实验报告

廖钰蕾　201928000206026

中国科学院数学与系统科学研究院计算数学与科学工程计算研究所
电子邮箱: liaoyulei@lsec.cc.ac.cn

1. 问题描述

我们希望利用有限元结合多重网格法求解平面弹性模型, 该模型可以表示为二阶椭圆方程:

$$\begin{cases} -(\mu\Delta u + (\lambda+\mu)\nabla\nabla\cdot u) = f, & x \in \Omega, \\ u = 0, & x \in \partial\Omega. \end{cases}$$

其中 $u : \mathbb{R}^2 \mapsto \mathbb{R}^2$ 为待求位移向量, $\lambda \in (0,\infty), \mu \in [\mu_1, \mu_2]$ 是 Lamé 常数, $\Omega \subset \mathbb{R}^2$ 为二维欧式空间中的有界开集, $\partial\Omega$ 为 Ω 的边界, n 为 Ω 的外法向量.

我们采用 P_1 协调有限元, W-cycle 几何多重网格法求解该问题, 并分析磨光算法与 Lamé 常数对迭代次数的影响.

2. 算法描述

2.1. 变分形式.

与有限差分方法等直接离散偏微分方程的方法不同的是, 有限元方法离散的是偏微分方程对应的变分问题, 因此我们首先需要得到上述问题的变分形式.

命题 2.1. *上述问题对应的变分问题为, 找到 $u \in [H_0^1(\Omega)]^2$, 使得对于任意 $v \in [H_0^1(\Omega)]^2$,*

$$a(u,v) = (f,v),$$

其中

$$a(u,v) = 2\mu(\epsilon(u),\epsilon(v)) + \lambda(\nabla\cdot u, \nabla\cdot v),$$

时间: 2020 年 12 月 25 日.

应变张量 $\epsilon(v) := (\nabla v + (\nabla v)^T)/2$, 张量内积

$$(A, B) := \int_\Omega \sum_{i,j=1}^{2} A_{ij} B_{ij} \mathrm{d}x.$$

证明. 定义应力张量

$$\sigma(v) := 2\mu\epsilon(v) + \lambda \mathrm{tr}(\epsilon(v)) I,$$

其中 I 是二阶单位张量, 由于 $\mathrm{tr}(\epsilon(v)) = \nabla \cdot v$, 故

$$a(u, v) = (\sigma(u), \epsilon(v)),$$

由于 $\sigma(u)$ 为对称张量,

$$(\sigma(u), \epsilon(v)) = \frac{1}{2}(\sigma(u), \nabla v + (\nabla v)^T) = (\sigma(u), \nabla v),$$

再由分部积分以及 $v \in [H_0^1(\Omega)]^2$,

$$(\sigma(u), \nabla v) = -(\nabla \cdot \sigma(u), v) = -(\mu \Delta u + (\lambda + \mu) \nabla \nabla \cdot u, v),$$

得证. □

2.2. 有限元方法.

有限元方法基本思想是将区域 Ω 做网格剖分, 在剖分出的每个小区域上用简单的多项式函数逼近方程的解, 之后将这些多项式拼接在一起构成整个区域 Ω 上的数值解. 基本步骤如下：

(1) 将区域 Ω 做三角形网格剖分, 每个小三角形区域称为一个有限元;
(2) 在每个有限元上构建有限元空间, 写出该函数空间的基函数;
(3) 根据变分问题及有限元空间结构计算单元刚度矩阵和单元负载向量;
(4) 将单元刚度矩阵拼入总刚度矩阵 A 中, 将单元负载向量拼入总负载向量 b 中, 得到形如 $Ac = b$ 的待求代数方程, 方程维度为总自由度个数;
(5) 边界上自由度值均为已知, 分离出区域内自由度满足的代数方程 $A(\text{free}, \text{free})c(\text{free}) = b(\text{free})$, 求解 $c(\text{free})$, 其中 free 为区域内自由度的标号;
(6) 将 c 带回到每个有限元空间中, 得到数值解.

我们使用三元组 (T, P_T, Σ_T) 描述每个有限元, T 表示顶点坐标为 (x_1, y_1), (x_2, y_2), (x_3, y_3) 的三角形. P_T 表示有限元上定义的函数空间,

Σ_T 表示函数空间的自由度, 则 T 中的点 (x, y) 可以映射到重心坐标 $(\lambda_1, \lambda_2, \lambda_3)$:

$$\lambda_i = \frac{1}{2|T|}(\eta_i x - \xi_i y + \omega_i), \quad i = 1, 2, 3,$$

其中 $|T|$ 为 T 的面积,

$$\xi_i = x_j - x_k, \quad \eta_i = y_j - y_k, \quad \omega_i = x_j y_k - x_k y_j,$$

顶点坐标 $(x_i, y_i), (x_j, y_j), (x_k, y_k)$ 按逆时针排列.

根据自由度定义 Σ_T 计算出对应的基函数 $\{\phi_i\}_{i=1}^M$, 满足 ϕ_i 在第 i 个自由度值为 1, 其余自由度值为 0. 我们用 $a_T(u_h, v_h)$ 表示将 $a(u, v)$ 的积分域 Ω 替换为 T, u_h, v_h 是在有限元函数空间上对 u, v 的近似, 用 $(\cdot, \cdot)_T$ 表示 T 上的 L^2 内积. 令 $u_h = \Psi c$, $v_h = \Psi d$, 由于 u, v 是二维向量函数, 故

$$c = (c_1^{(1)}, c_1^{(2)}, \ldots, c_M^{(1)}, c_M^{(2)})^T, \quad d = \left(d_1^{(1)}, d_1^{(2)}, \ldots, d_M^{(1)}, d_M^{(2)}\right)^T,$$

$$\psi = \begin{pmatrix} \psi_1 & 0 & \cdots & \psi_M & 0 \\ 0 & \psi_1 & \cdots & 0 & \psi_M \end{pmatrix}, \quad \Psi_{2i-1} = \begin{pmatrix} \phi_i \\ 0 \end{pmatrix}, \quad \Psi_{2i} = \begin{pmatrix} 0 \\ \phi_i \end{pmatrix}.$$

则单元刚度矩阵 $a_T(\psi, \psi)$ 的分量为

$$[a_T(\Psi, \Psi)]_{ij} = a_T(\Psi_j, \Psi_i), \quad i, j = 1, \ldots, 2M.$$

单元负载向量 $(\psi, f)_T$ 的分量为

$$[(\Psi, f)_T]_i = (\Psi_i, f)_T, \quad i = 1, \ldots, 2M.$$

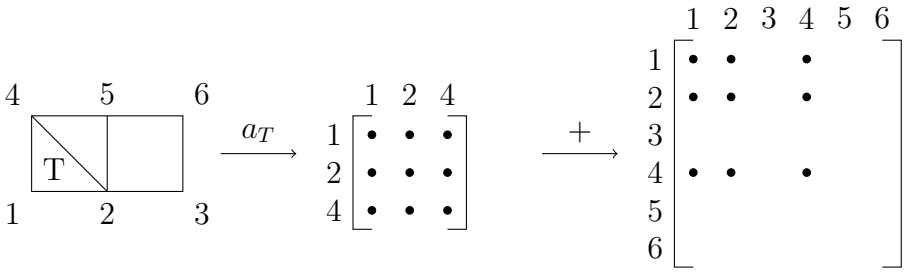

图 1. 将单元刚度矩阵拼入总刚度矩阵

现在我们对整个空间 Ω 上的节点标号, 使得相邻两个有限元自由度意义相同的节点共用. 以待求问题为标量函数, 三角形规则网格, 每个有限元自由度定义为顶点函数值为例, 对网格中节点依次标号, 总刚度矩阵初始是维度为总节点数 $\times$ 总节点数的零矩阵, 计算单元刚度矩阵 a_T, 并根据节点标号将 a_T 加到总刚度矩阵中, 如图 1所示. 由于

我们考虑的是二维向量函数, 标号时要注意每个节点对应两个自由度, 最终得到总刚度矩阵 A, 同样地, 我们可以得到总负载矩阵 b.

对于任意 $x \in \Omega$, 必然存在某个有限元 (T, P_T, Σ_T), 使得 $x \in T$, 由该有限元上节点编号从 c 中截取对应系数 c_T, 则在 T 上的数值解为 $u_h = \Psi c_T$.

2.3. 几何多重网格.

几何多重网格的基本思想是, 在 Ω 上建立一套粗细网格, 在最细层上磨光, 消去残量的高频部分, 然后将残量投影到粗网格上进行校正, 重复若干次, 直到得到满足精度的解.

磨光算法采用的 Gauss-Seidel 迭代和 SSOR 迭代 (松弛因子 ω 取 1). 因为刚度矩阵 A 是对称正定的, 这两种算法具有收敛性, 磨光效果好. 而 Jacobi 迭代收敛的充分条件为 $2\mathrm{diag}(A) - A$ 正定, 本问题不满足这一条件.

具体地, 将 A 分解为 $A = D - L - U$, 其中 D 为对角矩阵, L 为对角线元素为 0 的下三角矩阵, U 为对角线元素为 0 的上三角矩阵, 则 Gauss-Seidel 迭代格式为

$$(D - L)c_j = Uc_{j-1} + b, \quad j = 1, 2, \dots, \tag{2.1}$$

SSOR 迭代格式为

$$\begin{aligned} (D - \omega L)c_{j-1/2} =& (\omega U + (1-\omega)D)c_{j-1} + \omega b, \\ (D - \omega U)c_j =& (\omega L + (1-\omega)D)c_{j-1/2} + \omega b. \end{aligned} \tag{2.2}$$

设 $\mathcal{T}_1$ 是 Ω 的粗网格剖分, 将 $\mathcal{T}_1$ 中每个三角形 T 取中点并两两连接, 得到加细网格 $\mathcal{T}_2$, 不断重复, 最细网格为 $\mathcal{T}_L$. 在 $\mathcal{T}_l (1 \le l \le L)$ 上建立有限元空间 V_l, 则

$$V_1 \subset V_2 \subset \cdots \subset V_L.$$

记 $\mathcal{Q}_{l-1} : V_l \mapsto V_{l-1}$ 为 L^2 投影算子, 则共轭算子 $\mathcal{Q}_{l-1}^* : V_{l-1} \mapsto V_l$ 为自然嵌入. 算子在基函数上的矩阵形式记作 Q_{l-1}, Q_{l-1}^T.

现在, 假设我们已经计算出 $\mathcal{T}_L$ 网格上的刚度矩阵 A, 和负载向量 b, 则几何多重网格见算法 1. $p = 1$ 为 V-cycle 几何多重网格, $p = 2$ 为 W-cycle 几何多重网格. 实际计算中, 多次调用 $c \leftarrow \mathrm{GMG}(L, \mathcal{T}_L, A, b, c)$ 直至所需精度.

3. 数值实验

3.1. 实验介绍.

我们考虑区域 $\Omega = (0, 1)^2$, 最粗层网格 $\mathcal{T}_1$ 的网格粒度 $h = 1/8$, 由 MATLAB 的 `initmesh` 得到, 细层网格由 `refinemesh` 得到.

算法 1 几何多重网格 GMG

输入: 网格层数 l, 网格 $\mathcal{T}_l$, 刚度矩阵 $A_l \in \mathbb{R}^{N\times N}$, 负载向量 $b_l \in \mathbb{R}^N$, 自由度初值 $c_0 \in \mathbb{R}^N$, 磨光次数 m, 校正次数 p.

输出: 自由度取值 $c_{2m+1} \in \mathbb{R}^N$.

function GMG$(l, \mathcal{T}_l, A_l, b_l, c_0)$
  根据 $\mathcal{T}_l$ 计算区域内自由度标号 free
  if $l == 1$ **then**
    c_{2m+1}[free] $\leftarrow A_l$[free,free]$^{-1} b_l$[free]
  else
    for $j \leftarrow 1, \ldots, m$ **do** //前磨光
      调用磨光算法, c_j[free] $\leftarrow$ (2.1)或 (2.2), 迭代公式中 A 取 A_l[free,free], b 取 b_l[free]
    end for
    根据 $\mathcal{T}_l$ 计算网格 $\mathcal{T}_{l-1}$, 以及两层网格之间的投影矩阵 Q_{l-1}
    $A_{l-1} \leftarrow Q_{l-1} A_l Q_{l-1}^T, b_{l-1} \leftarrow Q_{l-1}(b_l - A_l c_m), q_0 \leftarrow 0$
    for $j \leftarrow 1, \ldots, p$ **do** //粗层校正
      $q_j \leftarrow$GMG$(l-1, \mathcal{T}_{l-1}, A_{l-1}, b_{l-1}, q_{j-1})$
    end for
    $c_{m+1} \leftarrow c_m + Q_{l-1}^T q_p$
    for $j \leftarrow m+2, \ldots, 2m+1$ **do** //后磨光
      调用磨光算法, c_j[free] $\leftarrow$ (2.1)或 (2.2), 迭代公式中 A 取 A_l[free,free], b 取 b_l[free]
    end for
  end if
  return c_{2m+1}
end function

网格 $\mathcal{T}_1, \mathcal{T}_2$ 见图 2.

实验中我们采用 P_1 协调有限元, 则重心坐标下的基函数为

$$\phi_i = \lambda_i, \quad i = 1, \ldots, 3.$$

刚度矩阵 A 以坐标表示 (COO) 的方式进行稀疏存储, 主要包含三个部分: 非零元素行坐标, 非零元素列坐标, 非零元素值, 再由 `sparse` 得到稀疏矩阵存储. 以 $\mathcal{T}_1$ 网格为例, 自由度个数为 242, 刚度矩阵 $A \in \mathbb{R}^{242\times 242}$ 中非零元个数为 3108. 利用 `spy` 可以看到数据分布情况, 见图 3.

根据 P_1 元自由度的定义, 容易得到粗网格上某一节点基函数 = 细网格上该节点基函数 $+1/2\times$ 相邻节点基函数.

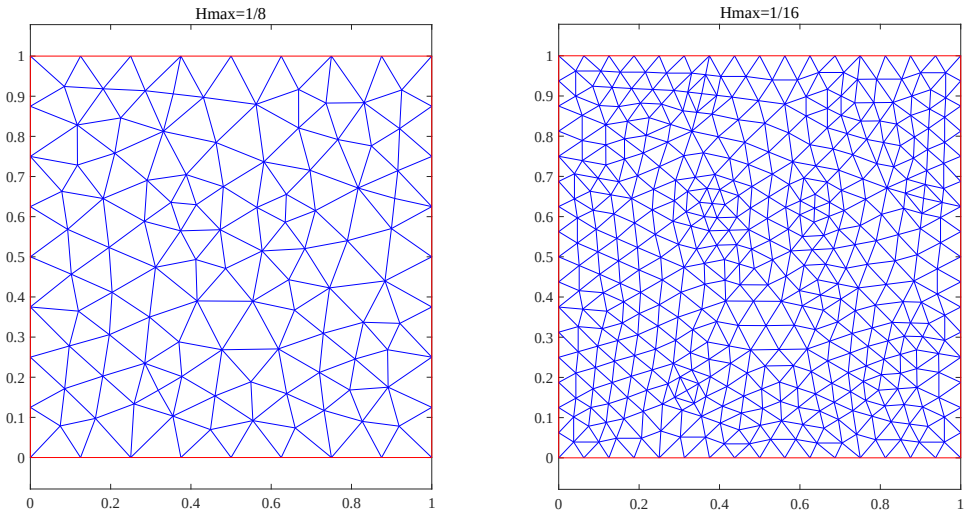

图 2. 左: 网格 $\mathcal{T}_1$, 右: 网格 $\mathcal{T}_2$

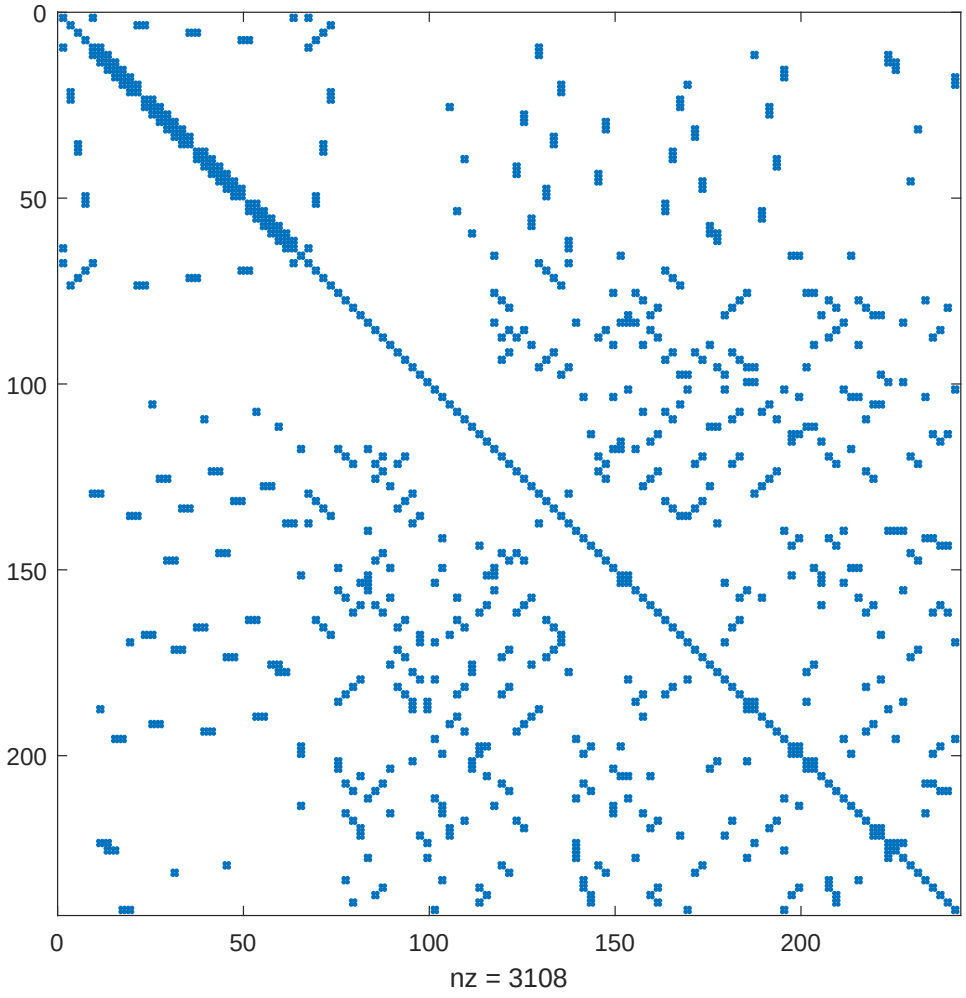

图 3. 最粗层网格上的刚度矩阵 A_1

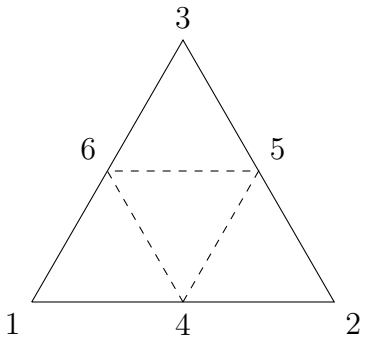

图 4. 实线为粗网格, 虚线为加细网格

以图 4为例, 投影矩阵

$$Q = \begin{pmatrix} 1 & 0 & 0 & 1/2 & 0 & 1/2 \\ 0 & 1 & 0 & 1/2 & 1/2 & 0 \\ 0 & 0 & 1 & 0 & 1/2 & 1/2 \end{pmatrix}.$$

实验中我们采用 W-cycle 几何多重网格, 即 GMG 算法中 (算法 1) 参数 $p = 2$, 前磨光与后磨光次数 $m = 5$. 算法的终止条件为

$$\| b[\text{free}] - A[\text{free}, \text{free}]c[\text{free}] \|_2 \leq \text{1e-8}.$$

我们还需要定义有限元能量范数误差 error=$\|\!|u - u_h|\!\|_h / \|\!|u|\!\|_h$. 这里能量范数定义为

$$\|\!|v|\!\|_h^2 := a_h(v, v) = \sum_{T \in \mathcal{T}_L} a_T(v, v).$$

考虑算例

$$u = \left(-\sin(\pi x)^2 \sin(2\pi y), \sin(2\pi x) \sin(\pi y)^2\right)^T,$$

带入方程计算得

$$f = 2\pi^2 \mu \left(\sin(2\pi y)(1 - 4\sin(\pi x)^2), -\sin(2\pi x)(1 - 4\sin(\pi y)^2)\right)^T.$$

3.2. 磨光算法.

首先选定 Lamé 参数 $\lambda = \mu = 1$, 实验中我们分别选用 Gauss-Seidel 迭代和 SSOR($\omega = 1$) 迭代作为几何多重网格中的磨光算法, 并与两种迭代直接作为求解器进行对比, 并记录了误差 (error), 收敛阶 (rate), 迭代次数 (iter), 运行时间 (time/s), 见表 1.

可以看出, P_1 有限元收敛阶为 1, 符合理论. Gauss-Seidel 迭代和 SSOR 迭代直接作为求解器时, 迭代次数成倍增加, 而几何多重网格的迭代次数保持不变. 在网格粒度 $h = 1/64$ 时, 几何多重网格速度明显优于其它方法, 为求解更细粒度以及更高自由度有限元提供可能. 我

表 1. 不同磨光算法, $\lambda = \mu = 1$

$\iota\backslash h$	1/8	1/16	1/32	1/64
Gauss-Seidel				
error	1.845e-01	9.490e-02	4.814e-02	2.421e-02
rate		0.96	0.98	0.99
iter	184	705	2701	10338
time	0.01	0.02	0.39	13.69
GMG+Gauss-Seidel				
error	1.845e-01	9.490e-02	4.814e-02	2.421e-02
rate		0.96	0.98	0.99
iter	1	6	6	6
time	0.01	0.03	0.23	2.34
SSOR				
error	1.845e-01	9.490e-02	4.814e-02	2.421e-02
rate		0.96	0.98	0.99
iter	153	590	2186	8327
time	0.01	0.02	0.36	13.99
GMG+SSOR				
error	1.845e-01	9.490e-02	4.814e-02	2.421e-02
rate		0.96	0.98	0.99
iter	1	6	6	6
time	0.01	0.04	0.25	2.44

们选用的两种磨光算法效果基本相当, 本实验中我们没有考虑 Jacobi 迭代作为磨光, 是因为本问题的刚度矩阵性质并不能使 Jacobi 迭代收敛.

3.3. **Lamé 常数.**

我们选用 Gauss-Seidel 迭代作为几何多重网格中的磨光算法, 改变 Lamé 常数, 并记录了误差 (error), 收敛阶 (rate), 迭代次数 (iter), 运行时间 (time/s), 见表 2.

注意到我们选用的算例计算出的 f 与 λ 无关, 这时候当 $\lambda \to \infty$ 时, 协调有限元无法收敛, 即所谓的 "闭锁" 问题 [1, Ch.11]. 实验中选取 $\lambda = 10, \mu = 1$ 已经看到降阶的现象, 而 $\lambda = 1, \mu = 10$ 时收敛效果反而更好, $\lambda = 1, \mu = 10$ 的收敛效果与 $\lambda = 1, \mu = 1$ 的收敛效果相当. 事实上, λ/μ 的比值决定了弹性材料的 Poisson 比 ν, 当 $\lambda \to \infty$ 时, $\nu \to 1/2$, 此时材料为不可压缩材料.

表 2. 不同 Lamé 常数, GMG+Gauss-Seidel

$\iota\backslash h$	1/8	1/16	1/32	1/64
$\lambda = 1, \mu = 1$				
error	1.845e-01	9.490e-02	4.814e-02	2.421e-02
rate		0.96	0.98	0.99
iter	1	6	6	6
time	0.01	0.03	0.23	2.34
$\lambda = 10, \mu = 1$				
error	3.040e-01	1.667e-01	8.732e-02	4.453e-02
rate		0.87	0.93	0.97
iter	1	12	12	13
time	0.01	0.09	0.44	4.89
$\lambda = 1, \mu = 10$				
error	1.638e-01	8.320e-02	4.199e-02	2.107e-02
rate		0.98	0.99	1.00
iter	1	6	6	6
time	0.01	0.03	0.23	2.28
$\lambda = 10, \mu = 10$				
error	1.845e-01	9.490e-02	4.814e-02	2.421e-02
rate		0.96	0.98	0.99
iter	1	7	7	7
time	0.01	0.04	0.27	2.65

我们可以粗浅地认为几何多重网格的运行速度与弹性材料的 Poisson 比相关. 当 λ/μ 增大时, 算法的迭代次数与运行时间显著增加. 当 λ 与 μ 等比例增大时, 迭代次数与运行时间略有增加, 但不显著, 几乎可以忽略.

参考文献

1. S. C. Brenner and L. R. Scott, *The Mathematical Theory of Finite Element Methods*, Springer, New York, 2008.