CR-P0 方法求解二维 Stokes 方程

谢文进

2021 年 9 月 29 日

1 理论推导

1.1 问题描述

对于 Stokes 问题

$$\begin{cases} -\nu\Delta\mathbf{u}+\nabla p=f, \quad \text{in } \Omega \\ \nabla\cdot\mathbf{u}=0=h, \quad \text{in } \Omega \\ \mathbf{u}=g=0, \quad \text{on } \partial\Omega \end{cases} \tag{1}$$

其中，取 $\nu=1$。这里函数 $h(\mathbf{x},t)$ 为预留函数接口，方便后续处理随机项。

方程 1的弱形式为：找到 $(\mathbf{u},p)\in\mathbf{H}_0^1(\Omega)\times L_0^2(\Omega)$ 使得

$$\begin{cases} a(\mathbf{u},\mathbf{v})+b(\mathbf{v},p)=(f,\mathbf{v}) \quad \forall\mathbf{v}\in\mathbf{H}_0^1(\Omega), \\ b(\mathbf{u},q)=0=(h,q) \quad \forall q\in L_0^2(\Omega), \end{cases} \tag{2}$$

其中

$$a(\mathbf{u},\mathbf{v})=\nu(\nabla\mathbf{u},\nabla\mathbf{v}), \quad b(\mathbf{v},p)=-(\mathrm{div}\ \mathbf{v},p)$$

对于二维情况，即找到 $u_1\in H_0^1(\Omega)$,$u_2\in H_0^1(\Omega)$, 以及 $p\in L_0^2(\Omega)$ 使得对 $\forall v_1\in H_0^1(\Omega)$, ,$\forall v_2\in H_0^1(\Omega)$, $\forall q\in L_0^2(\Omega)$ 有下式[1] 成立：

$$\int_\Omega \nu(2\frac{\partial u_1}{\partial x}\frac{\partial v_1}{\partial x}+2\frac{\partial u_1}{\partial x}\frac{\partial v_1}{\partial x}+\frac{\partial u_1}{\partial x}\frac{\partial v_1}{\partial x}+\frac{\partial u_1}{\partial x}\frac{\partial v_1}{\partial x}+\frac{\partial u_1}{\partial x}\frac{\partial v_1}{\partial x}+$$
$$\frac{\partial u_1}{\partial x}\frac{\partial v_1}{\partial x})dxdy-\int_\Omega(p\frac{\partial v_1}{\partial x}+p\frac{\partial v_2}{\partial y})dxdy=\int_\Omega(f_1v_1+f_2v_2)dxdy,$$
$$-\int_\Omega(q\frac{\partial u_1}{\partial x}+q\frac{\partial u_2}{\partial y})dxdy=0=\int_\Omega hqdxdy.$$

1.2 Crouzeix-Raviart 三角形元[2]

为方便计算，采用面积坐标 $\lambda_i(\mathbf{x}), i = 1, 2, 3$，$\lambda_i = \frac{S_i}{S}$。Crouzeix-Raviart 三角形元选取的是三边中点，如图 1，可以得到基函数为

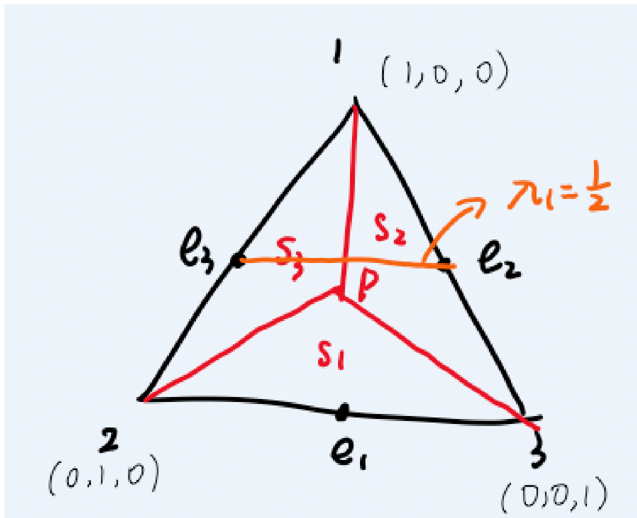

图 1: Crouzeix-Raviart 三角形元

$$\varphi_1 = -2(\lambda_1 - \frac{1}{2}), \tag{3}$$
$$\varphi_2 = -2(\lambda_2 - \frac{1}{2}), \tag{4}$$
$$\varphi_3 = -2(\lambda_3 - \frac{1}{2}). \tag{5}$$

对上述基函数求梯度得：

$$\frac{\partial \varphi_i}{\partial \lambda_j} = \begin{cases} -2, & i = j, \\ 0, & i \neq j. \end{cases} \tag{6}$$

1.3 记号说明

在推导离散格式之前，一些记号及编程数据结构说明[3] 如下：

- elem2dof: 在边上点的全局坐标
- edge: 边

- NE: 边的个数
- NT: 单元个数
- Nu: 速度 u 的个数
- Np: 压力 p 的个数
- N: 顶点的个数

1.4 有限元离散

为了能够由数值实验进行计算，需要考虑有限元空间 $\mathbf{U}_0^h \subset \mathbf{H}_0^1(\Omega)$ 及 $W_0^h \subset L_0^2(\Omega)$，即找到 $\mathbf{u}_h \in \mathbf{U}_0^h$ 及 $p_h \in W_0^h$，使得对 $\forall \mathbf{v}_h \in \mathbf{U}_0^h$ 及 $q_h \in W_0^h$ 有

$$a(\mathbf{u}_h, \mathbf{v}_h) + b(\mathbf{v}_h, p_h) = (f, \mathbf{v}_h),$$
$$b(\mathbf{u}_h, q_h) = 0 = (h, q_h).$$

这里选取 $U_h = span\{\phi_j\}_{j=1}^{N_u}$，$W_h = span\{\psi_j\}_{j=1}^{N_p}$，$N_u$ 表示 $\mathbf{u}_h$ 离散节点个数，N_p 表示 p_h 离散节点个数。ϕ_j 由 Crouzeix-Raviart 三角形元基函数确定，ψ_j 为 P_0 多项式。

参考 Chen [4]，下面推导刚度矩阵及载荷向量：

$$\begin{aligned}
a(\mathbf{u}_h, \mathbf{v}_h) &= a(\sum_i \mathbf{u}_i \phi_i, \sum_j \mathbf{v}_j \phi_j) = \sum_{i,j} a(\phi_i, \phi_j)\mathbf{u}_i \mathbf{v}_j = \mathbf{v}^T \mathbf{A}\mathbf{u},\\
b(\mathbf{v}_h, p_h) &= b(\sum_j \mathbf{v}_j \phi_j, \sum_i p_i \psi_i) = \sum_{i,j} b(\phi_j, \psi_i)\mathbf{v}_j p_i\\
&= [\sum_{i,j} b(\phi_i, \psi_j)\mathbf{v}_i p_j]^T = [\mathbf{p}^\mathbf{T}\mathbf{B}\mathbf{v}]^T = \mathbf{v}^\mathbf{T}\mathbf{B}^\mathbf{T}\mathbf{p},\\
(f, \mathbf{v}_h) &= (\sum_i f_i, \sum_j \mathbf{v}_j \phi_j) = \sum_{i,j} (f_i, \phi_j)\mathbf{v}_j = \mathbf{v}^\mathbf{T}\mathbf{f},\\
b(\mathbf{u}_h, q_h) &= b(\sum_i \mathbf{u}_i \phi_i, \sum_j q_j \psi_j) = \sum_{i,j} b(\phi_i, \psi_j)\mathbf{u}_i q_j = \mathbf{q}^\mathbf{T}\mathbf{B}\mathbf{u},\\
(h, q_h) &= (\sum_i h_i, \sum_j q_j \psi_j) = \sum_{i,j} (h_i, \psi_j) q_j = \mathbf{q}^\mathbf{T}\mathbf{h}.
\end{aligned}$$

写成矩阵形式，即为：

$$\begin{pmatrix} \mathbf{v}^\mathbf{T} & \mathbf{q}^\mathbf{T} \end{pmatrix} \begin{pmatrix} \mathbf{A} & \mathbf{B}^\mathbf{T} \\ \mathbf{B} & \mathbf{O} \end{pmatrix} \begin{pmatrix} \mathbf{u} \\ \mathbf{p} \end{pmatrix} = \begin{pmatrix} \mathbf{v}^\mathbf{T} & \mathbf{q}^\mathbf{T} \end{pmatrix} \begin{pmatrix} \mathbf{f} \\ \mathbf{h} \end{pmatrix},$$

则数值计算要求解的线性系统为

$$\begin{pmatrix} \mathbf{A} & \mathbf{B}^{\mathbf{T}} \\ \mathbf{B} & \mathbf{O} \end{pmatrix} \begin{pmatrix} \mathbf{u} \\ \mathbf{p} \end{pmatrix} = \begin{pmatrix} \mathbf{f} \\ \mathbf{h} \end{pmatrix}. \tag{7}$$

1.5 Dirichlet 边界处理

找到 Dirichlet 边界点，使得在这些点满足

$$\mathbf{u}_i = g_i = 0,$$

对于 Dirichlet 边界条件 $g(\mathbf{x}, t)$，在程序中需要修正矩阵 $\mathbf{A}$,$\mathbf{B}$ 及右端项 $\mathbf{f}$,$\mathbf{h}$。

2 数值实现

2.1 数值实例

取 $\Omega = [0,1] \times [0,1]$，Stokes 方程如下：

$$\begin{cases} -\Delta \mathbf{u} + \nabla p = \mathbf{f}, & \text{in } \Omega \\ \nabla \cdot \mathbf{u} = 0 = h, & \text{in } \Omega \\ \mathbf{u} = 0, & \text{on } \partial\Omega \end{cases} \tag{8}$$

其中

$$\begin{aligned} f_1 = 2^{10}[&(1-6x+6x^2)(y-3y^2+2y^3)+(x^2-2x^3+x^4)(-3+6y) \\ &-(-3+6x)(y^2-2y^3+y^4)], \\ f_2 = -2^{10}[&(-3+6x)(y^2-2y^3+y^4)+(x-3x^2+2x^3)(1-6y+6y^2) \\ &+(1-6x+6x^2)(y-3y^2+2y^3)]. \end{aligned}$$

方程 8的精确解为：

$$\begin{aligned} u_1 &= -2^8(x^2-2x^3+x^4)(2y-6y^2+4y^3), \\ u_2 &= 2^8(2x-6x^2+4x^3)(y^2-2y^3+y^4), \\ p &= -2^8(2-12x+12x^2)(y^2-2y^3+y^4). \end{aligned}$$

2.2 重心坐标

计算重心坐标的梯度[3]

$$\nabla\lambda_i = \frac{1}{2!|\tau|}\mathbf{n}_i \tag{9}$$

其中 $|\tau|$ 为三角形面积，在程序中 $\nabla\lambda_i$ 为Dlambda。

下面推导计算三角形面积 $|\tau|$，程序中变量名为area，记 $\mathbf{l}_i = \mathbf{x}_{i+1} - \mathbf{x}_{i-1}$，而 $\mathbf{n}_i = \mathbf{l}_i^\perp$，其中对于一个向量 $\mathbf{v} = (x, y)$，$\mathbf{v}^\perp = (-y, x)$。对于三角形 T，如图 2，记 $A_i = (x_i, y_i), i = 1, 2, 3$。

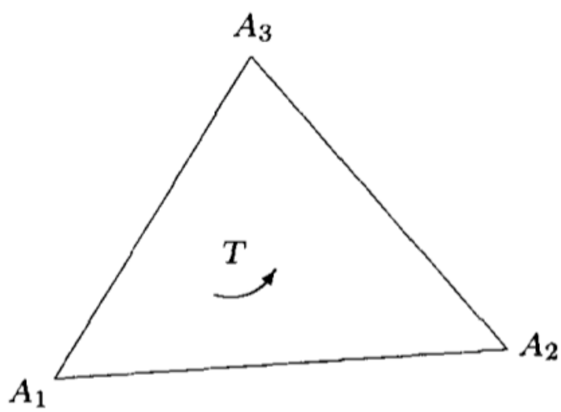

图 2: 三角形

该三角形面积为

$$|\tau| = \frac{1}{2}\begin{vmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{vmatrix}$$

而

$$l_1 = \begin{pmatrix} x_2 - x_3 \\ y_2 - y_3 \end{pmatrix}, l_2 = \begin{pmatrix} x_3 - x_1 \\ y_3 - y_1 \end{pmatrix}, l_3 = \begin{pmatrix} x_1 - x_2 \\ y_1 - y_2 \end{pmatrix},$$

经计算得到

$$
\begin{aligned}
-l_{31}l_{22}+l_{32}l_{21} &= -(x_1-x_2)(y_3-y_1)+(y_1-y_2)(x_3-x_1)\\
&= -(x_1y_3-x_1y_1-x_2y_3+x_2y_1)+(x_3y_1-x_1y_1-x_3y_2+x_1y_2)\\
&= (x_2y_3-x_3y_2)-(x_1y_3-x_3y_1)+(x_1y_2-x_2y_1)\\
&= \begin{vmatrix} x_1 & y_1 & 1\\ x_2 & y_2 & 1\\ x_3 & y_3 & 1 \end{vmatrix},
\end{aligned}
$$

因此

$$|\tau| = \frac{1}{2}(-l_{31}l_{22}+l_{32}l_{21})$$

关于该部分的计算代码见gradbasis.m[5]。

2.3 Laplace 算子

下面推导如何生成矩阵 $\mathbf{A}$，由前述理论推导知

$$
\begin{aligned}
\mathbf{A}_{ij} = a(\phi_i,\phi_j) = (\nabla\phi_i,\nabla\phi_j) = \iint \nabla\phi_i\cdot\nabla\phi_j dxdy\\
= \iint -2\nabla\lambda_i\cdot(-2\nabla\lambda_j)dxdy = 4|\tau|\nabla\lambda_i\cdot\nabla\lambda_j
\end{aligned}
$$

核心代码：

```
A = sparse(Nu,Nu);
for i = 1:3
    for j = i:3
        % 局部节点到全局节点
        ii = double(elem2dof(:,i));
        jj = double(elem2dof(:,j));
        % 局部刚度矩阵
        Aij = 4*dot(Dlambda(:,:,i),Dlambda(:,:,j),2).*area;
        if (j==i)
            A = A + sparse(ii,jj,Aij,Nu,Nu);
        else
            A = A + sparse([ii,jj],[jj,ii],[Aij; Aij],Nu,Nu);
        end
    end
end
clear Aij
A = blkdiag(A,A);
```

2.4 散度算子

下面推导如何生成矩阵 $\mathbf{B}$，由前述理论推导知

$$\begin{aligned}\mathbf{B}_{ij} = b(\phi_i, \psi_j) = -(div\phi_i, \psi_j) = -\iint div\phi_i \cdot \psi_j dxdy \\ = -\iint -2div\lambda_i \cdot \psi_j dxdy = -(-2|\tau| div\lambda_i \cdot \psi_j)\end{aligned}$$

这里 ψ_j 是 $P0$ 元，可以在三角形元中选择一个常数，即置 $\psi_j = C = 1$。

核心代码:

```
d1 = -2.*Dlambda(:, :, 1) .* [area, area];
d2 = -2.*Dlambda(:, :, 2) .* [area, area];
d3 = -2.*Dlambda(:, :, 3) .* [area, area];
Dx = sparse(repmat((1:Np)', 3, 1), double(elem2dof(:)), ...
     [d1(:, 1); d2(:, 1); d3(:, 1)], Np, Nu);
Dy = sparse(repmat((1:Np)', 3, 1), double(elem2dof(:)), ...
     [d1(:, 2); d2(:, 2); d3(:, 2)], Np, Nu);
B = -[Dx Dy];
```

2.5 右端项

先处理右端项 $\mathbf{f}$，由前述理论推导知

$$\mathbf{f}_{ij} = (f_i, \phi_j) = \iint f_i \cdot \phi_j dxdy = |\tau| f_i \cdot \phi_j.$$

核心代码:

```
mid1 = (node(elem(:,2),:) + node(elem(:,3),:))/2; % A2A3边的中点
mid2 = (node(elem(:,3),:) + node(elem(:,1),:))/2;
mid3 = (node(elem(:,1),:) + node(elem(:,2),:))/2;
ft1 = repmat(area, 1, 2).* pde.f(mid1)/3; % 要除以3
ft2 = repmat(area, 1, 2).* pde.f(mid2)/3;
ft3 = repmat(area, 1, 2).* pde.f(mid3)/3;
f1 = accumarray(elem2dof(:), [ft1(:,1); ft2(:,1); ft3(:,1)], [Nu 1]);
f2 = accumarray(elem2dof(:), [ft1(:,2); ft2(:,2); ft3(:,2)], [Nu 1]);
```

下面处理右端项 $\mathbf{h}$，其处理方式与 $\mathbf{f}$ 类似，即

$$\mathbf{h}_{ij} = (h_i, \psi_j) = \iint h_i \cdot \psi_j dxdy = |\tau| h_i \cdot \psi_j.$$

核心代码:

```
ht1 = repmat(area, 1, 2).* pde.h(mid1)/3;
ht2 = repmat(area, 1, 2).* pde.h(mid1)/3;
ht3 = repmat(area, 1, 2).* pde.h(mid1)/3;
h = accumarray(repmat((1:Np)',3,1), [ht1(:,1); ht2(:,1); ht3(:,1)],...
    [Np 1]);
```

2.6 Dirichlet 边界

第一步先找到 Dirichlet 边界点，程序中变量名为fixedDof。

```
ufreeDof = (1:Nu)';
pDof = (1:Np)';

fixedDof = find(isFixedDof); % Dirichlet边界点
ufreeDof = find(¬isFixedDof); % 非Dirichlet边界点
```

第二步修改矩阵 A 及 B。

```
% AD(fixedDof,fixedDof)=I, AD(fixedDof,ufreeDof)=0, ...
    AD(ufreeDof,fixedDof)=0.
% BD(:,fixedDof) = 0 and thus BD'(fixedDof,:) = 0.
bdidx = zeros(2*Nu,1);
bdidx(fixedDof) = 1;
bdidx(Nu+fixedDof) = 1;

% 系统函数 spdiags
%A = SPDIAGS(B,d,m,n) creates an m-by-n sparse matrix from the
%    columns of B and places them along the diagonals specified by d.

Tbd = spdiags(bdidx,0,2*Nu,2*Nu); % AD(fixedDof,fixedDof)=I
% 1 - bdidx = 0, AD(fixedDof,ufreeDof)=0, AD(ufreeDof,fixedDof)=0
T = spdiags(1-bdidx,0,2*Nu,2*Nu);
AD = T*A*T + Tbd;
BD = B*T; % BD(:,fixedDof) = 0
```

第三步，更新右端项。

```
u1 = zeros(Nu,1);
```

```
u2 = zeros(Nu,1);
bdEdgeMid = (node(edge(fixedDof,1),:)+node(edge(fixedDof,2),:))/2;
uD = pde.g_D(bdEdgeMid);            % bd values at middle points of edges
u1(fixedDof) = uD(:,1);
u2(fixedDof) = uD(:,2);
u = [u1; u2]; % Dirichlet bd condition is built into u

% 处理边界条件g_D(u,t)中u涉及到非边界点的情况
f = f - A*u;  % bring affect of nonhomgenous Dirichlet bd condition to

% 处理右端项h
g = h - B*u;
% 无右端项h的情况
% g = g - B*u;  % the right hand side

% mean(g):所有元素的均值,系统函数
g = g - mean(g); % impose the compatible condition

f(fixedDof) = u1(fixedDof);
f(fixedDof+Nu) = u2(fixedDof);
u = [u1; u2]; % Dirichlet bd condition is built into u
```

2.7 数值实验结果

数值实验结果如图 3所示

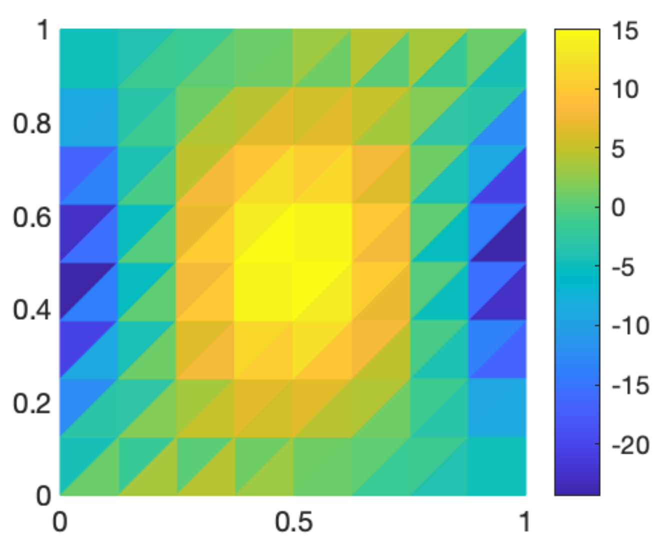

图 3: 数值结果

收敛情况如图 4，可以看到基本 $\mathbf{u}$ 和 p 都可以达到一阶收敛。

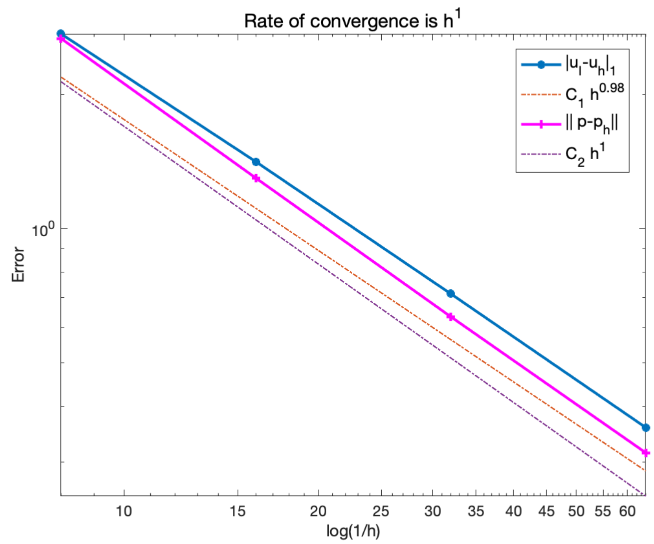

图 4: 收敛阶

2.8 程序清单

- StokesCRP0.mxl

  实时脚本，带入实例计算，求解 Stokes 方程，观察数值结果及收敛阶情况。

- JinJinStokesCRP0.m

  用 CRP0 元求解 Stokes 方程，对 iFEM[5] 包中StokesCRP0.m做了部分修改，加入随机项接口 h。

- MyStokesData2.m

  二维 Stokes 方程实例方程，存储右端项函数 f、h，Dirichlet 边界函数g_D，精确解exactu, exactp。该程序对 iFEM[5] 包中StokesData2.m做部分修改，加入随机函数接口 h。

- gradbasis.m

  返回重心坐标的梯度Dlambda，三角形元面积area。

3 Questions

1.JinJinStokesCRP0.m 105 行，对 p 的处理。

```
%% Post-process ???
if length(pDof) ≠ Np % p is unique up to a constant
    % impose the condition int(p)=0
    c = sum(p.*area)/sum(area);
    p = p - c;
end
```

2.JinJinStokesCRP0.m 246 行，对于只有 Dirichlet 边界的情况，对pDOf的处理。

```
% modfiy pressure dof for pure Dirichlet
if isempty(Neumann)
    pDof = (1:Np-1)'; % ??
end
```

参考文献

[1] HE X. Introduction and Basic Implementation for Finite Element Methods[EB/OL]. https://www.bilibili.com/video/BV1Zv411t7Lj?spm_id_from=333.999.0.0.

[2] 王烈衡, 许学军. 有限元方法的数学基础[M]. 北京: 科学出版社, 2007.

[3] CHEN L. Programming Of Finite Element Methods in MATLAB[EB/OL]. https://www.math.uci.edu/~chenlong/226/Ch3FEMCode.pdf.

[4] CHEN L. Introduction Of Finite Element Methods[EB/OL]. https://www.math.uci.edu/~chenlong/226/Ch2FEM.pdf.

[5] CHEN L. IFEM 软件包[EB/OL]. https://github.com/lyc102/ifem.