P_1$ 是满足inf-sup条件的。令 $X_h = X_{h/2}^1, P_h = P_h^1$, 则

定理 2.2. 如果 Ω 是一个多边形, 并且对所有 h 都有 $\Omega_h = \Omega$, 如果 Ω_h 中所有的三角形都至少有一个顶点不在 Ω 的边界 Γ 上, X_h, P_h 是满足 LBB 条件的, 那么问题(2.7)是有唯一解 $(\mathbf{u}_h, p_h) \in X_h \times (P_h/R)$ (p_h 除了一个常数外是确定的)。

定理的过程我们不再给出, [?]中还给出了 $P_1isoP_2P_1$ 元的误差估计:

$$|\nabla(\mathbf{u} - \mathbf{u}_h)|_0 \leq hK(||\mathbf{u}||_{(H^2(\Omega))^2} + ||p||_{H^1(\Omega)/R}), \quad (2.9)$$

$$|\nabla(p - p_h)|_0 \leq K(||\mathbf{u}||_{(H^2(\Omega))^2} + ||p||_{H^1(\Omega)/R}). \quad (2.10)$$

但是需要注意到图2.1中, 每个小三角形上的三个自由度, 其实是大的三角形6个自由度中的3个。因此, 在做速度单元和压力单元拼装的时候, 需要先找到速度单元基函数所在的大单元, 然后用大单元的基函数中正好位于当前速度单元上基函数与压力单元上的基函数进行拼装。这会带来一些技术上的困难。如果图2.1右中每个小三角形上的自由度都是局部在一个单元上, 这时跟 $P_1isoP_2P_1$ 拥有同样的网格结构, 也满足inf-sup条件, 但速度单元全部是线性元, 因此称作 $4P1 - P1$ 元。在这种情况下, 只要建立速度单元和压力单元之间的索引, 速度单元与压力单元间的拼装变得容易很多。这时速度和压力分别在两套网格上, 其中速度网格由压力网格加密一次得到。

2.2 几何遗传树结构

在上一节中，我们提到速度单元和压力单元之间的索引，而这个索引的建立利用了[?] 中的几何遗传树结构。这种树结构是用来做h-自适应，下面我们就介绍一下这种树结构。

在h-自适应中，两个耦合在一起的物理量(如上一节提到的速度 u 和压力 p)，有可能在不同的地方出现奇异，因此需要分别在两套网格上离散数值解。在拼装 $\int_{\Omega} \nabla \cdot \mathbf{u} p$ 这一项时，因为 \mathbf{u} 和 p 在不同的网格上，因此需要构建两套网格之间的联系。如果函数在相对于计算区域来说，很小的区域内有奇异，好的网格需要在这个奇异出现的地方网格非常集中。特别地，如果最小的网格单元和最大的网格单元面积相差上百倍，这时候建立两个网格之间对应的代价是非常昂贵的。为了创建高效的多网格自适应策略，[?] 中选取了几何遗传树，具有遗传性的网格结构可以描述为点、线、面、体。并且针对无结构网格，加密也是通过按层次来加密的。这种分层的数据结构可以提供简单的编程方法和快速的网格加密算法，以及多重网格求解器。

我们以二维区域 Ω 上的一个三角剖分 \mathcal{T}_0 为例，来介绍一下几何遗传树结构。为与有限元空间的单元区分开，称三角剖分中的单元为几何单元。 τ 为 \mathcal{T}_0 的几何单元。三角剖分中所有几何性质都有属于的关系，即如果一条边是一个三角形三边中的一条，则称这条边属于这个三角形。同样的，如果一个点是一条边两个顶点中的一个，则称这个点属于这条边。三角剖分中的一个三角形可以被加密成四个小的三角形。在三角形加密的过程中，三条边分别被分成了两部分。当三角剖分中所有的三角形都加密一次，我们可以得到一个更密的三角剖分 \mathcal{T}_1 ，这就是 \mathcal{T}_0 的一次全局加密。经过逐次加密，我们可以得到一系列的三角剖分 $\{\mathcal{T}_n\}$ 。如果 \mathcal{T}_n 中的一个三角形 τ_1 位于 \mathcal{T}_{n-1} 中的三角形 τ_0 中，则 τ_1 被称作是 τ_0 的一个孩子。我们给 \mathcal{T}_0 中的所有三角形一个虚拟的父亲，与所有从 \mathcal{T}_0 的三角形中派生出的所有四叉树组成了一个树的数据结构，如图2.4所示。这棵树也被称为几何遗传树。用几何遗传树结构来构建h-自适应的内容就不再详细介绍。

2.3 移动网格方法

移动网格方法常常用在数值计算中，寻找好的网格(跟具体问题有关)，并在网格上进行数值求解，使得在不增加自由度的前提下，尽可能的提高计算精度。移动网格方法包括网格的移动策略，以及在方程在网格上的数值求解算

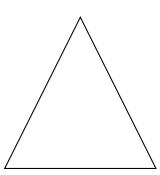


图 2.2: 全局加密一次的网格

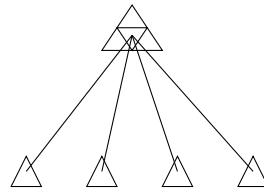
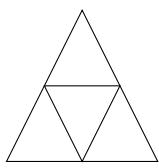


图 2.3: 它的几何遗传树

图 2.4: 几何遗传树结构

法。在[?]、[?]、[?] 中系统地给出了移动网格方法的介绍。移动网格的策略是基于等分布原则(Equi-distribute Principle)，这是de Boor在1974年[?] 提出的。等分布原则是根据一个能反映局部网格计算质量的指标，比如误差，移动网格节点使得这个指标在网格上均匀分布。如果在计算结果误差比较大的地方网格集中，尽量使得误差分布的比较均匀，从而提高计算精度。

一些符号的定义如下：令 $\Omega \subset \mathbb{R}^N$, $N = 1, 2, 3$ 代表物理区域， x 代表物理区域上的坐标。同样地， $\Omega_c \subset \mathbb{R}^N$ 表示逻辑区域(或称计算区域)， ξ 为其坐标。

de Boor 给出了一维情形的证明，如下：

假设 $\Omega = [a, b]$, $\Omega_c = [0, 1]$ 分别为物理区域和计算区域，在 Ω_c 上有一个均匀网格 $\{\xi_j = \frac{j}{N} | j = 0, 1, \dots, N\}$,

2.4 预处理方法

2.5 预备引理

$$\begin{aligned}
 |r(E(z), \lambda)| &\leq \sup_{m \geq 2} \left\{ \left| \sum_{j=2}^m c_{1,j} u^{j-2} \right| \right\} \cdot \left| 1 - \frac{r(z, \lambda)}{u} \right| \cdot \sum_{m=2}^{\infty} \left| \frac{r(z, \lambda)}{u} \right|^{m-2} \cdot |r(z, \lambda)|^2 \\
 &= \sup_{m \geq 2} \left\{ \left| \frac{S_{1,m}(u)}{u^2} \right| \right\} \cdot \frac{\left| 1 - \frac{r(z, \lambda)}{u} \right|}{1 - \left| \frac{r(z, \lambda)}{u} \right|} \cdot |r(z, \lambda)|^2 \\
 &= \sup_{m \geq 2} \left\{ \left| \frac{S_{1,m}(u)}{u^2} \right| \right\} \cdot \frac{|u| + |r(z, \lambda)|}{|u| - |r(z, \lambda)|} \cdot |r(z, \lambda)|^2.
 \end{aligned}$$

$$|r(z_{k+1}, \lambda)| = |\phi(r(z_k, \lambda))|$$

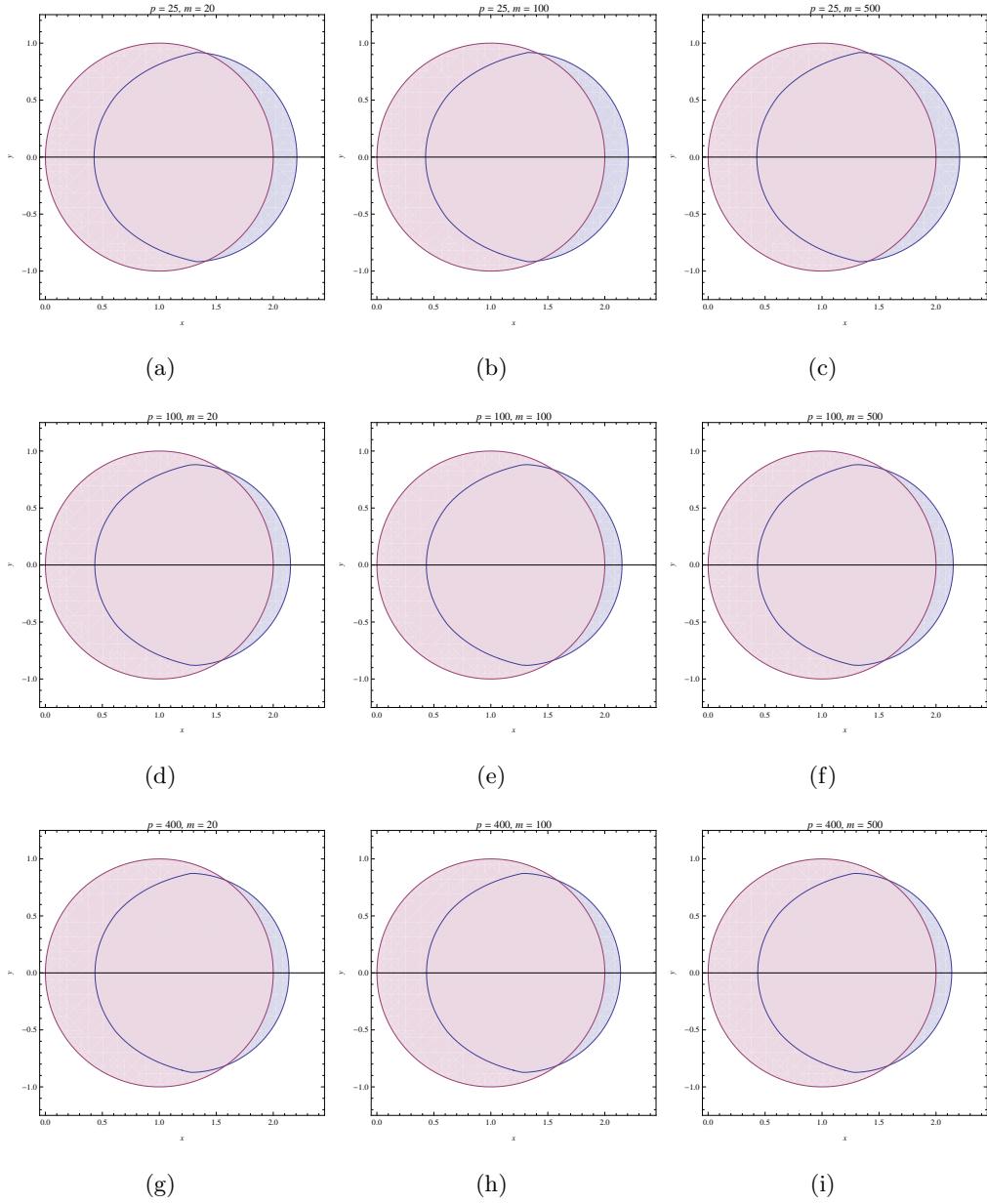


图 2.5: 由 (??) 给出的 \mathcal{R}_1 在九种不同情形 (p 分别取 25, 100, 400 及 m 分别取 20, 100, 500) 下的近似域, 其中红色域表示集合 $\{z \in \mathbb{C} : 1 - z \in \overline{\mathcal{D}}_1\}$, 而蓝色域表示集合 $\{z \in \mathbb{C} : 1 - z \in \mathcal{D}_{0,1} \cap \mathcal{D}_{2,1}\}$.

$$\leq \sup_{m \geq 2} \left\{ \frac{|S_{1,m}(r(z_0, \lambda))|}{|r(z_0, \lambda)|^2} \right\} \cdot \frac{|r(z_0, \lambda)| + |\phi_1(r(z_0, \lambda))|}{||r(z_0, \lambda)| - |\phi(r(z_0, \lambda))||} \cdot |r(z_k, \lambda)|^2$$

$$\begin{aligned}
 &\leq \sup_{m \geq 2} \left\{ \frac{|S_{1,m}(r(z_0, \lambda))|}{|r(z_0, \lambda)|^2} \right\} \cdot \frac{|r(z_0, \lambda)| + |\phi_1(r(z_0, \lambda))|}{||r(z_0, \lambda)| - |\phi_1(r(z_0, \lambda))||} \left[q_2^{2^k-1}(z_0) \right]^2 \cdot |r(z_0, \lambda)|^2 \\
 &= \sup_{m \geq 2} \left\{ \frac{|S_{1,m}(r(z_0, \lambda))|}{|r(z_0, \lambda)|} \right\} \cdot \frac{|r(z_0, \lambda)| + |\phi_1(r(z_0, \lambda))|}{||r(z_0, \lambda)| - |\phi_1(r(z_0, \lambda))||} \cdot [q_2(z_0)]^{2^{k+1}-2} \cdot |r(z_0, \lambda)| \\
 &= [q_2(z_0)]^{2^{k+1}-1} \cdot |r(z_0)|,
 \end{aligned}$$

由此, 根据归纳法知 (??) 得证. 证完.

算法 2.1 Preprocessing iterative framework for computing $A^{1/p}$

Given $A \in \mathbb{C}^{n \times n}$ with no nonpositive real eigenvalues, an integer $p = 2^{k_0}q$ with $k_0 \geq 0$ and q odd. This algorithm computes the principal p th root of matrix A via a Schur decomposition and some iterative method.

1. Compute the Schur decomposition of $A = QRQ^*$;
 2. If $q = 1$, then $k_1 = k_0$; else choose the smallest $k_1 \geq k_0$ such that for each eigenvalue λ of matrix A , $\lambda^{1/2^{k_1}}$ belongs to some region \mathcal{R} ;
 3. Compute $B = R^{1/2^{k_1}}$ by taking the square root k_1 times; if $q = 1$, then set $X = QBQ^T$; else continue;
 4. Compute $C = B^{1/q}$ by using some iterative method and set $X = QC^{2^{k_1-k_0}}Q^T$.
-

注 2.1.

$$\mathcal{D}_3 := \{z \in \mathbb{C} : 1 - z \in \overline{\mathcal{D}}_1\}, \quad (2.11)$$

$$\mathcal{D}_4 := \begin{cases} \left\{ z \in \mathbb{C} : \left| \frac{8}{5} - z \right| < \frac{6}{5} \right\}, & p = 2, 3, 4, \\ \left\{ z \in \mathbb{C} : |\arg(z)| < \frac{\pi}{6}, \left| \frac{5p-13}{4(p-3)} - z \right| < \frac{43p-105}{48(p-3)} \right\}, & p \geq 5. \end{cases}$$

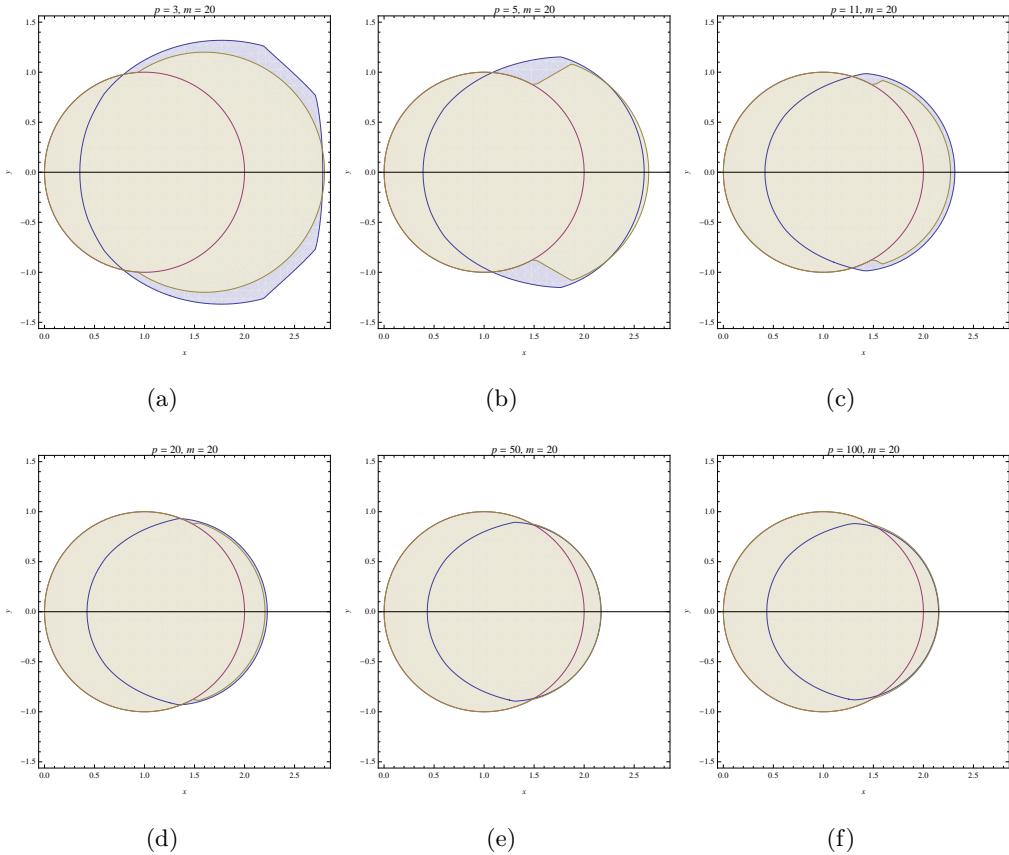


图 2.6: For fixed $m = 20$ and $p = 2, 5, 11, 20, 50, 100$, the actual convergence regions \mathcal{R}_1 defined in (??) (the union of the red and blue parts) and the approximate convergence regions \mathcal{R}_1^N defined in (??) (the yellow parts).

第3章 基于4P1-P1元求解Navier-Stokes方程的移动网格方法

根据上一章，我们知道混合元是求解不可压Navier-Stokes方程的一种重要方法。为了保证解的存在唯一性，需要使速度和压力两个解所在的空间满足LBB条件。其中一种办法是使得速度空间相对压力空间来说，自由度足够多，例如mini元，Taylor-Hood元等等。另外一种方法是对压力空间施加一些约束，比如说稳定化的 $P_1 - P_1$ 元， $P_1 - P_0$ 元。为了减少计算量，提高计算效率，通常会运用自适应网格方法。在文献[?], [?], [?]中，运用了h-自适应的P2P1元。在实际的工程计算中，我们通常倾向于使用线性元，而非高次元。[?]提出自适应网格和稳定化P1P1元、P1P0元相结合的策略。[?]将移动网格方法应用到不可压Navier-Stokes 方程的求解。我们移动网格部分的策略是基于文献[?]中的工作。然而，不满足LBB条件的有限元对应用到自适应网格方法上有一定的技术困难。综合以上考虑，我们选取稳定的 $P_1isoP_2P_1$ 元，它自然满足LBB条件，详细见[?]. 在P1ISOP2P1元，速度单元所在的网格可以有压单元的网格加密一次得到，如Figure (2.1)注意到在速度单元 \mathbf{u} 和压力单元 p 上均是线性元。但是，速度单元网格上的6个基函数不在同一个速度单元上。因此在拼装散度块矩阵，即拼装压力单元和速度单元，这个过程并不显然。

在本文的工作中，我们选取 $4P1 - P1$ 元，它与 $P_1isoP_2P_1$ 拥有相同的网格结构，自然也满足inf-sup条件，但需要指出的是 $P_1isoP_2P_1$ 元的速度单元上的基函数都局部的位于相同的速度单元内。这给我们拼装散度块矩阵提供了便利：只要我们建立四个速度单元和大的压力单元之间的索引。索引的建立又依赖于两层网格的数据结构，这种两层网格用我们上一章提到的几何遗传树结构，可以建立两层网格间的对应关系。

基于这种网格遗传树的结构，我们可以很容易的建立四个小的速度单元与压力宏单元间的索引。同时在[?]中h-自适应方法也是基于这种树结构。自然的，我们可以将h-自适应方法应用到4P1-P1元上。自适应网格可以减少计算量，同时可以研究流体局部小尺度上的现象，比如涡。同时，移动网格网格方法也可以应用在4P1-P1元上，网格移动我们只移动压力网格，当压力网格移动

完，速度网格由压力网格加密一次即可得到。但是需要注意的是，求解不可压的Navier-Stokes方程，在将旧网格的解插值到移动后的网格的过程，要满足散度为0的条件，这个工作由[?]中给出。

在第一节中，我们介绍Navier-Stokes方程的知识。在第二节中我们展示4P1-P1的数据结构，以及单元间索引的建立过程，接下来我们用4P1-P1元近似Navier-Stokes方程。移动网格的策略将在第四节中给出。最后，我们给出数值例子。

3.1 数据结构

4P1-P1是基于两套不同的网格和两种有限元空间。速度网格可以由压力网格全局加密一次得到。网格的数据结构是基于[?]中的几何遗传树结构，如Figure(2.4)所示。一个宏压力单元对应这四个速度单元，通过遍历一次所有的速度单元，利用几何遗传树结构，就可以建立速度单元和压力单元的1 – 1对应。

在AFEPack中，速度网格和压力网格分别存储在两张非正则网格irregularMeshV和irregularMeshP上，这两张非正则网格都是建立在同一棵树上。非正则网格上，可以进行全局加密，局部加密以及疏化的操作，能进行这种操作得益于它的四叉树结构。所有的网格节点从根节点到叶子节点，都存储在非正则网格中。这里irregularMeshV是由irregularMeshP进行全局一次加密得到的，根据非正则网格的树结构，我们可以通过遍历irregularMeshV的全部活动单元(即叶子节点)，通过活动单元来找到它的父亲单元(即叶子节点的父亲节点)。这样我们建立了从速度网格单元到压力网格单元之间的单向索引。还是根据活动单元的父亲节点也是具有树结构的，因此，我们可以根据父子节点单元(压力单元)来找到它对应的四个儿子单元(即四个速度单元)，这样我们也建立了从压力单元到速度单元的单向索引。到这里，速度和压力间的单元已经建立了1 – 1索引，注意到，只要不产生新的网格单元，我们建立的索引不需要重新改。应用到移动网格上，我们只需要一次构建索引就可以，不管网格如何移动，网格间的索引不会改变。建立索引的过程参见算法(3.1)。

Algorithm 3.1 构建速度单元和压力单元间的索引

```

achieveiterator ← irregualerMeshV.beginActiveElement()
enditerator ← irregualerMeshV.endActiveElement()
while achieveiterator ≠ enditerator do
    int index-v-element ← achieveiterator→ index
    HElement < DIM, DIM>* parent ← activeiterator→ parent
    int index-p-element ← parent→ index
    int n-child ← parent→ n-child
    index-p2v[index-p-element].resize(n-child)
    while i ≥ 0 and i < n-child do
        HElememt<DIM,DIM> *chi ← parent→ child[i]
        int index-v-element ← child→ index
        index-p2v[index-p-element][i] ← index-v-element
        index-v2p[index-v-element] ← index-p-element
    end while
end while

```

3.2 混合元近似

3.2.1 流体方程

在Stokes方程

$$\begin{aligned}
 -\Delta \mathbf{u} + \nabla p &= \mathbf{f} && \text{在 } \Omega \text{ 内,} \\
 \nabla \cdot \mathbf{u} &= 0 && \text{在 } \Omega \text{ 内,} \\
 \mathbf{u} &= 0 && \text{在 } \partial\Omega \text{ 上,}
 \end{aligned} \tag{3.1}$$

的基础上加一个对流项，再加上时间项，我们可以得到一个时间发展的Navier-Stokes方程

$$\begin{aligned}
 \frac{\partial \mathbf{u}}{\partial t} - \nu \nabla^2 \mathbf{u} + \mathbf{u} \cdot \nabla \mathbf{u} + \nabla p &= \mathbf{f}, \\
 \nabla \cdot \mathbf{u} &= 0.
 \end{aligned} \tag{3.2}$$

其中 $\nu > 0$ 是一个常数，称作动力学粘性系数。跟Stokes 方程中类似， \mathbf{u} 表示流体速度， p 表示压力。对流项 $\mathbf{u} \cdot \nabla \mathbf{u} := (\mathbf{u} \cdot \nabla) \mathbf{u}$ ，是非线性项，这也使得Navier-Stokes方程的解不唯一，这也给我们的数值计算带来了一定的挑战。

系统(3.2)的计算区域是 Ω , 可以是二维的或三维的。在边界 $\partial\Omega = \partial\Omega_D \cup \Omega_N$ 上的边界条件如下:

$$\begin{aligned}\mathbf{u} &= \mathbf{w}, && \text{on } \partial\Omega_D, \\ \nu \frac{\partial \mathbf{u}}{\partial n} - p &= \mathbf{0}, && \text{on } \partial\Omega_N.\end{aligned}\quad (3.3)$$

其中边界 $\partial\Omega = \partial\Omega_D \cup \partial\Omega_N$, \mathbf{n} 表示边界的外法向。Dirichlet边界根据边界上的速度和外法向的乘积, 可以细分为:

$$\begin{aligned}\partial\Omega_+ &= x \text{在} \partial\Omega \text{上} | \boldsymbol{\omega} \cdot \mathbf{n} > 0, && \text{出流边界} \\ \partial\Omega_0 &= x \text{在} \partial\Omega \text{上} | \boldsymbol{\omega} \cdot \mathbf{n} = 0, && \text{特征边界} \\ \partial\Omega_- &= x \text{在} \partial\Omega \text{上} | \boldsymbol{\omega} \cdot \mathbf{n} < 0, && \text{入流边界}\end{aligned}\quad (3.4)$$

如果边界全部是Dirichlet边界条件, 即 $\partial\Omega = \partial\Omega_D$, 那么Navier-Stokes问题(3.2)和(4.2)的压力解除去一个常数外是唯一的。我们对(3.2)中的不可压约束应用散度定理,

$$0 = \int_{\Omega} \nabla \cdot \mathbf{u} = \int_{\partial\Omega} \mathbf{u} \cdot \mathbf{n} = \int_{\Omega} \boldsymbol{\omega} \cdot \mathbf{n}. \quad (3.5)$$

即边界值要满足相容性条件

$$\int_{\partial\Omega_+} \boldsymbol{\omega} \cdot \mathbf{n} + \int_{\partial\Omega_-} \boldsymbol{\omega} \cdot \mathbf{n} = 0. \quad (3.6)$$

简单的讲, 就是说流入 Ω 的流体的体积, 要与流出的体积相等。这也是压力不唯一的原因。在处理入流/出流的问题时, 要注意保证这个相容性条件, 否则Navier-Stokes问题的解有可能不存在。一般情况下, 我们在出流边界设置自然条件, 相容性条件会自然满足, 因此这时候问题(3.2)和(4.2)的压力解是唯一的。

(3.2)是Navier-Stokes的原始变量形式, 它的另外一组表现形式是流函数形式。如果区域 Ω 是单连通的, 那么不可压缩条件

$$\frac{\partial u_x}{\partial x} = -\frac{\partial u_y}{\partial y} \quad (3.7)$$

意味着存在一个流函数 $\psi(x, y)$ 使得

$$u_x = \frac{\partial \psi}{\partial y}, \quad u_y = -\frac{\partial \psi}{\partial x}. \quad (3.8)$$

$\psi(x, y)$ 除去一个常数外被 \mathbf{u} 唯一确定。对(3.2)中的动量方程取curl，我们可以得到流函数形式：

$$-\nu \nabla^4 \psi + \frac{\partial \psi}{\partial y} \frac{\partial}{\partial x} (\nabla^2 \psi) - \frac{\partial \psi}{\partial x} \frac{\partial}{\partial y} (\nabla^2 \psi) = -\operatorname{curl} \mathbf{f} \quad (3.9)$$

但由于流函数形式(3.9)在三维空间没有一般形式，所以流函数形式只在二维。我们还是研究(3.2)中的原始变量形式。

3.2.2 Weak formulation

我们定义解和检验空间如下：

$$\mathbf{H}_E^1 := \{\mathbf{u} \in \mathcal{H}^1(\Omega)^d \mid \mathbf{u} = \mathbf{w} \text{ on } \partial\Omega_D\}, \quad (3.10)$$

$$\mathbf{H}_{E_0}^1 := \{\mathbf{v} \in \mathcal{H}^1(\Omega)^d \mid \mathbf{u} = \mathbf{0} \text{ on } \partial\Omega_D\}. \quad (3.11)$$

那么变分形式为：寻找 $(\mathbf{u}, p) \in (\mathbf{H}_E^1, L_2(\Omega))$ 使得

$$\int_{\Omega} \frac{\partial \mathbf{u}}{\partial t} \cdot \mathbf{v} + \nu \int_{\Omega} \nabla \mathbf{u} : \nabla \mathbf{v} + \int_{\Omega} (\mathbf{u} \cdot \nabla \mathbf{u}) \cdot \mathbf{v} - \int p (\nabla \cdot \mathbf{v}) = \int_{\Omega} \mathbf{f} \cdot \mathbf{v}, \quad (3.12)$$

$$\forall \mathbf{v} \in \mathbf{H}_{E_0}^1,$$

$$\int_{\Omega} q (\nabla \cdot \mathbf{u}) = 0, \quad (3.13)$$

$$\forall q \in L_2(\Omega).$$

其中 $\nabla \mathbf{u} : \nabla \mathbf{v}$ 表示纯量的乘积，在二维中为 $\nabla u_x \cdot \nabla v_x + \nabla u_y \cdot \nabla v_y$. 我们在构造混合元近似的时候要确保可解性条件

$$\int_{\Omega} p \nabla \cdot \mathbf{v} = 0 \quad \text{对任意的 } \mathbf{v} \in \mathbf{H}_{E_0}^1 \Rightarrow \begin{cases} p = \text{constant} & \text{如果 } \partial\Omega = \partial\Omega_D \\ p = 0 & \text{其他情况} \end{cases} \quad (3.14)$$

(3.14)可以由inf-sup 条件推出。

$$\inf_{q \neq \text{constant}} \sup_{\mathbf{v} \neq \mathbf{0}} \frac{|(q, \nabla \cdot \mathbf{v})|}{\|\operatorname{vec} \mathbf{v}\|_{H(\Omega)^1} \|q\|_{L(\Omega)^0}} \geq \gamma > 0. \quad (3.15)$$

假设 τ_h 是 Ω 上对压力网格的三角剖分，网格尺度 $h = \max_{T \in \tau_h} \operatorname{diam}(T)$, T 为三角剖分 τ_h 的单元。对应的， $\tau_{\frac{h}{2}}$ 是对速度网格的三角剖分。基于 $\tau_{\frac{h}{2}}$ 和 τ_h 上的有限元空间 X_E^h 和 P_h 满足

$$X_E^h \subset \mathcal{H}_E, \quad P_h \subset L_2(\Omega)$$

那么(3.12) 和(3.13) 可以写成如下形式: 寻找 $(\mathbf{u}_h, p_h) \in X_E^h \times P_h$ 使得

$$\begin{aligned} & \int_{\Omega} \frac{\partial \mathbf{u}_h}{\partial t} \cdot \mathbf{v}_h + \nu \int_{\Omega} \nabla \mathbf{u}_h : \nabla \mathbf{v}_h \\ & + \int_{\Omega} (\mathbf{u}_h \cdot \nabla \mathbf{u}_h) \cdot \mathbf{v}_h - \int_{\Omega} p_h (\nabla \cdot \mathbf{v}_h) = \int_{\Omega} \mathbf{f} \cdot \mathbf{v}_h, \quad \forall \mathbf{v}_h \in \mathbf{X}_0^h; \\ & \int_{\Omega} q_h (\nabla \cdot \mathbf{u}_h) = 0, \quad \forall q_h \in P^h. \end{aligned} \quad (3.16)$$

为了简便, 我们时间离散上用显示Euler格式: $\forall (\mathbf{v}_h, q_h) \in \mathbf{X}_0^h \times P^h$

$$\begin{aligned} & \int_{\Omega} \frac{\mathbf{u}_h^{(n+1)} - \mathbf{u}_h^{(n)}}{\delta t} \cdot \mathbf{v} + \nu \int_{\Omega} \nabla \mathbf{u}_h^{(n+1)} : \nabla \mathbf{v}_h - \int_{\Omega} p_h^{(n+1)} (\nabla \cdot \mathbf{v}_h) = \int_{\Omega} (\mathbf{f} + \mathbf{u}_h^{(n)} \cdot \nabla \mathbf{u}_h^{(n)}) \cdot \mathbf{v} \\ & \int_{\Omega} q_h \nabla \cdot \mathbf{u}_h^{(n+1)} = 0. \end{aligned} \quad (3.17)$$

令 $\{\phi_j\}_{j=1}^n$ 和 $\{\psi_k\}_{k=1}^m$ 分别为速度和压力的线性元基函数。则数值解 $\mathbf{u}_h^{(n+1)} = (u_{xh}^{(n+1)}, u_{yh}^{(n+1)}), p_h$ 可以写成如下形式:

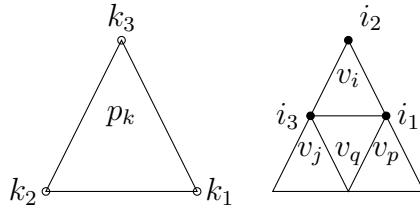
$$u_{xh}^{(n+1)} = \sum_{j=1}^n u_j \phi_j, \quad u_{yh}^{(n+1)} = \sum_{j=1}^n v_j \phi_j, \quad p_h^{(n+1)} = \sum_{k=1}^m p_k \psi_k$$

将 $u_{xh}^{(n+1)}, u_{yh}^{(n+1)}, p_h^{(n+1)}$ 带入离散弱形式(3.17) 中, 可以得到线性方程组

$$\begin{bmatrix} \frac{1}{dt} M + \nu A & 0 & B_x^T \\ 0 & \frac{1}{dt} M + \nu A & B_y^T \\ B_x & B_y & 0 \end{bmatrix} \begin{bmatrix} u_x \\ u_y \\ p \end{bmatrix} = \begin{bmatrix} f_x \\ f_y \\ g \end{bmatrix}, \quad (3.18)$$

其中 M 是 $n \times n$ 的质量矩阵, A 是拉普拉斯矩阵, 由以下形式:

$$\begin{aligned} A &= [a_{ij}], \quad a_{ij} = \int_{\Omega} \nabla \phi_i \cdot \nabla \phi_j \\ M &= [m_{ij}], \quad m_{ij} = \int_{\Omega} \phi_i \phi_j \\ B_x^T &= [bx_{ik}^T], \quad bx_{ik}^T = \int_{\Omega} \psi_k \frac{\partial \phi_i}{\partial x} \\ B_y^T &= [by_{ik}^T], \quad by_{ik}^T = \int_{\Omega} \psi_k \frac{\partial \phi_i}{\partial y} \\ f_x &= [f_i], \quad f_i = \int_{\Omega} \left(\frac{u_{xh}^{(n)}}{dt} - (u_{xh}^{(n)} \frac{\partial u_{xh}^{(n)}}{\partial x} + u_{yh}^{(n)} \frac{\partial u_{xh}^{(n)}}{\partial y}) \right) \phi_i \end{aligned}$$


 图 3.1: 左: p 单元; 右: 与压力单元对应的4个速度 v 单元

$$\begin{aligned} f_y &= [f_i], \quad f_i = \int_{\Omega} \left(\frac{u_{yh}^{(n)}}{dt} - (u_{xh}^{(n)} \frac{\partial u_{yh}^{(n)}}{\partial x} + u_{yh}^{(n)} \frac{\partial u_{yh}^{(n)}}{\partial y}) \phi_i \right. \\ g &= [g_k], \quad g_i = 0. \end{aligned}$$

接下来我们着重解释一下散度块矩阵 $B_x B_y, B_x^T, B_y^T$ 的拼装。以 B_x^T 为例, 令 Δ_{v_i} 为一个速度单元, 我们可以通过上一节建立的单元素索引来找到对应的压力单元 Δ_{p_k} (如Figure (3.1)). $\phi_{i_1}, \phi_{i_2}, \phi_{i_3}$ 是定义在速度单元 Δ_{v_i} 上的线性元基函数, 下标 i_1, i_2, i_3 表示该顶点上的自由度在速度全部自由度的全局编号。同时 $\psi_{k_1}, \psi_{k_2}, \psi_{k_3}$ 是定义在压力单元 Δ_{p_k} 上的线性元基函数, k_1, k_2, k_3 代表单元 Δ_{p_k} 上的自由度在压力全部自由度中的全局编号。因此单元 Δ_{p_k} 和 Δ_{v_i} 对 B_x^T 的贡献为

$$\begin{bmatrix} \int_{\Delta_{v_i}} \psi_{k_1} \frac{\partial \phi_{i_1}}{\partial x} & \int_{\Delta_{v_i}} \psi_{k_2} \frac{\partial \phi_{i_1}}{\partial x} & \int_{\Delta_{v_i}} \psi_{k_3} \frac{\partial \phi_{i_1}}{\partial x} \\ \int_{\Delta_{v_i}} \psi_{k_1} \frac{\partial \phi_{i_2}}{\partial x} & \int_{\Delta_{v_i}} \psi_{k_2} \frac{\partial \phi_{i_2}}{\partial x} & \int_{\Delta_{v_i}} \psi_{k_3} \frac{\partial \phi_{i_2}}{\partial x} \\ \int_{\Delta_{v_i}} \psi_{k_1} \frac{\partial \phi_{i_3}}{\partial x} & \int_{\Delta_{v_i}} \psi_{k_2} \frac{\partial \phi_{i_3}}{\partial x} & \int_{\Delta_{v_i}} \psi_{k_3} \frac{\partial \phi_{i_3}}{\partial x} \end{bmatrix} \quad (3.19)$$

将贡献矩阵(3.19) 加到 B_x^T 中的相应位置 $(i_1, k_1), (i_1, k_2), (i_1, k_3), (i_2, k_1), (i_2, k_2), (i_2, k_3), (i_3, k_1), (i_3, k_2), (i_3, k_3)$ to B_x^T . 上。通过遍历所有的速度单元, 重复上面的操作, B_x^T 拼装完成。 B_y^T 也可以同样拼装。

B_x 和 B_y 的拼装是遍历所有的压力单元, 通过单元素索引, 找到对应的四个速度单元, 然后分别用压力单元和四个速度单元进行拼装。过程相似, 便不再赘述。

我们主要的想法是有两套网格, 速度网格和压力网格, 速度网格可以由压力网格加密一次得到。因此我们对压力网格进行移动, 然后对移动完的压力网格全局加密一次, 即可实现对速度网格的同步移动。为了更好的说明我们的数值方法, 我们将算法的流程图如Algorithm(3.2)所示.

Algorithm 3.2 移动网格方法来求解Navier Stokes方程

-
- 1: 在初始速度网格 $\Delta_v^{(0)}$ 和压力网格 $\Delta_p^{(0)}$ 上, 求解 $t = t_n$ 时刻Stokes方程(3.1), 获得数值解 $\mathbf{u}_h^{(0)}, p_h^{(0)}$.
 - 2: **while** $t_n < T$ **do**
 - 3: 在压力网格 $\Delta_p^{(n)}$ 上, 用 $\mathbf{u}_h^{(n)}, p_h^{(n)}$ 来计算控制函数, 并通过求解(3.25), 获得 ξ^* .
 - 4: 判断 $\|\xi^* - \xi^{(0)}\|_{L^2}$ 是否小于容忍量 ϵ , 如果是迭代结束, 否则, 继续做5 - 8.
 - 5: 用 $\xi^* - \xi^{(0)}$ 两者之差来计算网格 $\Delta_p^{(n)}$ 的移动量 $\delta\mathbf{x}$.
 - 6: 利用5中 $\delta\mathbf{x}$, 在速度网格 $\Delta_v^{(n)}$ 上求解更新数值解的方程(3.38), 得到新网格上的中间量 $\mathbf{u}_{h,*}^{(n)}, p_{h,*}^{(n)}$.
 - 7: 更新 $\Delta_p^{(n)}$, 通过几何遗传树结构, 来同步 $\Delta_v^{(n)}$, 得到新的 $\Delta_p^{(n+1)}$ 和 $\Delta_v^{(n+1)}$
 - 8: 回到3.
 - 9: 在 $\Delta_v^{(n+1)}$ 和 $\Delta_p^{(n+1)}$ 上求解Navier-Stokes方程(3.18).从而真正获得 $t = t_{n+1}$ 时刻的数值解 $\mathbf{u}_h^{(n+1)}, p_h^{(n+1)}$.
 - 10: $n = n + 1$
 - 11: **end while**
-

3.3 移动网格策略

在 $t = t_{n+1}$ 时刻, 用上一章的方法可以得到有限元解 $(\mathbf{u}_h^{(n+1)}, p_h^{(n+1)})$. 下面的问题是如何用新的数值解和旧的网格 $\mathcal{T}_h^{(n)}$ 来获得新的网格 $\mathcal{T}_h^{(n+1)}$. 我们采取文献[?]中的方法, 注意到我们区域的边界均是Dirichlet边界, 分为以下四步:

3.3.1 Step 1 获取Monitor

选择一个合适的控制函数, 对于移动网格的结果是非常重要的。应用在不可压Navier-Stokes方程上的控制函数主要有如下几种. 令 $m = \frac{1}{G}$ 其中 m 是(3.25)中的一个纯量函数。关于 G 有集中不同的选择.一种是基于涡量

$$G_0 = \sqrt{1 + \alpha|\omega|^\beta} \quad (3.20)$$

其中 $\omega = \nabla \times \mathbf{u}$, α 和 β 是两个正的常数。

另外一个选择， G 是基于数值解的梯度

$$G_1 = \sqrt{1 + \alpha |\nabla \mathbf{u}|^\beta} \quad (3.21)$$

对于线性元 v_h 逼近真实解 v ，下面的后验误差估计公式可以用来近似计算误差

$$|v - v_h|_{1,\Omega} \sim \eta(v_h) := \sqrt{\sum_{l: \text{内部边界}} \int_l [\nabla v_h \cdot \mathbf{n}_l]^2 dl} \quad (3.22)$$

其中 $[\cdot]_l$ 意味着边 l 上的跳跃，即 $[v]_l = v|_{l+} - v|_{l-}$ 。很自然的在每个单元上等分布数值误差 $\eta(v_h)$ ，控制函数为一下形式

$$G_2 = \sqrt{1 + \alpha \eta^2(v_h)} \quad (3.23)$$

[?]中对(3.23)进行了改进

$$G_3 = \sqrt{1 + \alpha [\eta(v_h)/\max \eta(v_h)]^\beta} \quad (3.24)$$

其中 $\beta > 2$ 时有更好的效果。

3.3.2 Step 2 获取新的逻辑网格

求解椭圆形方程

$$\begin{aligned} \nabla_x (m \nabla_x \xi) &= 0 \\ \xi &= \xi_b \end{aligned} \quad (3.25)$$

其中 m 是上一节中的纯量函数，通常依赖于 $(\mathbf{u}_h^{(n+1)}, p_h^{(n+1)})$ 。我们定义初始的逻辑网格 $\mathcal{T}_c(\mathcal{A}^0$ 为它的节点)。一旦初始的逻辑网格给定，在整个的求解过程中，将一直保持不变。通过求解(3.25) 我们可以得到新的逻辑网格 $\mathcal{T}_c^*(\mathcal{A}^*$ 为它的节点)。

3.3.3 Step 3 物理网格的移动方向

我们先引入一些定义。 \mathcal{T}_h 为物理区域上的三角剖分。第*i*个点定义为 X_i ，以 X_i 为顶点的单元的集合称之为 T_i 。相应的计算区域上的标记为 \mathcal{T}_c , \mathcal{A}_i 和 $T_{i,c}$ 。

\mathcal{A}_i 点在计算区域上的坐标定义为 $(\mathcal{A}_i^1, \mathcal{A}_i^2)^T$ 。在Step 1 结束后，我们得到了新的逻辑网格 \mathcal{T}_c^* 和它的顶点 \mathcal{A}_i^* 。从而我们得到新旧逻辑网格的差：

$$\delta\mathcal{A}_i = \mathcal{A}^{(0)} - \mathcal{A}_i^* \quad (3.26)$$

对于一个给定的单元 $E \in \mathcal{T}_h$, $X_{E_k}, 0 \leq k \leq 2$, 作为它的三个顶点。从 $V_{\mathcal{T}_c^*}(\Omega)$ 到 $V_{\mathcal{T}}(\Omega)$ 的分片线性映射在单元 E 上的梯度是常数，并且满足下面的方程组：

$$\begin{aligned} & \begin{pmatrix} \mathcal{A}_{E_1}^{*,1} - \mathcal{A}_{E_0}^{*,1} & \mathcal{A}_{E_2}^{*,1} - \mathcal{A}_{E_0}^{*,1} \\ \mathcal{A}_{E_1}^{*,2} - \mathcal{A}_{E_0}^{*,2} & \mathcal{A}_{E_2}^{*,2} - \mathcal{A}_{E_0}^{*,2} \end{pmatrix} \begin{pmatrix} \frac{\partial x^1}{\partial \xi^1} & \frac{\partial x^1}{\partial \xi^2} \\ \frac{\partial x^2}{\partial \xi^1} & \frac{\partial x^2}{\partial \xi^2} \end{pmatrix} \\ &= \begin{pmatrix} X_{E_1}^1 - X_{E_0}^1 & X_{E_2}^1 - X_{E_0}^1 \\ X_{E_1}^2 - X_{E_0}^2 & X_{E_2}^2 - X_{E_0}^2 \end{pmatrix} \end{aligned}$$

求解上面的方程组，可以获得单元 E 上的 $\partial \mathbf{x} / \partial \xi$ 。如果以单元的面积作为权重，则第*i*个点的加权平均的位移定义如下：

$$\delta X_i = \frac{\sum_{E \in T_i} |E| \frac{\partial \mathbf{x}}{\partial \xi}|_{in E} \delta \mathcal{A}_i}{\sum_{E \in T_i} |E|}. \quad (3.27)$$

其中 $|E|$ 代表单元 E 的面积。为了避免网格发生缠结，在网格移动向量前乘上一个常量 μ ，即物理区域上新网格 \mathcal{T}^* 的节点表示为：

$$X_i^* = X_i + \mu \delta X_i. \quad (3.28)$$

文献[?] 中提出 μ 按以下方式给出：

$$\begin{vmatrix} 1 & 1 & 1 \\ x_0^1 + \mu \delta x_0^1 & x_1^1 + \mu \delta x_1^1 & x_2^1 + \mu \delta x_2^1 \\ x_0^2 + \mu \delta x_0^2 & x_1^2 + \mu \delta x_1^2 & x_2^2 + \mu \delta x_2^2 \end{vmatrix} = 0 \quad (3.29)$$

其中 $\mathbf{x}_i = (x_i^1, x_i^2), 0 \leq i \leq 2$ 表示第*i*个点的坐标。令 μ_i^* 为方程(3.29)的最小正根，则令

$$\mu = \min(1, \frac{\mu_i^*}{2}). \quad (3.30)$$

3.3.4 Step 4 散度为0的插值

用移动网格方法求解不可压流体时，要保证插值的过程散度是为0的。通过求解一个对流方程，对流的速度是网格的移动速度，从而实现旧的物理网格上的数值解到新的物理网格上数值解的插值。令 $u_h = \sum u_i \phi_i$, $u_h \in \mathcal{X}_E^h$, ϕ_i 是有限元空间 \mathcal{X}_h 的基函数。引入一个虚拟的时间 τ , 假设基函数 ϕ_i 和 u_i 均是关于 τ 的函数, 即 $\phi_i = \phi_i(\tau)$, $u_i = u_i(\tau)$. 我们引入一个从旧网格 $x^{\text{旧}}$ 到新网格 $x^{\text{新}}$ 网格点的连续变换:

$$x_i(\tau) = X_i + \tau(X_i^* - X_i), \quad \tau \in [0, 1] \quad (3.31)$$

其中 $X_i^* = x_i^{\text{新}}$, $X_i = x_i^{\text{旧}}$ 基于(3.31)的连续形式 $x(\tau) = x^{\text{旧}} + \tau(x^{\text{新}} - x^{\text{旧}})$, 基函数可以定义为 $\phi_i(\tau) = \phi_i(x(\tau))$ 并且 $u_i = u_i(x(\tau))$.

在插值的过程中，我们要保持解曲线 $u_h = \sum u_i \phi_i$ 关于 τ 在弱形式下是不变的. 即对 $\forall \psi \in \mathcal{X}_h$, $(\partial_\tau u_h, \psi) = 0$ 。通过直接计算可得

$$\frac{\partial \phi}{\partial \tau} = -\nabla_{\mathbf{x}} \phi_i \cdot \delta \mathbf{x} \quad (3.32)$$

其中 $\delta \mathbf{x} = x^{\text{旧}} - x^{\text{新}}$ 。紧接着

$$\begin{aligned} 0 &= (\partial_\tau u_h, \psi) \\ &= (\partial_\tau \sum u_i(x(\tau)) \phi_i, \psi) \\ &= (\sum \phi_i \partial_\tau u_i(x(\tau)) + \sum u_i \partial_\tau \phi_i) \\ &= (\sum \phi_i \partial_\tau u_i(x(\tau)) - \sum u_i \nabla_{\mathbf{x}} \phi_i \cdot \delta \mathbf{x}, \psi) \\ &= (\sum \phi_i \partial_\tau u_i(x(\tau)) - \nabla_{\mathbf{x}} u_h \cdot \delta \mathbf{x}, \psi) \end{aligned} \quad (3.33)$$

我们将(3.33)应用到不可压流上，即速度场要满足散度为0的条件。令 \mathcal{X}_h 为散度为0的空间:

$$\mathcal{X}_E^h = X_E^h \cap \{\mathbf{u}_h | \nabla \cdot \mathbf{u}_h = 0\} \quad (3.34)$$

那么(3.33)将变成: 寻找 $w_h \in \mathcal{X}_h$ 使得

$$\left(\sum \phi_i \partial_\tau u_i - \sum u_i \nabla_{\mathbf{x}} \phi_i \cdot \delta \mathbf{x}, z_h \right) = 0 \quad \forall z_h \in \mathcal{X}_h. \quad (3.35)$$

上面的结果意味着

$$\sum \phi_i \partial_\tau u_i - \sum u_i \nabla_{\mathbf{x}} \phi_i \cdot \delta \mathbf{x} \in \mathcal{X}_h^\perp \quad (3.36)$$

其中 $\mathcal{X}_h^\perp + \mathcal{X}_h = L^2$. 根据文献[?]中的定理2.7, 如果区域 Ω 是单连通的, 那么

$$\mathcal{X}_h^\perp = \{\nabla q | q \in H^1(\Omega)\} \quad (3.37)$$

则存在 $\nabla p \in \mathcal{X}_h^\perp$ 使得

$$\begin{aligned} \sum \phi_i \partial_\tau u_i - \sum u_i \nabla_x \phi_i \cdot \delta \mathbf{x} &= -\nabla p \\ \nabla_x \cdot u_h &= 0. \end{aligned} \quad (3.38)$$

注 3.1. 这里的 p 跟外部 Navier-Stokes 方程的解 p 不一致, 只是一个辅助量。

(3.38) 的弱形式: 寻找 $(\mathbf{u}_h, p_h) \in X_E^h \times P_h$ 使得

$$\begin{aligned} \left(\sum \phi_i \partial_\tau u_i - \sum u_i \nabla_x \phi_i \cdot \delta \mathbf{x}, v_h \right) &= (p_h, \nabla v_h), \quad \forall v_h \in X_E^h. \\ (\nabla_x \cdot u_h, q_h) &= 0, \quad \forall q_h \in P_h \end{aligned} \quad (3.39)$$

(3.38) 和 (3.39) 的初值为在 $t = t_n$ 时刻的网格上, $t = t_{n+1}$ 时刻外部 Navier-Stokes 方程的解。

时间方向的离散我们暂时先用线性 Euler 方法:

$$\begin{aligned} \left(\frac{\sum \phi_i u_{i,*}^{(n)} - \sum \phi_i u_i^{(n)}}{\Delta \tau}, v_h \right) - \left(\sum u_i^{(n)} \nabla \phi_i \cdot \delta \mathbf{x}, v_h \right) &= (p_h^{(n)}, \nabla v_h). \\ (\nabla \cdot u_{h,*}^{(n)}, q_h) &= 0 \end{aligned} \quad (3.40)$$

注意到在做数值解插值的过程中, 物理网格还没有发生移动, 这时候基函数 ϕ 仍然是 $t = t_n$ 时刻网格上的基函数。所以 $u_{h,*}^{(n)} = \sum u_{i,*}^{(n)} \phi_i^{(n)}$, $u_h^{(n)} = \sum u_i^{(n)} \phi_i^{(n)}$ 简单整理一下:

$$\begin{aligned} \left(\frac{u_{h,*}^{(n)} - u_h^{(n)}}{\Delta \tau}, v_h \right) - \left(\nabla u_h^{(n)} \cdot \delta \mathbf{x}, v_h \right) &= (p_h^{(n)}, \nabla v_h). \\ (\nabla \cdot u_{h,*}^{(n)}, q_h) &= 0 \end{aligned} \quad (3.41)$$

其中 $u_h^{(n)}$ 和 p_h 是在 $t = t_n$ 时刻的网格上, $t = t_{n+1}$ 时刻, Navier-Stokes 方程的数值解。而 $u_{h,*}^{(n)}$ 和 $p_{h,*}^{(n)}$ 是在新的网格上 $t = t_{n+1}$ 时刻更新的解。但这组解不能当作外部 Navier-Stokes 方程的解。需要在新网格上重新求解 Navier-Stokes 方程, 得到的解才是我们想要的解。

3.4 数值算例

3.4.1 Colliding Flow

这个例子为稳态Stokes方程的精确解，粘性系数 $\nu = 1.0$:

$$u_x = 20xy^3; \quad u_y = 5x^4 - 5y^4; \quad p = 60x^2y - 20y^3 + \text{constant}. \quad (3.42)$$

其中计算区域 $\Omega = [-1, -1] \times [1, 1]$, 边界条件全部是Dirichlet 条件。这个例子是用来检验移动网格方法的收敛阶，此时解比较光滑。从文献[?] 可知，我们期望移动网格的收敛阶：速度有二阶收敛，压力一阶收敛。我们先给出均匀网格时，误差的收敛阶，如Table(??) 和Table(??)所示。

网格	$\ \mathbf{u} - \mathbf{u}_h\ _{L^2}$	误差阶	$\ \mathbf{u} - \mathbf{u}_h\ _{H^1}$	$\ p - p_h\ _{L^0}$	误差阶	$\ p - p_h\ _{H^1}$
10×10	1.18×10^{-1}		3.37×10^0	8.49×10^{-1}		1.84×10^1
20×20	2.94×10^{-2}	2.01	1.67×10^0	2.330×10^{-1}	1.82	1.06×10^1
40×40	7.37×10^{-3}	1.99	8.35×10^{-1}	6.54×10^{-2}	1.78	5.89×10^0
80×80						

表 3.1: 用移动网格计算碰撞流的误差, $\nu = 1.0$.

网格	涡量 L^2 误差	误差阶	散度 L^2 误差	误差阶
10×10	2.35×10^0		2.41×10^0	
20×20	1.16×10^0	1.01	1.20×10^0	1.00
40×40	5.79×10^{-1}	1.00	6.01×10^{-1}	1.00
80×80				

表 3.2: 用移动网格计算碰撞流的散度和涡量误差, $\nu = 1.0$.

我们采用上面一章中的移动网格方法来求解这个算例。选取(3.21)为控制函数，其中 $\mathbf{u} = (u_x, u_y)^T$. α 和 β 分别取为0.002和2. 从Table(3.1)中可以看出速度 L^2 误差有二阶，压力收敛阶是一阶。从Table(3.2)可以看出速度散度和涡量均有一阶收敛。

首先我们判断网格是不是往正确的方向移动。从Figure(3.2)中看出控制函数 G_1 最大的地方分布在区域的四个顶角上，中间区域控制函数的值是比较小

的。网格应该从控制函数小的地方移动到控制函数值大的地方。而网格的移动方向也确实是往四个顶角上移动。因此我们可以确定网格移动方向是对的。再者，判断选取 G_1 为控制函数是否合适。从Figure (3.3)中看到：在均匀网格下，速度 L^2 误差最大的地方分布在四个顶角上，这与控制函数的分布是一致的。所以选取 G_1 为控制函数是合理的。

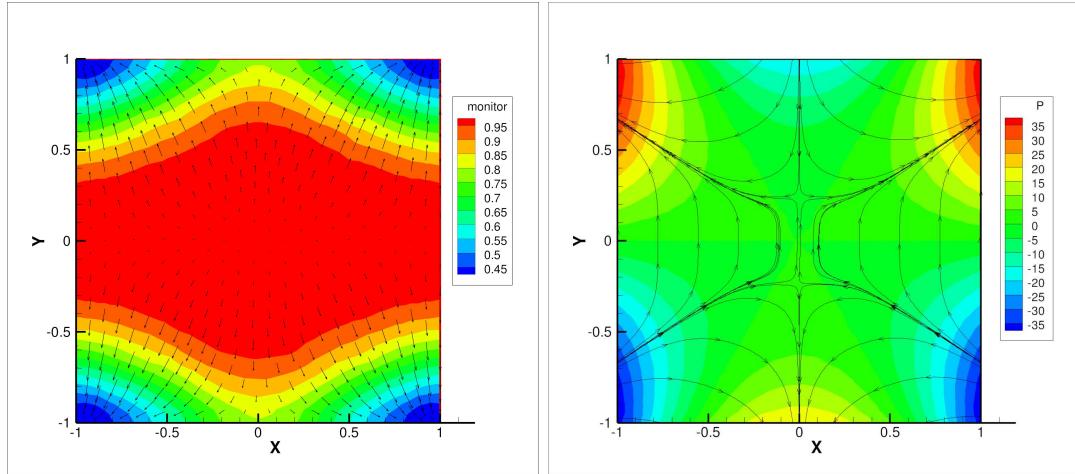


图 3.2: 左: $m = \frac{1}{G_1}$ 的等高线和网格移动方向; 右: 压力等高线和速度的流速线 $\alpha = 0.002, \beta = 2.0$.

3.4.2 喷射流

这个算例模拟的是一个很细的流体喷射入一个静止的流场，这种现象可以参考[?]，一本有趣的画册。我们的计算区域是 $\Omega = [0, 12] \times [-3, 3]$ 并且粘性系数是 $\nu = 0.0005$ 。自然条件设置在出流边界 $x = 12$ 上，同时入流边 $x = 0, y \in [-0.1, 0.1]$ 上设置Poiseuille流条件：

$$u_x = 1 - 100y^2, u_y = 0. \quad (3.43)$$

无滑移边界条件设置在 $\partial\Omega$ 的其余部分。

在移动网格策略中，我们选取(3.20)作为控制函数，参数 $\alpha = 2.0, \beta = 2.0$ 。据我们所知，当细流喷射到静止的流场中，会产生两个对称的涡。为了提高计算精度，在入射流周围需要更多的网格。从图3.5中可以看出，网格集中在涡量值比较大的地方。随着时间的发展，流体的不稳定现象会出现，这跟[?]中的物理现象是一致的。

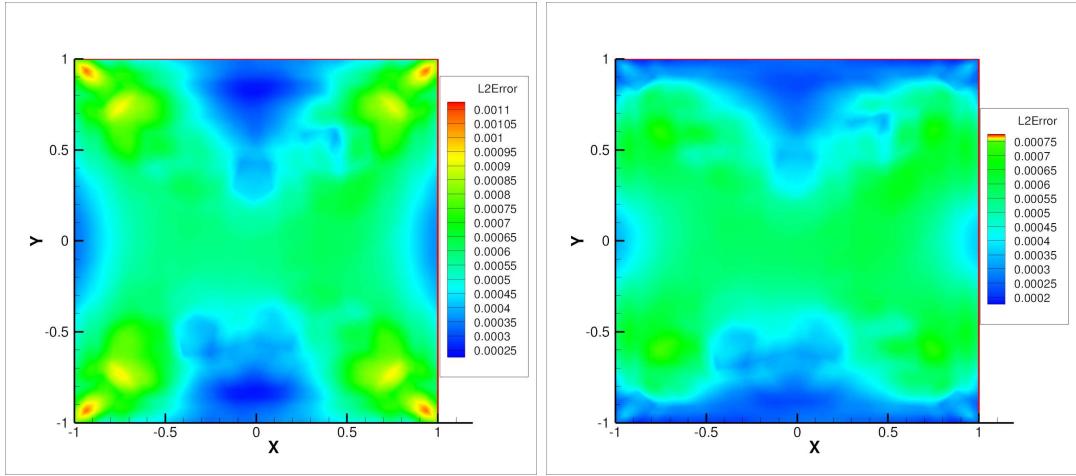


图 3.3: 速度 L^2 误差分布, 左: 均匀网格; 右: 移动网格 $\alpha = 0.002, \beta = 2.0$.

3.4.3 Navier-Stokes 流经过一个凹形台阶

我们考虑[?]中的算例, Navier-Stokes 流体以 $Re = 1000$ 经过一个凹形的台阶, 计算区域是 $\Omega = (0, 4) \times (0, 1) / (1.2, 1.6) \times (0, 0.4)$, 在上下边界上, 设置 $\mathbf{u} = (0, 0)^T$ 的边界条件。入流边上的边界条件为 $\mathbf{u} = (4y(1 - y), 0)$, 出流边是自然边界条件。

我们选取涡量作为控制函数, 控制函数的参数是 $\alpha = 0.4, \beta = 2.0$ 。众所周知, 在这个问题中流体在凹进去的角上, 会出现奇异性。从图3.7中可以看出, 初始的网格集中在凹进去的边上。 $t = 0.5s, t = 1s$, and $t = 2s$ 时刻的网格以及涡量的等高线在图3.8, 3.9 和3.11中展示出来。我们发现, 我们的网格跟涡的结构是一致的。

$t = 2s$ 时, 压力的等高线以及速度的流速线在图3.12中给出。我们可以看出, 在拐角处, 压力的等高线是平滑的。

3.4.4 圆柱绕流

这个是经典算例, 在一个长方形的管道中, 流体流过一个圆柱。在[?]中, 作者用移动有限元方法来求解这个模型。圆柱圆心是 $(0, 0)$, 半径是 $r = 0.3$ 。粘性系数是 $\nu = 0.003$, 计算区域是 $\Omega = [-1, 5] \times [-1, 1]$ 。Poiseuille流 $u = 1 - y^2, v = 0$ 设置在入流边界 $x = -1$ 。边界条件 $u = v = 0$ 施加在管道的上下边界。出流边界 $x = 5$ 是自然边界条件。

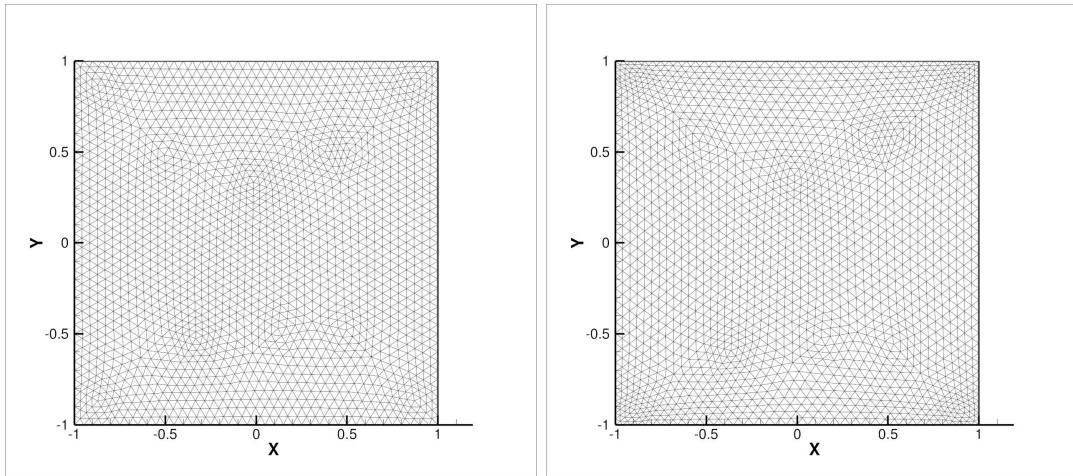


图 3.4: 网格对比, 左: 均匀网格 20×20 ; 右: 移动网格 $\alpha = 0.002, \beta = 2.0$.

如果我们关注流体的细微结构, 因此需要在小结构上有高分辨率。为了抓住涡的结构, 在我们的移动策略中, (3.20) 作为控制函数是一个很好的选择。具体的参数是 α 和 β 分别为1.0, 2.0。

在图3.13 和图3.14 中, 展示了网格和涡的等高线的变化。从中可以看出, 我们的网格可以抓住涡的结构, 并且我们的网格质量比[?]中要好。当我们改变雷诺数为 $Re = 2000$ 时, 从图3.15中可以看出, 网格和涡量的结构并不像 $Re = 2000/3$ 时那么明显。

where

$$\begin{aligned}\mathcal{D}_3 &:= \{z \in \mathbb{C} : 1 - z \in \overline{\mathcal{D}}_1\}, \\ \mathcal{D}_4 &:= \left\{z \in \mathbb{C} : |\arg(z)| < \frac{\pi}{4}, |1 - z| < \frac{31}{24}\right\}.\end{aligned}$$

3.5 Proof of Theorem ??

3.5.1 Technical lemmas

$$\begin{cases} |r(E(z), \lambda)| < |r(z, \lambda)|^3, & \text{if } r(z, \lambda) \in \overline{\mathcal{D}}_1 \setminus \{0\} \text{ (44)} \\ |r(E(z), \lambda)| \leq \sup_{m \geq 3} \left\{ \left| \frac{S_m(u)}{u^3} \right| \right\} \cdot \frac{|u| + |r(z, \lambda)|}{|u| - |r(z, \lambda)|} \cdot |r(z, \lambda)|^3, & \text{if } r(z, \lambda) \in \mathcal{D}_0 \setminus \{1\} \text{ (3.45)} \end{cases}$$

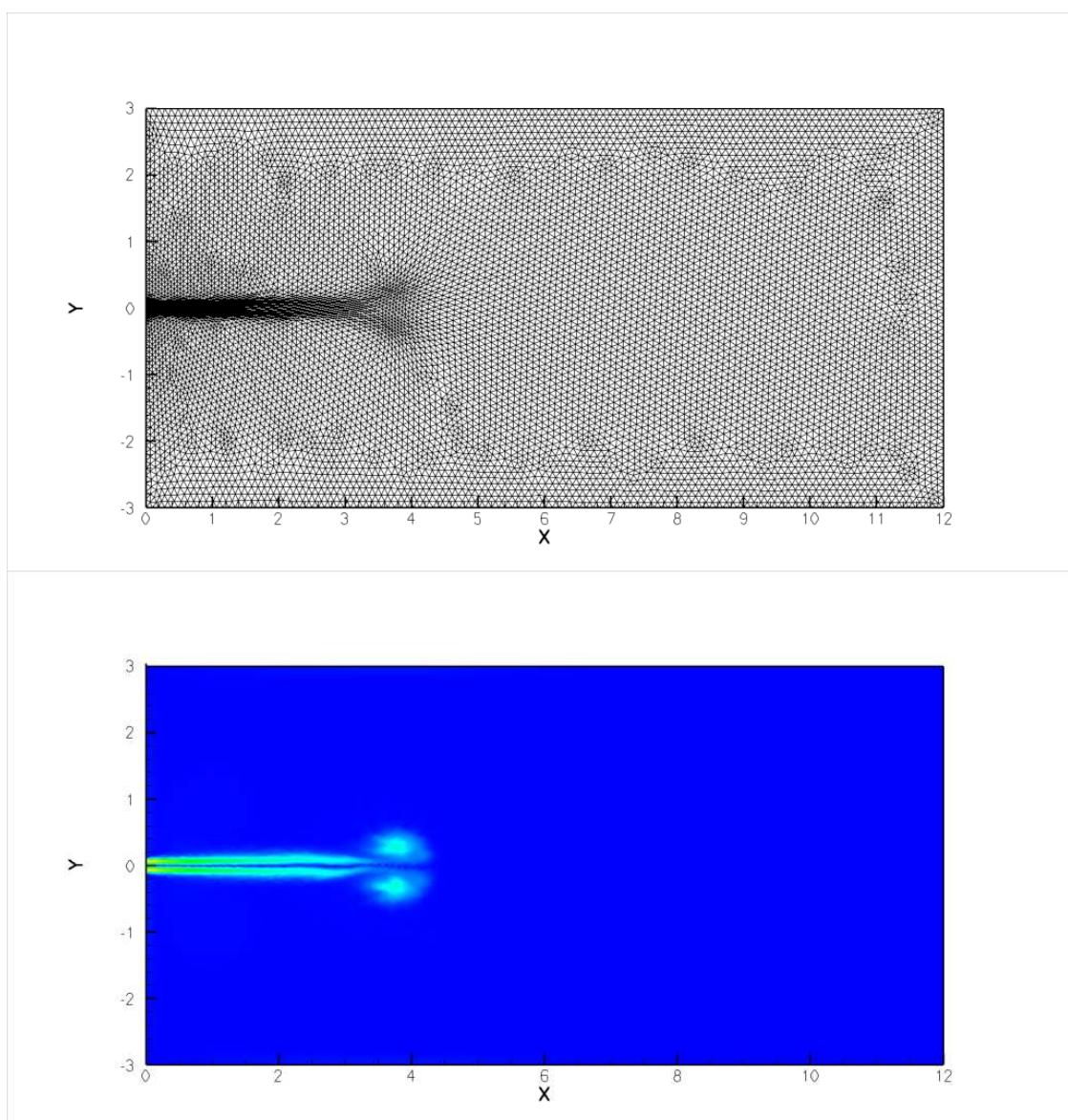


图 3.5: Jetting flow: top: mesh, bottom: vorticity contour at $t = 12s$.

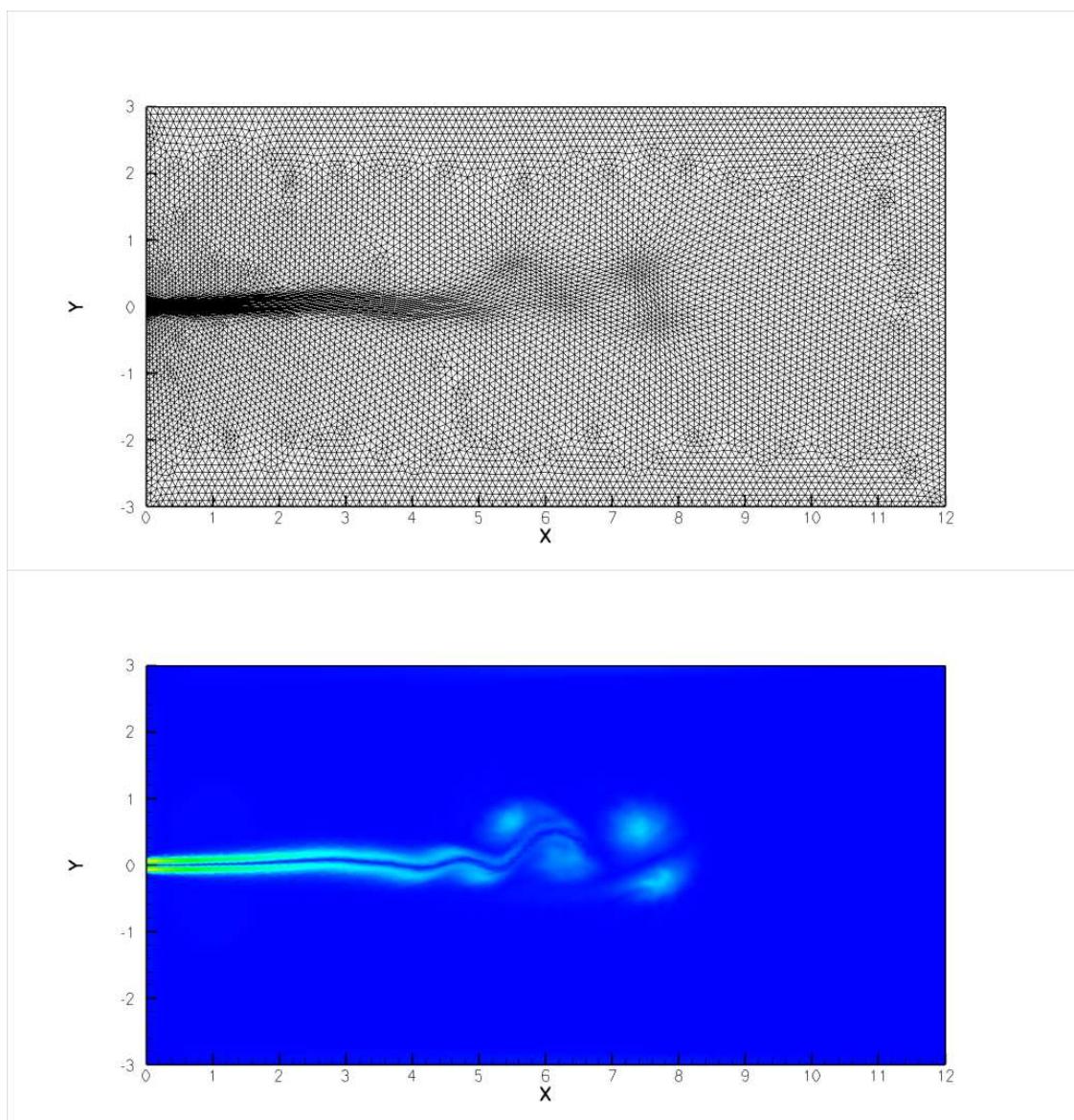


图 3.6: Jetting flow: top: mesh, bottom: vorticity contour at $t = 27s$.

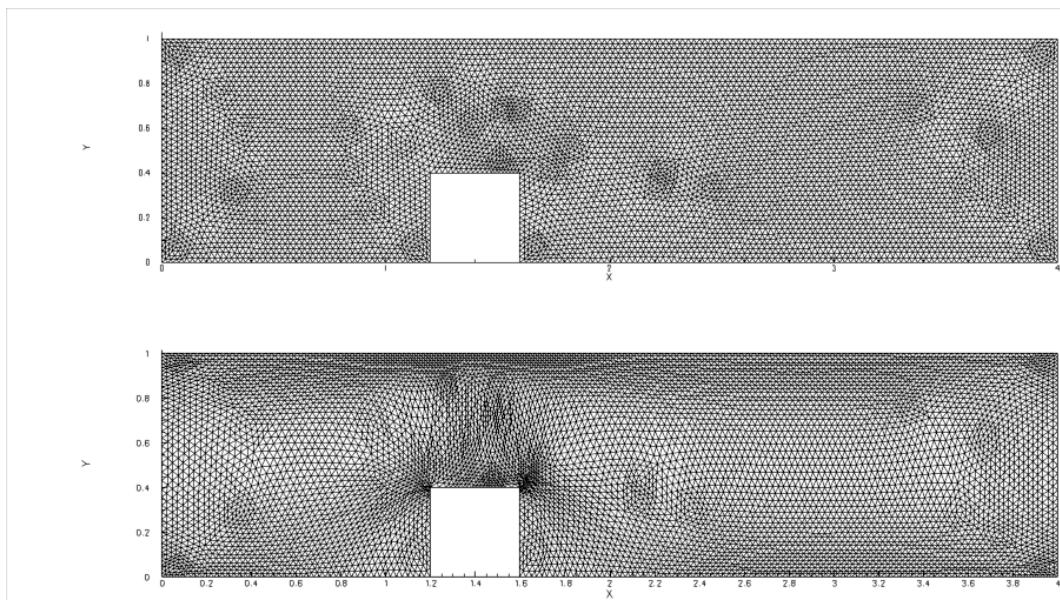


图 3.7: Top: initial uniform mesh; bottom: initial moving mesh.

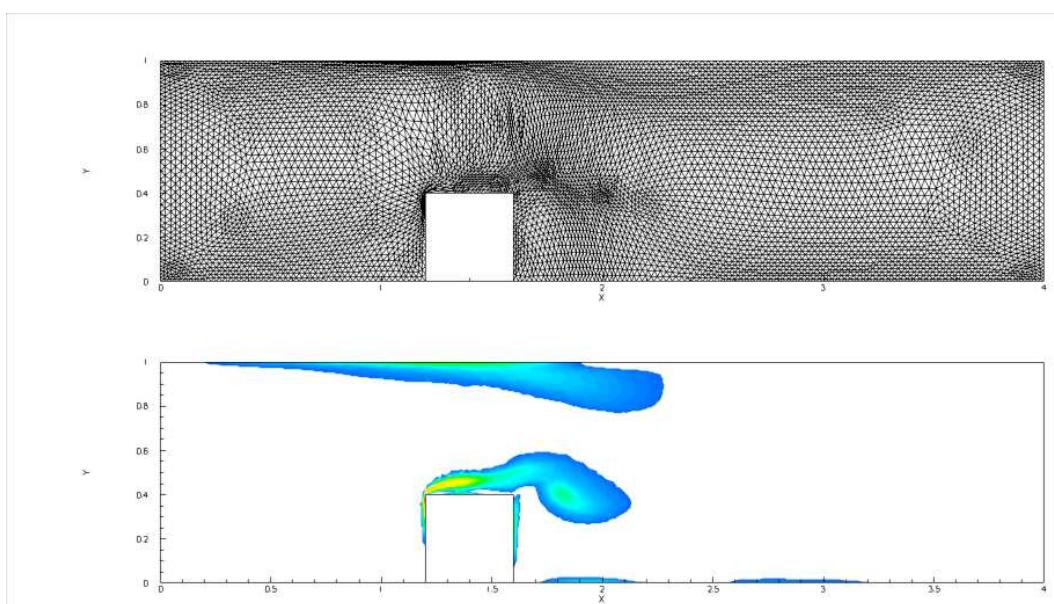


图 3.8: Top: moving mesh; bottom: contour of vorticity at $t = 0.5s$

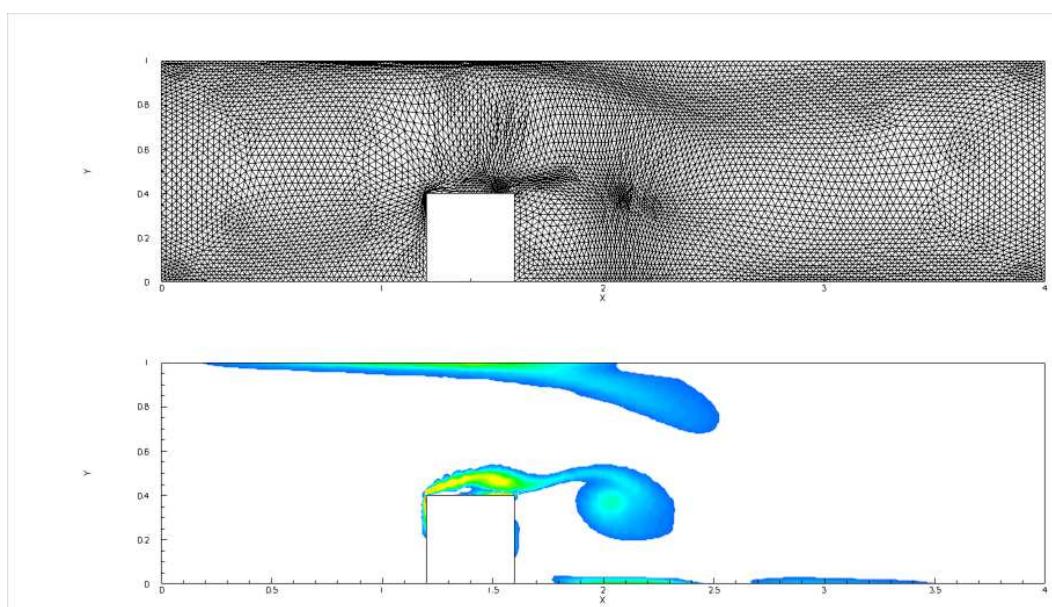


图 3.9: Top: moving mesh; bottom: contour of vorticity at $t = 1$ s

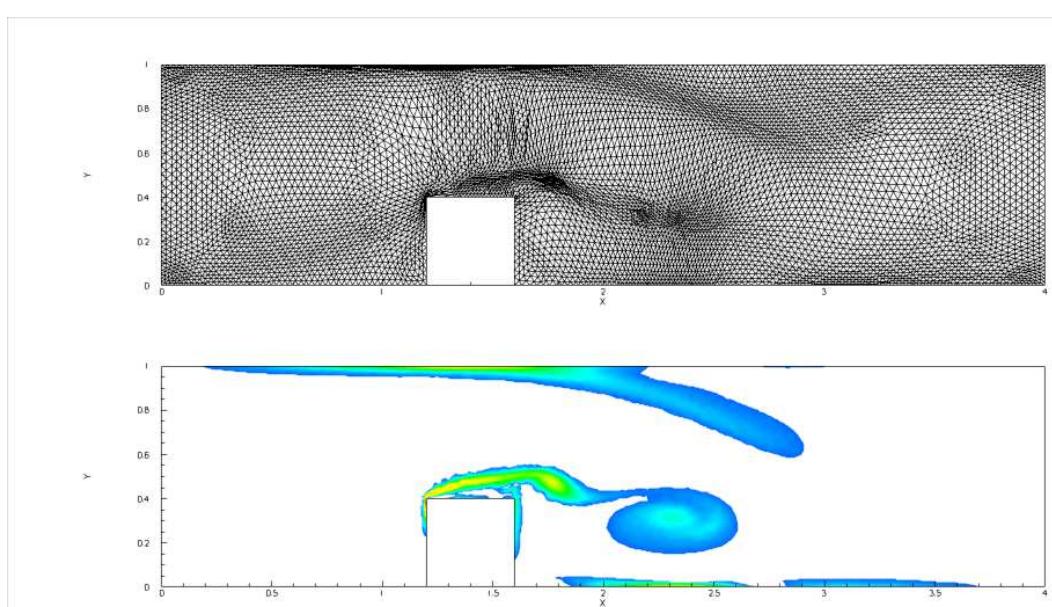


图 3.10: Top: moving mesh; bottom: contour of vorticity at $t = 1.5$ s

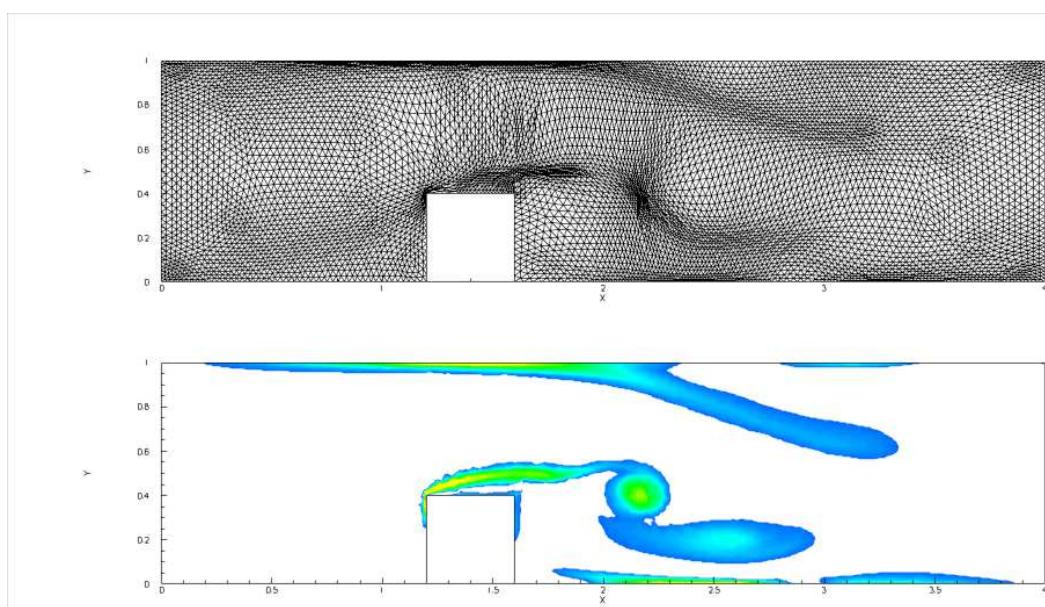


图 3.11: Top: moving mesh; bottom: contour of vorticity at $t = 2\text{s}$

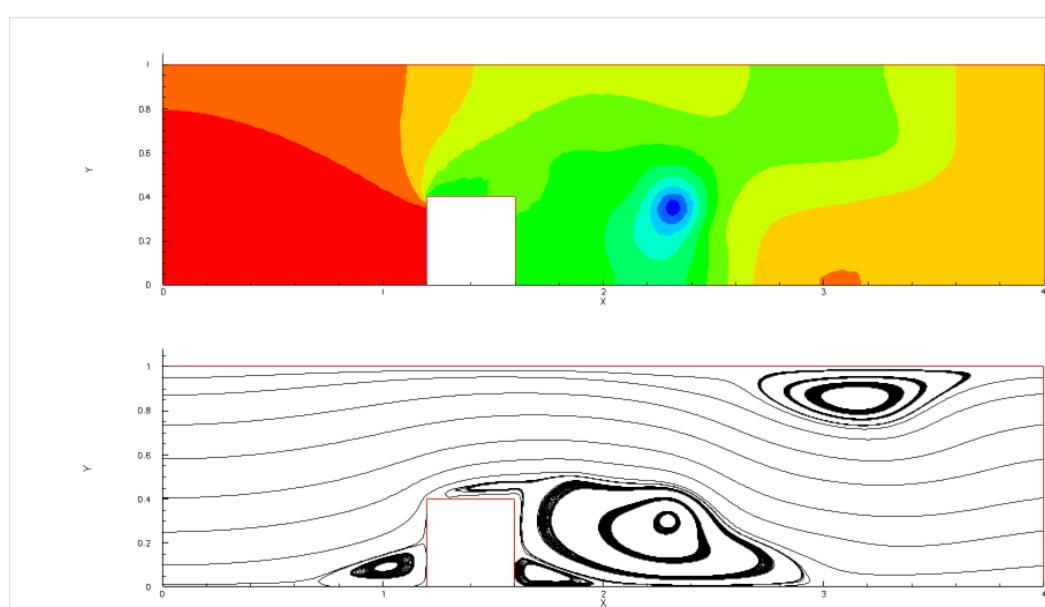


图 3.12: Top: pressure contour; bottom: velocity streamline at $t = 2\text{s}$

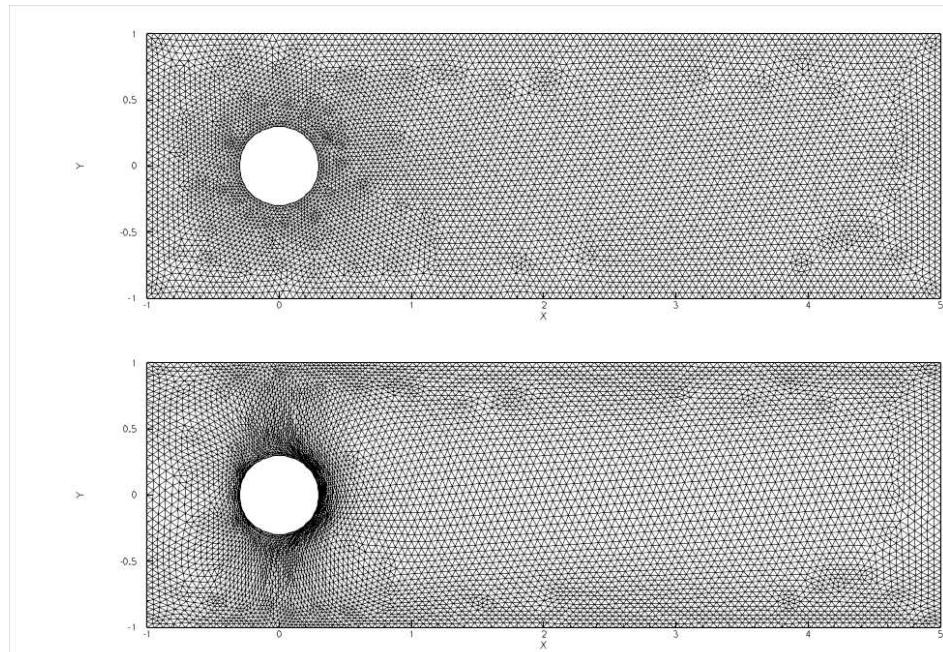


图 3.13: Top: uniform mesh; bottom: initial moving mesh, viscosity $\nu = 0.003$.

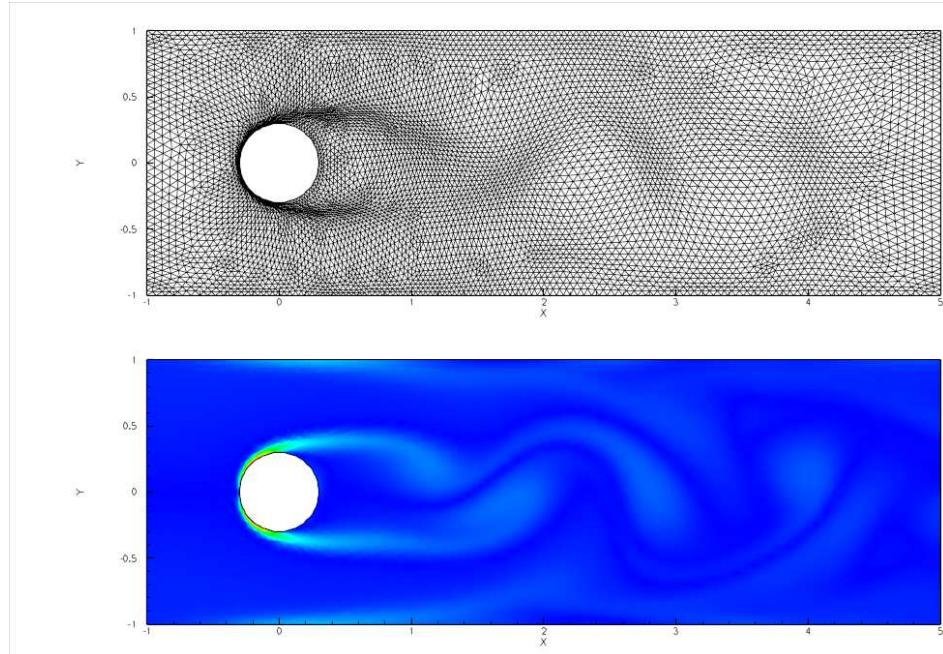


图 3.14: Top: moving mesh; bottom: vorticity contour at $t = 24.5s$, viscosity $\nu = 0.003$.

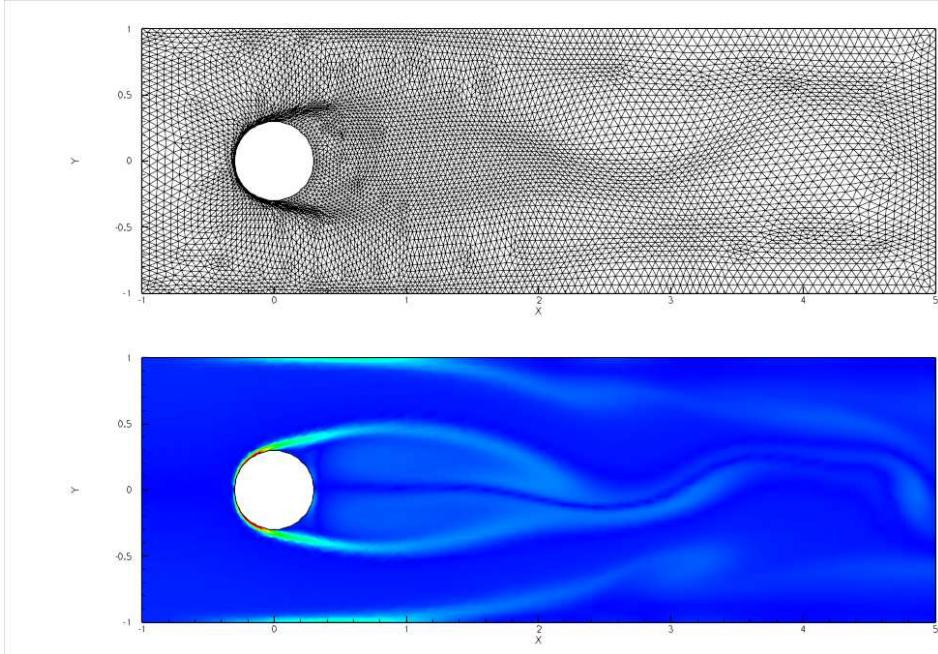


图 3.15: Top: moving mesh; bottom: vorticity contour at $t = 23s$, viscosity $\nu = 0.001$.

3.6 Numerical Experiments

TEST 4. Consider the following two matrices which take from [?] and [?], respectively.

$$S_1 = \begin{bmatrix} 0.44 & -0.88 & -0.38 & -0.50 \\ 0.68 & 2.15 & 0.48 & 0.11 \\ 0.61 & 0.77 & 2.14 & 1.04 \\ -0.16 & -0.30 & -0.67 & 1.33 \end{bmatrix}, \quad S_2 = \begin{bmatrix} -1 & -2 & 2 \\ -4 & -6 & 6 \\ -4 & -16 & 13 \end{bmatrix}.$$

Let $A_1 = S_1^5$ and $A_2 = S_2^{15}$. The eigenvalues of A_1 are $15.2477, 0.2724 \pm 16.0066i, 1.1030$ and of A_2 are $1, 2, 3$. We now compute $A_1^{1/5}$ and $A_2^{1/15}$ using the four algorithms. The computational results are given in Table 3.3. The relative error is computed by $\|X - S\|/\|S\|$ and the relative residuals is computed by using (??). As is shown in Table 3.3, Algorithm ?? has a less pronounced advantage for computing $A_1^{1/5}$ and $A_2^{1/15}$.

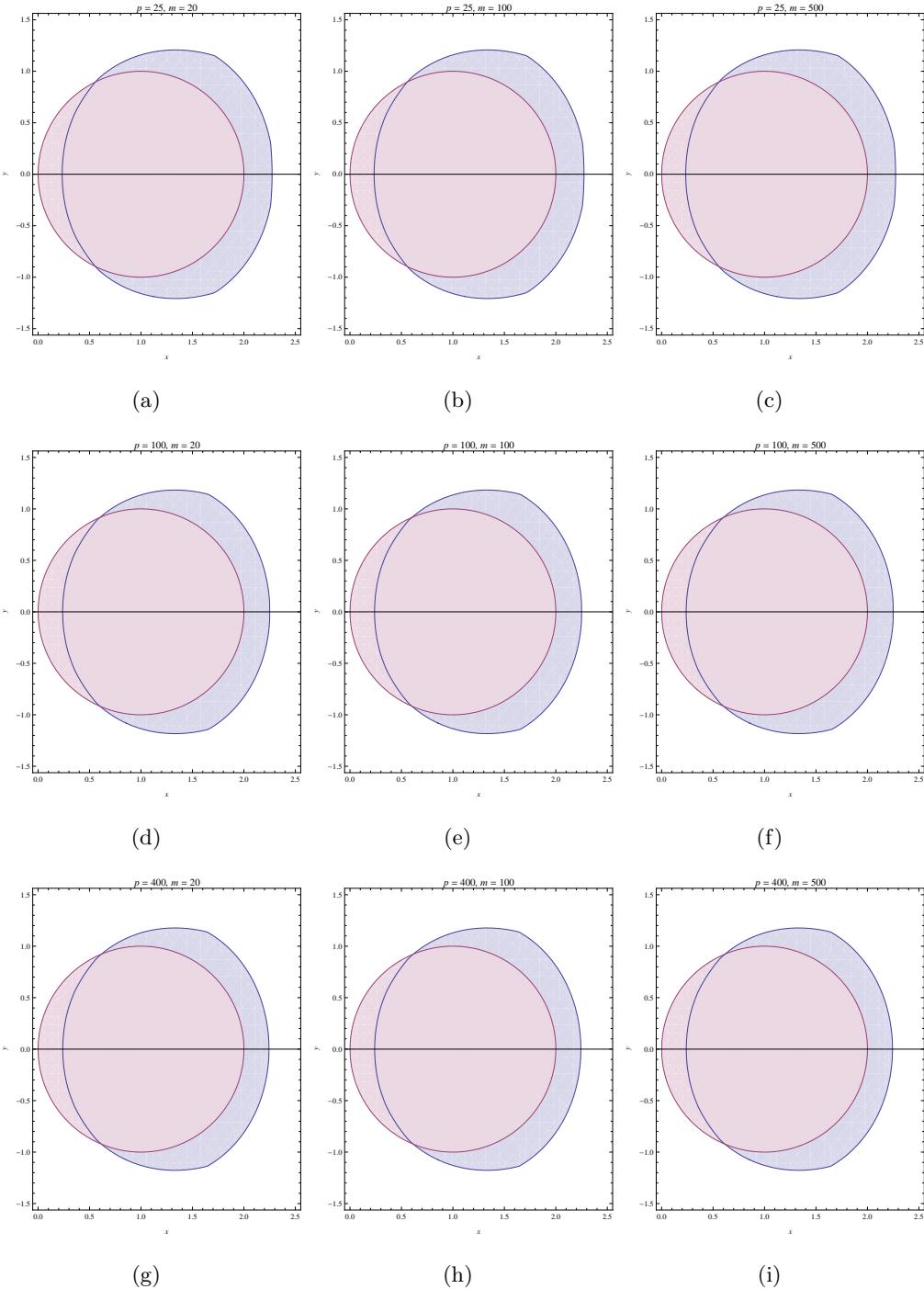


图 3.16: The approximating regions of \mathcal{R} defined in (??) for $p = 25, 100, 400$ and $m = 20, 100, 500$, where the red and blue regions denote the set $\{z \in \mathbb{C} : 1 - z \in \overline{\mathcal{D}}_1\}$ and $\{z \in \mathbb{C} : 1 - z \in \mathcal{D}_0 \cap \mathcal{D}_2\}$, respectively.

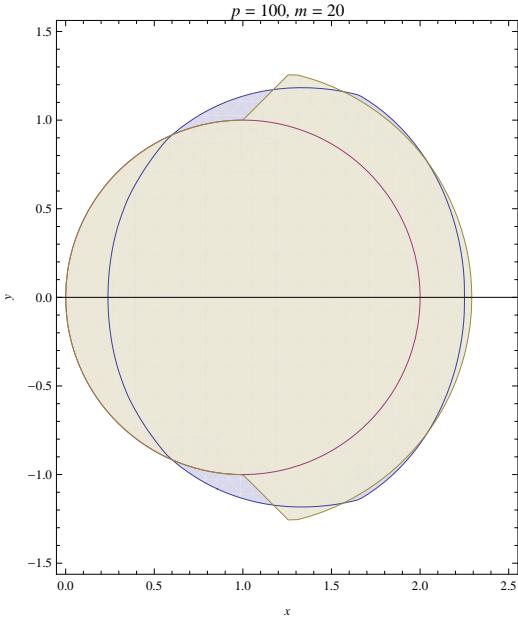


图 3.17: For $p = 100, m = 20$, the actual convergence regions \mathcal{R} defined in (??) (the union of the red and blue parts) and the approximate convergence regions \mathcal{R}_E defined in (??) of Euler's method (the yellow parts).

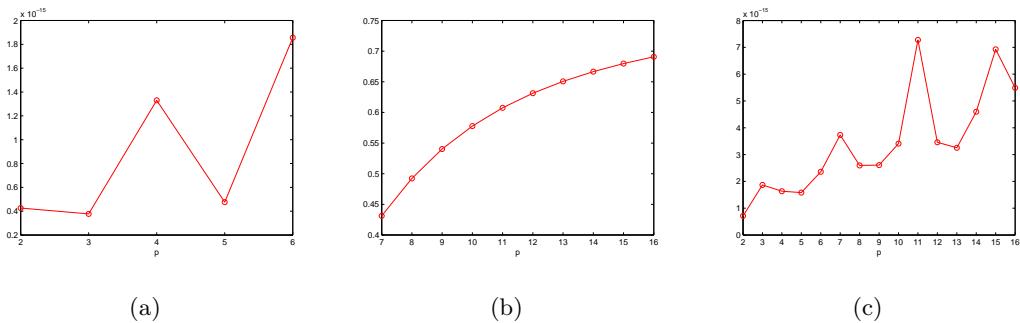


图 3.18: (a) and (b) show the relative error for the p th root $A^{1/p}$ by using coupled Euler iteration (??); (c) is the result of Algorithm ??.

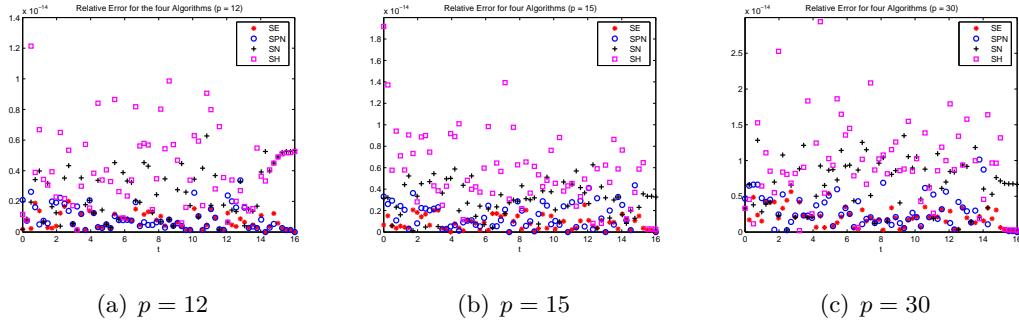


图 3.19: The relative errors for the four algorithms on matrix (??).

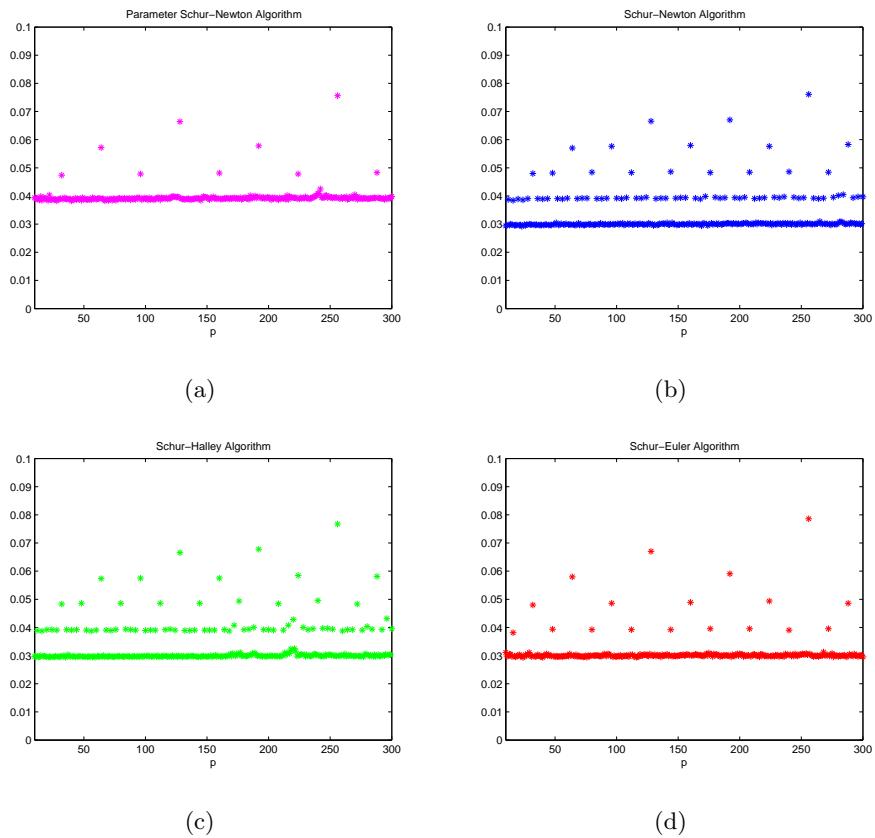


图 3.20: The CPU time (in seconds) required by using the four Algorithms to compute the principal p th root of 15×15 Frank matrix for $p = 10 : 300$.

Algorithm	CPU time (s)	iter	k_1	$\rho(X)$	err(X)
$A_1^{1/5}$					
PSN	3.91e-03	5	2	1.75e-15	2.05e-15
SN	3.28e-03	6	2	4.67e-16	1.05e-15
SH	3.13e-03	3	2	6.15e-16	9.63e-16
SE	3.59e-03	3	3	8.77e-16	1.55e-15
$A_2^{1/15}$					
PSN	6.09e-03	5	5	1.48e-15	2.67e-08
SN	5.17e-03	6	4	5.74e-15	2.67e-08
SH	5.00e-03	3	4	7.82e-15	2.67e-08
SE	4.53e-03	5	5	2.25e-14	2.67e-08

表 3.3: Results for computing $A_1^{1/5}$ and $A_2^{1/15}$ by using the four Algorithms.

Algorithm	CPU time (s)	iter	k_1	$\rho(X)$	err(X)
$p = 5$					
PSN	2.34e-03	5	3	8.68e-15	3.11e-12
SN	2.97e-03	5	2	8.62e-15	3.08e-12
SH	3.13e-03	3	2	8.54e-15	3.06e-12
SE	1.41e-03	3	2	8.52e-15	3.05e-12
$p = 7$					
PSN	5.00e-03	5	3	1.45e-14	3.83e-12
SN	4.22e-03	4	2	1.50e-14	3.96e-12
SH	4.38e-03	3	2	1.55e-14	4.07e-12
SE	4.06e-03	4	2	1.47e-14	3.87e-12

表 3.4: Results for a random nonnormal matrix by using the four Algorithms.

	$p = 18$					$p = 33$					$p = 81$					第3章 基于4PI-PI求解NAVIER-STOKES方程的移动网格方法
	CPU time	iter	k_1	$\rho(X)$	err(X)	CPU time	iter	k_1	$\rho(X)$	err(X)	CPU time	iter	k	$\rho(X)$	err(X)	
Hilbert matrix (7×7)																
PSN	1.06e-02	5	5	1.31e-14	6.47e-14	1.06e-02	5	5	3.77e-14	2.12e-13	1.05e-02	5	5	6.77e-14	4.30e-13	15
SN	8.44e-03	6	4	6.09e-15	3.01e-14	8.59e-03	6	4	1.65e-14	9.31e-14	8.59e-02	6	4	2.63e-14	1.67e-13	4
SH	8.59e-03	4	4	1.34e-15	6.60e-15	8.59e-03	4	4	2.43e-14	1.37e-13	8.59e-03	4	4	2.51e-14	1.60e-13	4
SE	7.03e-03	5	3	3.19e-15	1.57e-14	8.44e-03	4	4	4.53e-15	2.55e-14	8.44e-03	4	4	2.45e-14	1.56e-13	4
Prolate matrix (10×10)																
PSN	2.16e-02	5	5	3.96e-15	3.45e-14	2.03e-02	5	5	1.33e-14	1.03e-13	2.31e-02	4	5	3.79e-14	3.64e-13	15
SN	1.81e-02	6	4	1.29e-15	1.12e-14	1.81e-02	6	4	5.74e-15	5.26e-14	1.77e-02	6	4	9.05e-15	8.70e-14	4
SH	1.72e-02	4	4	1.84e-15	1.61e-14	1.59e-02	4	4	1.03e-14	9.47e-14	1.81e-02	4	4	2.55e-14	2.45e-13	4
SE	1.23e-02	4	3	1.29e-15	1.13e-14	1.33e-02	4	3	3.41e-15	3.13e-14	1.25e-02	4	3	1.08e-14	1.04e-13	3
Frank matrix (12×12)																
PSN	2.31e-02	5	4	1.46e-15	1.54e-08	2.39e-02	5	4	7.65e-15	2.51e-08	2.44e-02	5	4	1.17e-13	6.58e-08	4
SN	1.69e-02	6	3	1.46e-15	1.55e-08	1.67e-02	6	3	5.06e-15	1.66e-08	1.70e-02	6	3	1.15e-13	6.45e-08	3
SH	1.77e-02	4	3	1.52e-15	1.61e-08	1.67e-02	4	3	8.09e-15	2.65e-08	1.80e-02	4	3	8.24e-14	4.73e-08	3
SE	1.63e-02	4	3	1.17e-15	1.24e-08	1.80e-02	4	3	7.45e-15	2.44e-08	1.78e-02	4	3	1.07e-13	6.00e-08	3

表 3.5: Results for computing principal p th root of 7×7 Hilbert matrix, 10×10 Prolate matrix and 12×12 Frank matrix by using the four Algorithms.

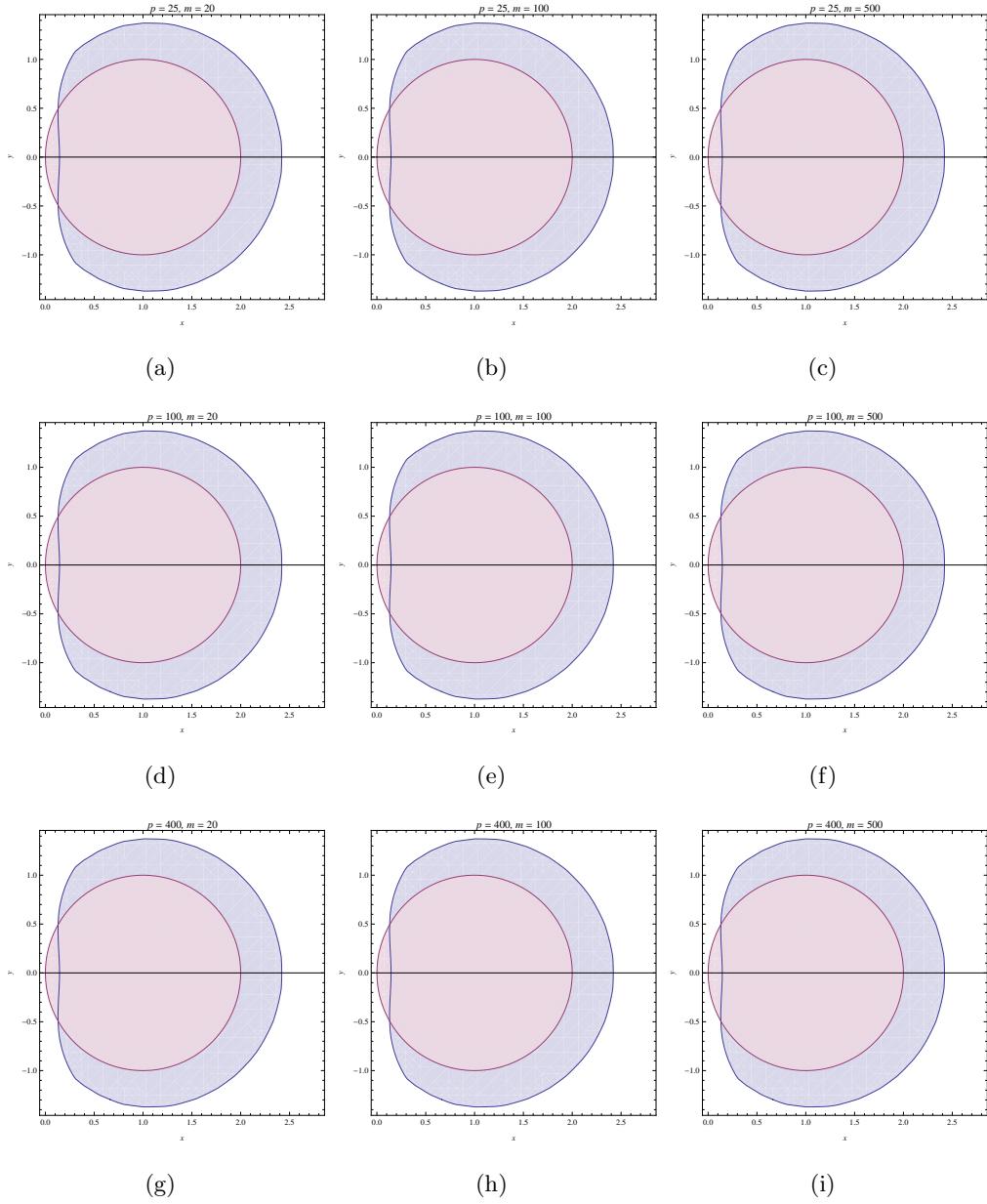


图 3.21: The approximating regions of \mathcal{R}_2 defined in (??) for $p = 25, 100, 400$ and $m = 20, 100, 500$, where the red and blue regions denote the sets $\{z \in \mathbb{C} : 1 - z \in \overline{\mathcal{D}}_1\}$ and $\{z \in \mathbb{C} : 1 - z \in \mathcal{D}_{0,2} \cap \mathcal{D}_{2,2}\}$, respectively.

3.7 关于 Halley 法的注记

注 3.2.

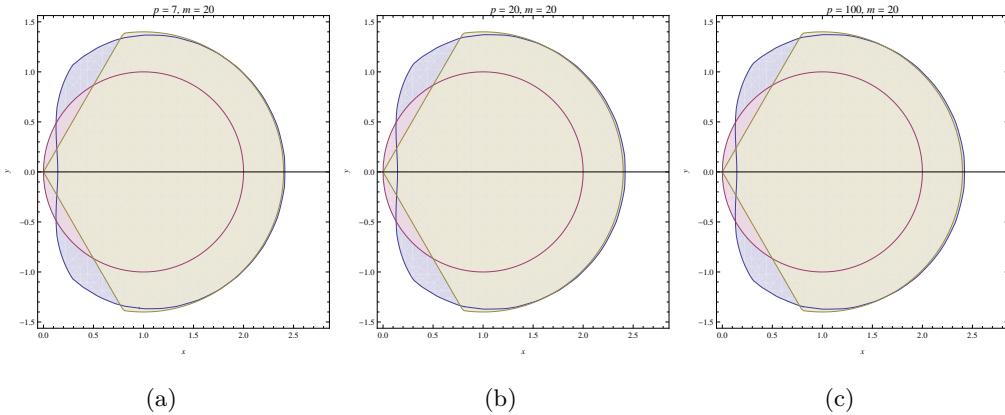


图 3.22: For fixed $m = 20$ and $p = 7, 20, 100$, the actual convergence regions \mathcal{R}_2 defined in (??) (the union of the red and blue parts) and the approximate convergence regions \mathcal{R}_2^H defined in (??) (the yellow parts).

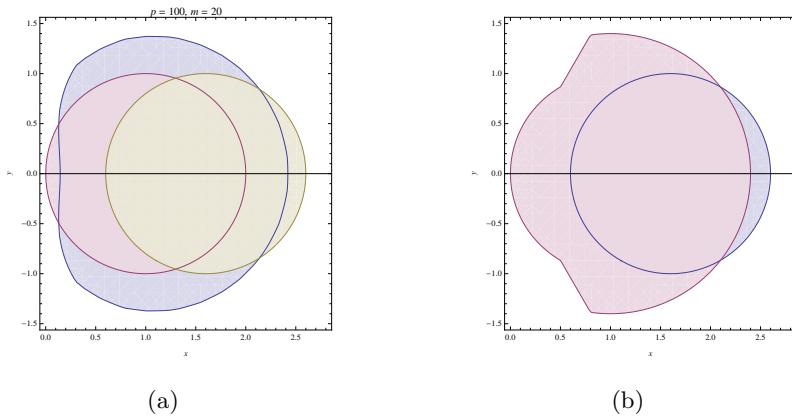


图 3.23: (a) The convergence region \mathcal{R}_2 defined in (??) with $p = 100$ and $m = 20$ (the union of the red and blue parts) and the the disk of center $8/5$ and radius 1 given by Iannazzo (the yellow parts); (b) the feasible region \mathcal{R}_2^H defined in (??) (the red part) and the disk of center $8/5$ and radius 1 (the blue part).

注 3.3.

第4章 移动网格有限元的多重网格预处理方法

原始变量形式的不可压Navier-Stokes方程

$$\begin{aligned}\partial_t \mathbf{u} - \nu \nabla^2 \mathbf{u} + (\mathbf{u} \cdot \nabla) \mathbf{u} + \nabla p &= \mathbf{f}, \\ \nabla \cdot \mathbf{u} &= 0,\end{aligned}\tag{4.1}$$

$\partial\Omega = \partial\Omega_D \cup \partial\Omega_N$ 上的边界条件以及初值条件:

$$\begin{aligned}\mathbf{u} &= \mathbf{w}, && \text{on } \partial\Omega_D \times [0, T] \\ \nu \frac{\partial \mathbf{u}}{\partial n} - p &= \mathbf{0}, && \text{on } \partial\Omega_N \times [0, T], \\ \mathbf{u}|_{t=0} &= \mathbf{u}_0, && \text{in } \Omega.\end{aligned}\tag{4.2}$$

其中 $\Omega \in \mathcal{R}^d, (d = 2, 3)$ 是计算区域, $[0, T]$ 是时间区间, \mathbf{u} 是速度变量并且常量 p 是压力, \mathbf{n} 是边界 $\partial\Omega$ 上的外法向方向, $\nu > 0$ 是粘性动力学系数。我们用[?]和[?]中提出的移动网格有限元方法求解(4.1)和(4.2)。过去几十年, 很多人在移动网格移动网格方法上做出了很多工作([?], [?], [?], [?], [?], [?], [?], [?])。Winslow [?]提出了用移动网格方法求解椭圆方程。作为Winslow的工作的拓展, Dvinsky [?]提出了调和函数理论可以用来产生网格。收到Dvinsky工作的启发, L1, Tang and Zhang [?]提出了基于调和映射的移动网格有限元方法。邸[?]把这种移动网格策略应用到求解原始变量形式的不可压Navier-Stokes方程。文中在移动网格策略中提出了保持散度的插值方法, 这是通过求解一个类似线性的Navier-Stokes方程。我们在上一章中, 提到了我们将 $4P1 - P1$ 元应用到移动网格有限元方法上。 $4P1 - P1$ 元自然满足LBB(inf-sup)条件, 在[?]给出了 $P1isop2P1$ 元的误差收敛阶, 速度 L^2 误差是二阶收敛, 压力是一阶收敛。 $4P1 - P1$ 跟 $P1isoP2P1$ 元具有相同的数据结构, 但是 $4P1 - P1$ 元中的速度单元上的基函数全部位于一个速度单元上, 在速度单元和压力单元进行拼装的时候, 只要建立速度单元和压力单元间的索引, 便是一个简单的过程。

据我们所知, 用满足inf-sup条件的 $4P1 - P1$ 元可以推出一个鞍点问题, 许和何([?], [?], [?])将两格子方法应用到Navier-Stokes方程的求解。很多人在为krylov子空间方法提供预处理子方面做出了很多工作。在[?]文中, 概括了

求解鞍点问题常用的数值方法，例如块预处理和多重网格预处理。文献([?], [?], [?], [?])中提出了许多块预处理方法，来找到鞍点问题的schur补的更好的近似。[?]中提出了求解Ossen系统的增广的基于lagarian方法。在[?]中提出了求解不可压Navier-Stokes方程的维数分解的预处理方法。([?], [?])中提出了求解Navier-Stokes方程的一种高效的Krylov子空间方法，这种方法是用代数多重网格做预处理子。但是据我们所知，对于鞍点问题的高效预处理方法基本上都是基于均匀网格上的，尽管在[?]一文中，考虑了拉伸网格的情形。

在[?]的基础上，我们将为移动网格有限元方法求解(4.1)和(4.2)提供了一个代数多重网格预处理方法。我们通过数值例子来表明，我们的预处理子的高效性。

4.1 时间格式的离散

在时间层离散上，我们把时间区间 $[0, T]$ 分成 N 份， $\{t_i\}_{i=1}^N$ 。令 \mathbf{u}^j 和 p^j 是连续形式解 $\mathbf{u}(\cdot, t_j)$ 和 $p(\cdot, t_j)$ 的离散近似。在时间格式的离散上，主要分为算子分裂方法和全隐格式。在算子分裂方法，也可以看成是分部方法。最简单的方法是将算子分成两步，在每一步中分别用向前和向后欧拉格式，可以参考[?]。在[?]中，将这种两步分裂方法应用到Navier-Stokes方程的时间离散上。通常的两步方法如下：在(??)中 \mathbf{u}^* 的取值要保证散度为0的条件，并且 $\alpha + \beta = 1$ 。

Algorithm 4.1 Peaceman-Rachford 格式

给定 $\mathbf{u}^0, p^0, \theta \in [0, 1], \alpha \in (0, 1), \beta \in (0, 1)$ ，假设 \mathbf{u}^n 已知，通过以下求解 \mathbf{u}^{n+1} ：

$$\frac{\mathbf{u}^{n+\theta} - \mathbf{u}^n}{\theta \delta t} - \alpha \nu \nabla^2 \mathbf{u}^{n+\theta} + \mathbf{u}^* \cdot \nabla \mathbf{u}^{n+\theta} = \beta \nu \nabla^2 \mathbf{u}^n - \nabla p^n \quad \text{在 } \Omega \text{ 内}, \quad (4.3)$$

$$\mathbf{u}^{n+\theta} = \mathbf{g}^{n+\theta} \quad \text{在 } \partial \Omega \text{ 上}:$$

$$\frac{\mathbf{u}^{n+1} - \mathbf{u}^{N+\theta}}{(1-\theta)\delta t} - \beta \nu \nabla^2 \mathbf{u}^{n+1} + \nabla p^{n+1} = \alpha \nu \nabla^2 \mathbf{u}^{n+\theta} - \mathbf{u}^* - \nabla \mathbf{u}^{n+\theta} \quad \text{在 } \Omega \text{ 内}, \quad (4.4)$$

$$\nabla \cdot \mathbf{u}^{n+1} = 0 \quad \text{在 } \Omega \text{ 内},$$

$$\mathbf{u}^{n+1} = \mathbf{g}^{n+1} \quad \text{在 } \partial \Omega.$$

它实际上是先求解一个对流方程(??), 然后再求解一个广义的Stokes方程(??)。这个算法第一步计算的时候需要 p^0 的值, 它可以在很小的时间步长情况下, 全隐格式向前发展一步得到。一般自然的, $\mathbf{u}^* = \mathbf{u}^n$, 这样在时间格式上只有一阶精度的。这种方法不是无条件稳定的, 并且需要比较小的时间步长。当 $\theta = \frac{1}{2}$, $\mathbf{u}^* = \mathbf{u}^{n+\text{theta}}$ 时, 可以获得二阶精度, 这时候求解(??)变成了一个求解非线性的对流方程。详细内容可以参考([?])。[?]将两步Peaceman-Rachford推广到三步, 保持了当 $t \leftarrow \infty$ 时的数值稳定性: 这种算法需要在每个时间步中

Algorithm 4.2 Glowinski Θ 格式

给定 $\mathbf{u}^0, \theta \in [0, 1/2], \alpha \in (0, 1), \beta \in (0, 1)$, 假设 \mathbf{u}^n 已知, 通过以下求解 \mathbf{u}^{n+1} :

$$\begin{aligned} \frac{\mathbf{u}^{n+\theta} - \mathbf{u}^n}{\theta \delta t} - \alpha \nu \nabla^2 \mathbf{u}^{n+\theta} + \nabla p^{n+\theta} &= \beta \nu \nabla^2 \mathbf{u}^n - \mathbf{u}^n \cdot \nabla \mathbf{u}^n \quad \text{在 } \Omega \text{ 内}, \\ \nabla \cdot \mathbf{u}^{n+\theta} &= 0 \quad \text{在 } \Omega \text{ 内}, \\ \mathbf{u}^{n+\theta} &= \mathbf{g}^{n+\theta} \quad \text{在 } \partial \Omega \text{ 上}; \end{aligned} \tag{4.5}$$

$$\begin{aligned} \frac{\mathbf{u}^{n+1-\theta} - \mathbf{u}^{n+\theta}}{(1-2\theta)\delta t} - \beta \nu \nabla^2 \mathbf{u}^{n+1-\theta} + \mathbf{u}^* \cdot \nabla \mathbf{u}^{n+1-\theta} &= \alpha \nu \nabla^2 \mathbf{u}^{n+\theta} - \nabla p^{n+\theta} \cdot \nabla \mathbf{u}^n \quad \text{在 } \Omega \text{ 内}, \\ \mathbf{u}^{n+1-\theta} &= \mathbf{g}^{n+1-\theta} \quad \text{在 } \partial \Omega \text{ 上}; \end{aligned} \tag{4.6}$$

$$\begin{aligned} \frac{\mathbf{u}^{n+1} - \mathbf{u}^{n+1-\theta}}{\theta \delta t} - \alpha \nu \nabla^2 \mathbf{u}^{n+1} + \nabla p^{n+1} &= \beta \nu \nabla^2 \mathbf{u}^{n+1-\theta} - \mathbf{u}^* \cdot \nabla \mathbf{u}^{n+1-\theta} \quad \text{在 } \Omega \text{ 内}, \\ \nabla \cdot \mathbf{u}^{n+1} &= 0 \quad \text{在 } \Omega \text{ 内}, \\ \mathbf{u}^{n+\theta} &= \mathbf{g}^{n+\theta} \quad \text{在 } \partial \Omega \text{ 上}. \end{aligned} \tag{4.7}$$

计算两个广义的Stokes方程(??) 和(??), 和一个非线性的对流方程(??)。当我们取 $\alpha = \beta = 1/2$ 或者 $\theta = 1 - 1/\sqrt{2}$, $\alpha + \beta = 1$, 当 $t \leftarrow \infty$ 时, 该格式时间上有二阶精度。特别的, 当 $\theta = 1 - 1/\sqrt{2}$, $\alpha = (1-2\theta)/(1-\theta)$, $\beta = \theta/(1-\theta)$ 时, 这种方法在时间上是有二阶精度的, 并且是无条件稳定的。([?])中提出了一种线性化 Θ -格式, 它保持了二阶精度。它的想法是将对流方程(??)中的对流项

$$\mathbf{u}^* = \frac{2\theta-1}{\theta} \mathbf{u}^n + \frac{1-\theta}{\theta} \mathbf{u}^{n+\theta}. \tag{4.8}$$

从而使得非线性对流扩散方程变成了线性化的对流扩散方程。我们将会全隐格式的近似方法表述如下：最简单的时间格式是一步有限差分离散，在如何处理非线性对流项上可以归纳为以下算法：其中

Algorithm 4.3 非线性隐式 Θ 格式

给定 $\mathbf{u}^0, \theta \in [0, 1]$, 假设 \mathbf{u}^n 已知, 通过以下求解 \mathbf{u}^{n+1} :

$$\frac{\mathbf{u}^{n+1} - \mathbf{u}^n}{\delta t} - \nu \nabla^2 \mathbf{u}^{n+\theta} + \mathbf{u}^* \cdot \nabla \mathbf{u}^{n+\theta} + \nabla p^{n+\theta} = 0 \quad \text{在}\Omega\text{内}, \quad (4.9)$$

$$\nabla \cdot \mathbf{u}^{n+\theta} = 0 \quad \text{在}\Omega\text{内}, \quad (4.10)$$

$$\mathbf{u}^{n+\theta} = \mathbf{g}^{n+\theta} \quad \text{在}\partial\Omega\text{上}. \quad (4.11)$$

$$\mathbf{u}^{n+\theta} = \theta \mathbf{u}^{n+1} + (1 - \theta) \mathbf{u}^n. \quad (4.12)$$

$$p^{n+\theta} = \theta p^{n+1} - (1 - \theta) p^n. \quad (4.13)$$

$\theta = 1$ 时, 是将非线性项隐式化处理了。当 $\mathbf{u}^{n+\theta} = \mathbf{u}^{n+1}, \mathbf{u}^* = \mathbf{u}^{n+1}$ 格式变成了时间方向一阶精度的向后欧拉格式, 当 $\mathbf{u}^{n+\theta} = \mathbf{u}^{n+1/2}, \mathbf{u}^* = \mathbf{u}^{n+1/2}$ 时, 就是时间方向的二阶精度的Crank-Nicolson格式。但是这两种格式在每个时间步中都都需要求解一个非线性问题, 因此在计算效率上都不是高效的。一种折中的办法是将 $\mathbf{u}^{n+\theta} = \mathbf{u}^{n+1}, \mathbf{u}^* = \mathbf{u}^n, p^{n+\theta} = p^{n+1}$, 这样虽然在时间上损失了一阶精度, 但是时间离散格式还是无条件稳定的, 因此我们本文中的方法就是采用的这种线性化策略。我们来验证线性化向后欧拉方法的稳定性。不失一般性, 我们取 $\mathbf{g}^{n+1} = \mathbf{0}$, 对(??)方程两边同时乘上 \mathbf{u}^{n+1} , 并在 Ω 上做 L_2 内积。我们可以通过选取基函数使得对流项:

$$\int_{\Omega} (\mathbf{u}^n \cdot \mathbf{u}^{n+1}) \cdot \mathbf{u}^{n+1} = 0. \quad (4.14)$$

并且对质量守恒方程(??)关于 p^{n+1} 做 L_2 内积, 再利用分部积分:

$$\int_{\Omega} \nabla p^{n+1} \cdot \mathbf{u}^{n+1} = - \int_{\Omega} p^{n+1} (\nabla \cdot \mathbf{u}^{n+1}) = 0.$$

因此,

$$\frac{1}{\delta t} \int_{\Omega} (\mathbf{u}^{n+1} - \mathbf{u}^n) \mathbf{u}^{n+1} + \nu \int_{\Omega} \nabla \mathbf{u}^{n+1} \cdot \mathbf{u}^{n+1} = 0. \quad (4.15)$$

对(??)在 $[t_n, t_{n+1}]$ 上进行积分, 可得

$$\begin{aligned} \|\boldsymbol{u}^{n+1}\|^2 + \nu \int_{t_n}^{t_{n+1}} \|\nabla \boldsymbol{u}\|^2 &= \int_{\Omega} \boldsymbol{u}^n \boldsymbol{u}^{n+1} \\ &\leq \frac{1}{2} \int_{\Omega} ((\boldsymbol{u}^n)^2 + (\boldsymbol{u}^{n+1})^2) \\ &= \frac{1}{2} \|\boldsymbol{u}^n\|^2 + \frac{1}{2} \|\boldsymbol{u}^{n+1}\|^2. \end{aligned} \quad (4.16)$$

从而我们可以得到

$$\frac{1}{2} \|\boldsymbol{u}^{n+1}\|^2 + \nu \int_{t_n}^{t_{n+1}} \|\boldsymbol{u}^{n+1}\|^2 \leq \frac{1}{2} \|\boldsymbol{u}^n\|^2. \quad (4.17)$$

注意到(??)中的第二项是严格正的, 因此

$$\frac{1}{2} \|\boldsymbol{u}^{n+1}\|^2 < \frac{1}{2} \|\boldsymbol{u}^n\|^2. \quad (4.18)$$

的成立跟时间步长 δt 无关。即线性化向后欧拉格式是无条件稳定的。在这种线性化策略的基础上, 一种保持时间上二阶精度的方法是 $\theta = 1/2$, $\boldsymbol{u}^* = 3/2\boldsymbol{u}^n - 1/2\boldsymbol{u}^{n-1}$, 这是[?]中提出的。我们将这种两步方法表述如下: 其

Algorithm 4.4 Simo-Armero 格式

给定 $\boldsymbol{u}^0 \boldsymbol{u}^1$, 通过以下求解 $\boldsymbol{u}^k, k = 1, \dots, n+1$:

$$\begin{aligned} \frac{\boldsymbol{u}^{n+1} - \boldsymbol{u}^n}{\delta t} - \nu \nabla^2 \boldsymbol{u}^{n+1/2} + \left(\frac{3}{2}\boldsymbol{u}^n - \frac{1}{2}\boldsymbol{u}^{n-1}\right) \cdot \nabla \boldsymbol{u}^{n+1/2} + \nabla p^{n+1/2} &= 0 \quad \text{在 } \Omega \text{ 内}, \\ \nabla \cdot \boldsymbol{u}^{n+1/2} &= 0 \quad \text{在 } \Omega \text{ 内}, \\ \boldsymbol{u}^{n+1/2} &= \boldsymbol{g}^{n+1/2} \quad \text{在 } \partial\Omega \text{ 上}. \end{aligned} \quad (4.19)$$

中 \boldsymbol{u}^1 可以由向后欧拉格式得到, 算法(??)也是无条件稳定的, 证明过程可以根据线性化向后欧拉格式的证明。

根据上一章中, 我们提到的 $4p1 - P1$ 元, 这种有限元是基于两套三角形网格和两个有限元空间。根据[?]中的几何遗传树中的四叉树结构, 速度网格可以通过压力网格全局加密一次得到, 如图??所示。速度单元和压力单元间的一一对应关系可以根据几何遗传树结构, 相对容易的获得。我

们先给出一些符号的定义: \mathcal{T}_h 是速度网格的一个三角剖分, 最大的网格尺度 $h = \max_{T \in \mathcal{T}_h} \text{diam}(T)$, 同样的, $\mathcal{T}_H (H = 2h)$ 是压力网格的三角剖分。速度和压力的有限维空间分别为 $X_h \subset (H_0^1(\Omega))^2$ 和 $P_H \subset \mathcal{L}^2(\Omega)$ 。那么 Navier-Stokes 问题的全离散形式为:

给定 t_n 时刻的 (\mathbf{u}_h^n, p_H^n) , 来计算 $(\mathbf{u}_h^{n+1}, p_H^{n+1})$ 通过

$$\begin{aligned} \frac{1}{dt}(\mathbf{u}_h^{n+1}, \mathbf{v}_h) + \nu(\nabla \mathbf{u}_h^{n+1}, \nabla \mathbf{v}_h) + (\mathbf{u}_h^n \cdot \nabla \mathbf{u}_h^{n+1}, \mathbf{v}_h) - (p_H^{n+1}, \nabla \mathbf{v}_h) &= \frac{1}{dt}(\mathbf{u}_h^n, \mathbf{v}) \\ (\nabla \cdot \mathbf{u}_h^{n+1}, q_H) &= 0. \end{aligned} \quad (4.20)$$

对所有的 $(\mathbf{v}_h, q_H) \in \mathcal{X}_h \times P_H$.

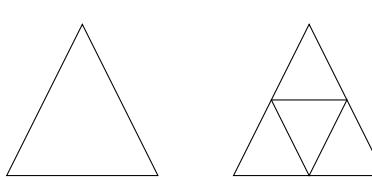


图 4.1: 全局加密一次的网格

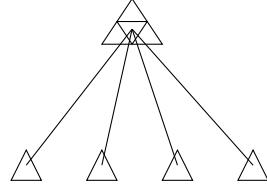


图 4.2: 它的几何遗传树

图 4.3: 几何遗传树结构

4.2 快速Krylov求解

令 $(\{\phi_j\}_{j=1}^n, 0)^T$ 和 $(0, \{\phi_j\}_{j=1}^n)^T$ 是速度空间 X_h 的线性元基函数。同时, $\{\psi_k\}_{k=1}^m$ 为压力空间 P_H 的线性元基函数。 t_{n+1} 时刻, 速度解的分量形式 $\mathbf{u}_h^{n+1} = (u_h^{x,n+1}, u_h^{y,n+1})^T$, 压力数值解为 \mathbf{p}_H^{n+1} at $t = t_{n+1}$, 可以写为:

$$u_h^{x,n+1} = \sum_{j=1}^{n_u} \alpha_j^{x,n+1} \phi_j, \quad u_h^{y,n+1} = \sum_{j=1}^{n_u} \alpha_j^{y,n+1} \phi_j, \quad p_H^{n+1} = \sum_{k=1}^{n_p} \alpha_k^{p,n+1} \psi_k. \quad (4.21)$$

将(??) 带入弱形式(??)中, 可以得到一个鞍点问题:

$$\begin{bmatrix} \frac{1}{dt}M + \nu A + N & 0 & B_x^T \\ 0 & \frac{1}{dt}M + \nu A + N & B_y^T \\ B_x & B_y & 0 \end{bmatrix} \begin{bmatrix} \alpha^{x,n+1} \\ \alpha^{y,n+1} \\ \alpha^{p,n+1} \end{bmatrix} = \begin{bmatrix} f_x \\ f_y \\ 0 \end{bmatrix}, \quad (4.22)$$

注意到散度矩阵 $B = [Bx, By]$ 为

$$B_x := [B_x]_{kj} = -\left(\psi_k, \frac{\partial \phi_j}{\partial x}\right), k = 1, \dots, n_p, j = 1, \dots, n_u, \quad (4.23)$$

$$B_y := [B_y]_{kj} = -\left(\psi_k, \frac{\partial \phi_j}{\partial y}\right), k = 1, \dots, n_p, j = 1, \dots, n_u. \quad (4.24)$$

因为速度单元上的基函数和压力单元上的基函数并不在同一张网格上，所以装配矩阵 B 并不是一个显然的过程。这需要根据上面提到的速度单元和压力单元间的 $1 - 1$ 索引，我们可以仅仅只用速度和压力单元上的线性元函数来装配 B ，拼装 B^T 的过程类似。我们定义 $F_\nu^{n+1} = \frac{1}{dt}M + \nu A + N$ ，其中

$$M := [M]_{ij} = (\phi_i, \phi_j), \quad i, j = 1, \dots, n_u, \quad (4.25)$$

$$A := [A]_{ij} = (\nabla \phi_i, \nabla \phi_j), \quad i, j = 1, \dots, n_u, \quad (4.26)$$

$$N := [N]_{ij} = (\mathbf{u}_h^n, \nabla \phi_i, \phi_j), \quad i, j = 1, \dots, n_u. \quad (4.27)$$

为了高效的求解线性方程组(3.18)，我们用预处理的GMRES(generalized minimal residual algorithm)作为求解器。在[?] 中考虑的块三角预处理 \mathcal{P} 定义如下：

$$\mathcal{P} = \begin{pmatrix} F & 0 & B_x^T \\ 0 & F & B_y^T \\ 0 & 0 & S \end{pmatrix} \quad (4.28)$$

其中 $S = B_x F^{-1} B_x^T + B_y F^{-1} B_y^T$ 是 Schur 补矩阵。预处理就是要完成近似矩阵的求逆，实施 \mathcal{P}^{-1} 的过程分为两步，第一步是求解 schur 补的系统，第二步是求解两个纯量的跟 F 有关的系统。直接求解 schur 补的方程组是非常耗时的，因为它系数矩阵里包含一个逆矩阵。所以，在实际的计算中，找到 schur 补的近似矩阵，然后用近似矩阵代替 schur 补矩阵来进行求解。[?] 一文中提到 PCD 预处理可以用来作为 schur 补的近似。PCD 矩阵定义为：

$$S_* = A_p F_p^{-1} Q_p. \quad (4.29)$$

其中 A_p, F_p 和 Q_p 全部定义在压力空间上。 Q_p 是压力质量矩阵， A_p 是压力刚度矩阵， F_p 是对流扩散矩阵，它们分别定义为：

$$F_p := [F_p]_{ij} = \nu(\nabla \psi_i, \nabla \psi_j) + (\mathbf{u}_h^n \cdot \nabla \psi_i, \psi_j), \quad i, j = 1, \dots, n_p, \quad (4.30)$$

$$A_p := [A_p]_{ij} = (\nabla \psi_i, \nabla \psi_j) \quad i, j = 1, \dots, n_p. \quad (4.31)$$

令

$$W_p^n := [W_p^n]_{ij} = (\mathbf{u}_h^n \cdot \nabla \psi_i, \psi_j), \quad i, j = 1, \dots, n_p. \quad (4.32)$$

那么 F_p 可以被重新写成:

$$F_p = \nu A_p + W_p^n. \quad (4.33)$$

我们实施PCD预处理通过

$$S_*^{-1} \approx Q_p^{-1} F_p A_p^{-1}. \quad (4.34)$$

精确的PCD 预处理算子定义为

$$\mathcal{M}^{-1} = \begin{pmatrix} F^{-1} & 0 & B_x^T \\ 0 & F^{-1} & B_y^T \\ 0 & 0 & S_*^{-1} \end{pmatrix} \quad (4.35)$$

我们分两步解释预处理的求解过程: 令 $V^d = \mathcal{M}^{-1}V^s$, 其中

$$V^d = (V_x^d, V_y^d, V_p^d)^T, V^s = (V_x^s, V_y^s, V_p^s)^T. \quad (4.36)$$

第一步我们求解:

$$V_p^d = S_*^{-1}V_p^s = Q_p^{-1}F_p A_p^{-1}V_p^s. \quad (4.37)$$

它其中包含两个posision问题的求解 Q_p^{-1} 和 A_p^{-1} , 所以我们可以用为求解posision问题设计的多重网格求解器来实现求解。第二步我们再用这种多重网格求解器去求解

$$\begin{aligned} V_x^d &= F^{-1}(V_x^s - B_x^T V_p^d), \\ V_y^d &= F^{-1}(V_y^s - B_y^T V_p^d). \end{aligned} \quad (4.38)$$

将(??) 和(??)中的解合并在一起, 就得到了要求解的 V^d .

在实际的计算中, 我们对矩阵 F , F_p , Q_p , 和 A_p 的求解, 是用固定的几步代数多重网格迭代(algebraic multi-grid)来代替精确求解。这就是迭代的PCD方法。我们用的多重网格求解器是在AFEPack中(一个自适应有限元包), 可以从<http://dsec.pku.edu.cn/~rli>获得。在([?], 第10章)中, 数值算例说明了PCD预处理的高效。[?] 一文将PCD预处理方法应用到bouyancy驱动流问题上。在我们的数值实验中, 发现如果将(??)中的

$$F_p = \nu A_p + W_p^n. \quad (4.39)$$

中 A_p 前面的系数去掉，会发现GMRES的迭代步数会减少。我们采用([?])中在Neumann边界上的处理矩阵 F_p 和 A_p 的方法，来提高求解效率。 F_p 需要在边界 $\partial\Omega = \partial\Omega_D \cup \partial\Omega_N$ 上满足边界条件

$$\nu \frac{\partial p_h}{\partial n} + (\mathbf{w}_h \cdot \mathbf{n}) p_h = 0. \quad (4.40)$$

我们知道对于方腔流，在所有的边界上 $\mathbf{w}_h \cdot \mathbf{n} = 0$ ，所以(??)将会退化成

$$\frac{\partial p_h}{\partial n} = 0. \quad (4.41)$$

这就意味着我们在所有的边界 $\partial\Omega$ 上，对 F_p 不采取任何操作， A_p 采取同样的操作。在本文中，我们将PCD预处理方法应用到移动网格有限元方法，来高效地求解Navier-Stokes问题(3.18)。

4.3 移动网格策略

4.3.1 网格移动的流程

再上一章中我们详细的介绍过移动网格的策略，这一章中简要描述一下。在 $t = t_n$ 时刻，我们获得了数值解 $\mathbf{u}_h^{(n)}, p_H^{(n)}$ 在 t_n 时刻的网格 \mathcal{T}_h^n 上。我们根据[?]中的方法，用保持散度为0的差值方法将 \mathcal{T}_h^n 上的数值解差值到 t_{n+1} 时刻的网格 $\mathcal{T}_h^{(n+1)}$ 上。简单的来说，总共分为三步：

step 1 获取控制函数。令 $m = 1/G$ ，其中 G 是控制函数。基于涡量的控制函数

$$G = \sqrt{1 + \alpha|\omega|^\beta}. \quad (4.42)$$

其中 $\omega = \nabla \times \mathbf{u}$ ， α, β 是两个正的常数。在本文中， $\beta = 2$ 有比较的效果， α 根据不同的问题，选取不同的值。problems.

step 2 获取物理网格上的移动方向。求解

$$\begin{aligned} \nabla_x(m \nabla_x \boldsymbol{\xi}) &= 0, \\ \boldsymbol{\xi}|_{\partial\Omega} &= \boldsymbol{\xi}_b. \end{aligned} \quad (4.43)$$

来获得一个新的逻辑网格 \mathcal{T}_c^* ， \mathcal{A}^* 作为它的节点。我们可以得到新的逻辑网格和初始逻辑网格 \mathcal{T}_c^0 (节点为 \mathcal{A}^0)之间的误差

$$\delta\mathcal{A} = \mathcal{A}^0 - \mathcal{A}^*. \quad (4.44)$$

我们可以根据 δA 来获得物理区域的位移 δX_i , 通常再乘上一个正的常数 μ 避免网格缠接:

$$X_i^{(n+1)} = X_i^{(n)} + \mu \delta X_i. \quad (4.45)$$

step 3 保持散度为0的差值。在用移动网格有限元方法来求解不可压流体方程的时候, 需要在数值解差值的过程中保持散度为0。在[?]中, 数值解在新网格 $\mathcal{T}^{(n+1)}$ 上的重新分布, 是通过求解一个类似无粘的Navier-Stokes方程。

$$\begin{aligned} \frac{\partial \mathbf{u}}{\partial \tau} - \nabla_{\mathbf{x}} \mathbf{u} \cdot \delta \mathbf{x} &= -\nabla \hat{p}, \\ \nabla_{\mathbf{x}} \cdot \mathbf{u} &= 0. \end{aligned} \quad (4.46)$$

其中 $\delta \mathbf{x} := x^{\text{old}} - x^{\text{new}}$, $x^{\text{old}}, x^{\text{new}}$ 是物理区域上的两组坐标。 τ 是一个虚拟时间, 通常取成1.0。因为对流速度 $\delta \mathbf{x}$ 相对较小。这里 \hat{p} 是一个临时变量, 为了跟(4.1)中的压力变量区别开来。

(3.38)的弱形式是: 寻找 $(\mathbf{u}_h, \hat{p}_H) \in X_E^h \times P^H$ 使得

$$\begin{aligned} (\partial_{\tau} \mathbf{u}_h - \nabla_{\mathbf{x}} \mathbf{u}_h \cdot \delta \mathbf{x}, \mathbf{v}_h) &= (\hat{p}_H, \nabla \mathbf{v}_h), \quad \forall \mathbf{v}_h \in X_E^h, \\ (\nabla_{\mathbf{x}} \cdot \mathbf{u}, q_H) &= 0, \quad \forall q_H \in P^H. \end{aligned} \quad (4.47)$$

在本文中, 我们用时间层上用显示格式离散: (??) for time discretization:

$$\begin{aligned} \left(\frac{\mathbf{u}_{h,*}^{(n)} - \mathbf{u}_h^{(n)}}{\delta t}, \mathbf{v}_h \right) + \left(\delta \mathbf{x} \cdot \nabla \mathbf{u}_h^{(n)}, \mathbf{v}_h \right) &= \left(\hat{p}_{H,*}^{(n)}, \nabla \mathbf{v}_h \right), \quad \forall \mathbf{v}_h \in X_E^h. \\ (\nabla \cdot \mathbf{u}_{h,*}^n, q_H) &= 0, \quad \forall q_H \in P^H. \end{aligned} \quad (4.48)$$

其中 $\mathbf{u}_h^{(n)}$ 和 $p_H^{(n)}$ 是方程(4.1)在 $t = t_n$ 时刻的网格上计算的数值解。 $\mathbf{u}_{h,*}^{(n)}$ 和 $p_{h,*}^{(n)}$ 是在新网格 $\mathcal{T}^{(n+1)}$ 上, t_n 时刻的数值解, 是一个中间变量。

4.3.2 用AMG 预处理求解(??)

(??) 将会得到一个线性系统, 它的系数矩阵为 \mathcal{M}^p , 定义如下:

$$\mathcal{M}^p = \begin{pmatrix} \frac{1}{\delta t} Q_p & 0 & B_x^T \\ 0 & \frac{1}{\delta t} Q_p & B_y^T \\ B_x & B_y & 0 \end{pmatrix} \quad (4.49)$$

据我们所知，矩阵 \mathcal{M}^p 的schur补矩阵为：

$$M_S = B_x Q_p^{-1} B_x^T + B_y Q_p^{-1} B_y^T. \quad (4.50)$$

根据([?], 第5章)，对于满足LBB条件的混合元近似，当边界条件全部是封闭流体边界时， M_S 是跟压力空间的刚度矩阵 A_p 谱等价的。因此我们可以用 A_p 来作为schur补矩阵的近似矩阵。那么我们可以选取

$$\mathcal{K} = \begin{pmatrix} Q_p & 0 & B_x^T \\ 0 & Q_p & B_y^T \\ 0 & 0 & M_S^* \end{pmatrix} \quad (4.51)$$

来作为(??)的块预处理矩阵，其中 $M_S^* = A_p$ 或者 $M_S^* = \frac{1}{\nu} A_p$ 。我们选择不同的 M_S^* 来作为schur补矩阵的近似矩阵，来对比求解效率的差别。在我们的实际计算中， $\frac{1}{\nu} A_p$ 要比 A_p 效率要高。对于入流/出流问题， A_p 需要在Neumann边界上做出一些修改来提高计算效率。在出流边界 $\partial\Omega_N$ 上，离散的压力 p_h 需要满足一个齐次的Dirichlet边界条件。而在 Ω_D 上，压力 p_h 要满足Neumann条件 $\frac{\partial p_h}{\partial n} = 0$ ，那就意味着我们在Dirichlet边界上，对 A_p 不采取任何操作。细节可以参考[?]。注意到，一旦网格发生移动，所有的矩阵 $M, B_x^T, B_y^T, B_x, B_y, A_p$ 都需要重新构建。

在我们的算法中，PCD预处理的GMRES作为求解线性系统(3.18)的求解器。我们定义GMRES收敛的停止准则为

$$\|r^{(k)}\| \leq 10^{-6} \|r^{(0)}\| \quad (4.52)$$

其中 $r^{(k)}$ 是线性系统(3.18)的残差， r^0 是(3.18)的右端项的残差。最后，为了清楚地表述我们的算法，我们给出算法的流程图如下：

4.4 数值算例

我们用三个数值算例来展示我们的方法。在实际的计算过程中，我们采用稳态的Stokes方程的解作为Navier-Stokes方程的初始值，边界条件设置跟Navier-Stokes方程一致。在我们的算法中，初始的物理区域和逻辑区域是一致的。网格的移动效果以及数值解在下面展示出来。我们的代码是基于有限元包AEPack。

Algorithm 4.5 移动网格有限元方法求解Navier-Stokes

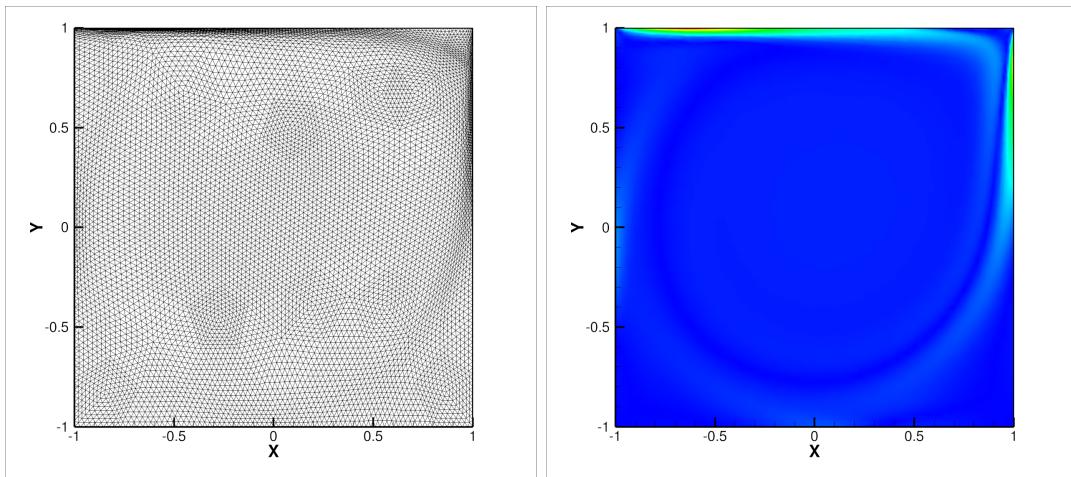
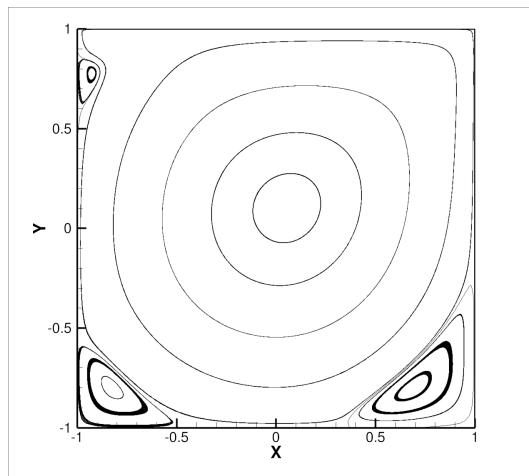
-
- 1: 用AMG预处理来求解稳态的Stokes方程来获得初值值 $\mathbf{u}_h^{(0)}, p_H^{(0)}$.
 - 2: **while** $t_n < T$ **do**
 - 3: 在 $\Delta_p^{(n)}$ 上, 用 $\mathbf{u}_h^{(n)}, p_H^{(n)}$ 计算控制函数, 并且通过求解调和映射(3.25)来获得新的逻辑网格 ξ^* 。
 - 4: 判断如果 $\xi^* - \xi^{(0)}$ 的 L_2 范数是不是比容忍量小。若是, 则迭代结束, 否则继续5 - 8。
 - 5: 用 $\xi^* - \xi^{(0)}$ 之间的误差来计算压力网格 $\Delta_p^{(n)}$ 在物理区域上的移动量 $\delta\mathbf{x}$ 。
 - 6: 在速度网格 $\Delta_v^{(n)}$ 上, 用AMG预处理方法求解方程(??) 来获取中间变量 $\mathbf{u}_{h,*}^{(n)}, p_{H,*}^{(n)}$ 。
 - 7: 更新压力网格 $\Delta_p^{(n)}$ 到 $\Delta_p^{(n+1)}$ 并且利用几何遗传树结构来同步 $\Delta_v^{(n)}$ 到 $\Delta_v^{(n+1)}$ 。
 - 8: 回到3。
 - 9: 用AMG预处理方法来求解Navier-Stokes 问题(3.18), 从而获得新网格 $\Delta_v^{(n+1)}$ 和 $\Delta_p^{(n+1)}$ 上的数值解 $\mathbf{u}_h^{(n+1)}, p_H^{(n+1)}$ 。
 - 10: **end while**
-

4.4.1 方腔驱动流

我们考虑经典算例: 正则化的方腔流。我们的计算区域是 $\Omega = [-1, 1] \times [-1, 1]$, 粘性系数是 $\nu = 0.001$ 。Dirichlet 边界条件设置在所有的边界 $\partial\Omega$ 上。在顶部边上, 速度 $\mathbf{u} = (1 - x^4, 0)^T$, 无滑移条件设置在 $\partial\Omega$ 的其它部分。

在我们的移动策略中, 我们选取涡量(3.20)为控制函数。参数取成 $\alpha = 0.5, \beta = 2.0$ 时网格移动效果表现好。在图??中展示的是发展到稳态的移动网格以及涡量的等高线。我们可以看出网格集中在顶部边和右边边界, 这些地方都是涡量比较大的地方。速度的流速线如图??。

从表??中, 我们得知在PCD预处理中, 选取 $F_p = A_p + W_p^n$ 比 $F_p = \nu A_p + W_p^n$ 需要的迭代步数要少很多。在每个时间步, 求解线性方程组的(3.18)的GMRES迭代步数如图??。它需要 $12 - 24$ 步迭代收敛, 当流体趋于稳定的时候, 迭代次数在 $15 - 16$ 步。

图 4.4: Cavity flow, left: mesh, right: vorticity contour, pressure mesh 40×40 , $\nu = 0.001$.图 4.5: Cavity flow: velocity streamline, pressure mesh 40×40 , $\nu = 0.001$.

pressure mesh	time step	GMRES step number	
		$F_p = \nu A_p + W_p^n$	$F_p = A_p + W_p^n$
20×20	0.00656	41	8
40×40	0.00312	43	12
80×80	0.00153	48	18

表 4.1: Cavity flow: GMRES step number of solving linear system (3.18) with different F_p in PCD preconditioning at first time step, $\nu = 0.001$.

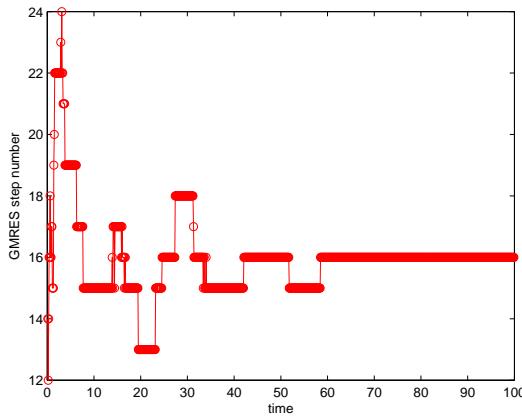


图 4.6: Cavity flow: GMRES step counts of solving (3.18) with modified PCD preconditioner, pressure mesh 40×40 , $\nu = 0.001$.

4.4.2 Backward step flow

这个算例模拟的是流体经过一个向后的台阶。管道的长度是 $l = 5$ ，在入流边界 $x = -1, y \in (0, -1)$ 设置的是 Poiseuille 流条件 $\mathbf{u} = (1 - y^2, 0)^T$ 。在管道的顶端和底端设置的无滑移边界条件 $\mathbf{u} = (0, 0)^T$ ，自然条件设置在出流边界 $x = 5, y \in (-1, 1)$ 上。我们选取粘性系数为 $\nu = 0.02$ ，那么当 $t \rightarrow \infty$ 时，流体趋向于稳态。

我们选取(3.20)为控制函数，控制函数中的参数是 $\alpha = 2.0, \beta = 2.0$ 。我们知道奇异性将会出现在流体扩展的拐角处，因此这里需要更多的网格。在图??中，网格集中在凹进去的拐角处，这跟我们的设想一致。

要满足 CFL 条件，我们的计算时间步长在 0.008 左右。求解(3.18) 和(??)的 GMRES 迭代步数在图?? 中给出。从中可以发现求解(3.18) 需要少于 21 步迭代就能收敛。

4.4.3 Flow over cylinder

这个算例模拟的是流体在长方形的管道流中，流过一个圆形的障碍物。这个问题在[?] 中考虑过，是以流函数形式计算的。圆形障碍的圆心是 $(0, 0)$ ，半径是 $r = 0.3$ ，雷诺数是 $Re = 600$ ，计算区域是 $\Omega = [-1, 5] \times [-1, 1]$ 。在入流边界 $x = -1$ ，设置具有 poiseuille 性质的边界条件 $\mathbf{u} = (1 - y^2, 0)^T$ 。在管道的上边界和下边界，设定条件 $\mathbf{u} = (0, 0)^T$ 。自然条件设置在出流边界 $x = 5$ 上。

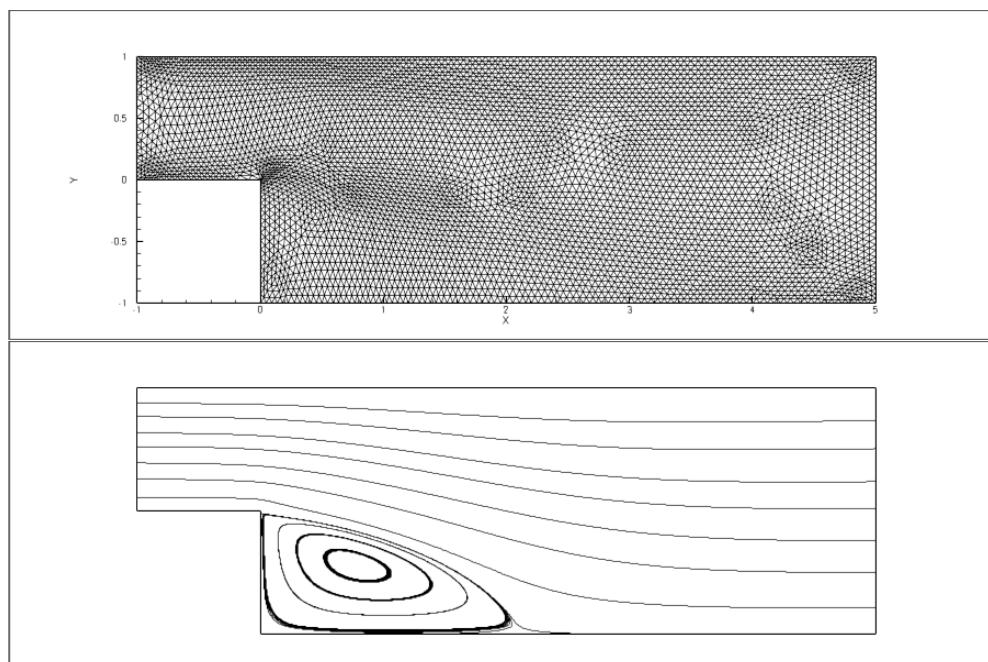


图 4.7: Top: moving mesh, bottom: velocity streamline in step flow at $t = 100s$, viscosity $\nu = 0.02$.

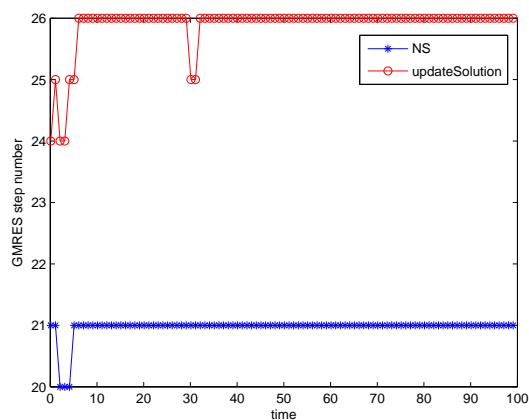


图 4.8: Step flow: GMRES iteration counts via time, $\nu = 0.02$

在我们的移动策略中，用户可以设置控制函数(3.20)的参数 α 和 β 。 α 的值越大，网格聚集的程度就越大。从图??中，我们可以看出 $\alpha = 5, \beta = 2$ 时GMRES的迭代步数要比 $\alpha = 1, \beta = 2$ 时的迭代次数多。当我们选择不同的PCD不同的 F_p ，GMRES迭代次数的比较在图??中给出。我们发现选择 $F_p = A + W_p^n$ 的计算效率要高于 $F_p = \nu A + W_p^n$ 。

在图??中，展示了 $t = 2s$ 时的网格移动效果。可以发现，网格明显的聚集在圆形障碍物的周围。据我们所知，当选取适当的雷诺数时，会出现涡街现象，参考[?]一文，正如图Figure ??所展示的。

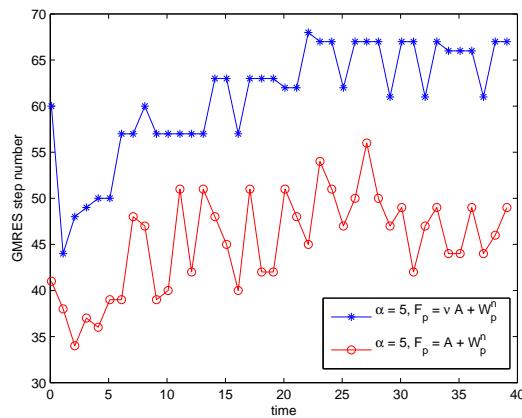


图 4.9: Flow over cylinder: GMRES iteration counts of solving(3.18) with different F_p in preconditioning, $\alpha = 5.0, \nu = 1/300$.

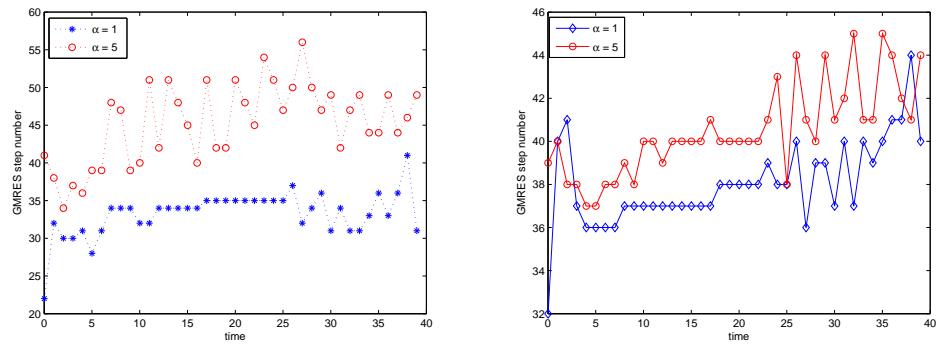


图 4.10: Flow over cylinder, left: GMRES iteration counts of solving (3.18), right: GMRES iteration counts of solving (??), $\nu = 1/300$.

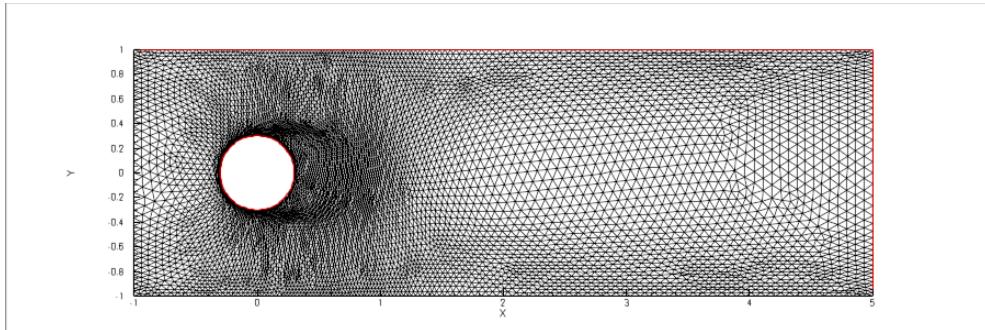


图 4.11: Flow over cylinder: moving mesh at $t = 2s$, viscosity $\nu = 1/300$

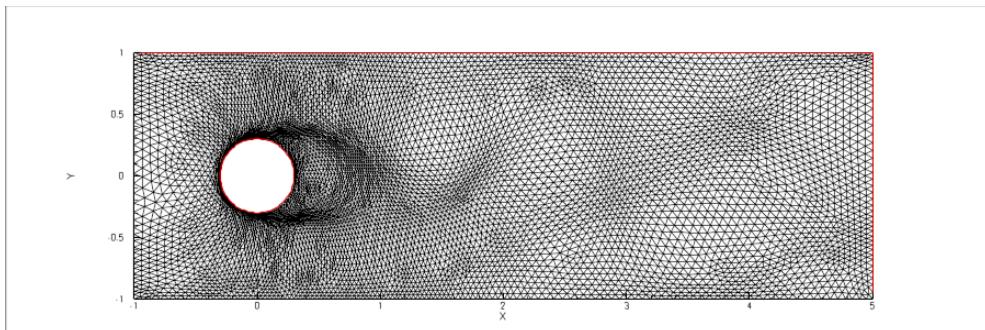


图 4.12: Flow over cylinder: moving mesh at $t = 40s$, viscosity $\nu = 1/300$.

攻读学位期间取得的研究成果

- [1] An analysis on efficiency of Euler's method for computing the matrix pth root, 2013, in preparation.
- [2] A note on Newton's method and Halley's method for computing the matrix pth root, 2013, in preparation.
- [3] On the semilocal convergence behavior for Halley's method, Comput. Optim. Appl., 2014.
- [4] A globally convergent inexact Newton-like Cayley transform method for inverse eigenvalue problems, J. Appl. Math., 2013.
- [5] Convergence behavior for Newton-Steffensen's method under gamma-condition of second derivative, Abstr. Appl. Anal., 2013.

致 谢

本文是在黄正达教授的悉心指导下完成的。非常感谢黄老师给予我自由选择研究课题的余地，使我能够有机会在一个非常有意思的领域内做一些研究工作。黄老师渊博的知识、严谨的治学态度使我受益匪浅，同时在思想、生活上对我也很关心和照顾，值此毕业之际，谨向他致以崇高的敬意和诚挚的谢意。

借此机会，我还要感谢我的同门师兄姐弟妹们，三年来在讨论班及实验室里的交流使我受益匪浅，他们是：孔祥银、高芹、孔维镇、周小燕、王会迪、郑璇、陈敏红、黄敏、郭明、王湘美、王玉芳、郑聪、赵晓梵、潜陈印、张燕、张勇。

感谢所有关心我、支持我、帮助我的亲人和朋友们！

致 谢

浙江大学学位论文独创性声明

本人声明所呈交的学位论文是我个人在导师指导下进行的研究工作及取得的研究成果。论文中除了特别加以标注和致谢的地方外，不包含其他人或其他机构已经发表或撰写过的研究成果。其他同志对本研究的启发和所做的贡献均已在论文中作了明确的声明并表示了谢意。本人完全意识到本声明的法律结果由本人承担。

作者签名： 日期： 年 月 日

学位论文使用授权声明

本人完全了解浙江大学有关保留、使用学位论文的规定，即：学校有权保留并向国家有关机关或机构送交论文的复印件和电子文档，允许论文被查阅和借阅，可以采用影印、缩印或扫描等手段保存、汇编学位论文。同意浙江大学可以用不同方式在不同媒体上发表、传播论文的全部或部分内容。

保密的学位论文在解密后遵守此协议。

作者签名: 导师签名: 日期: 年 月
日