PROGRAMMING OF MAC SCHEME FOR STOKES EQUATIONS

LONG CHEN

CONTENTS

In this notes, we present the most popular finite difference method, MAC [2], for the Stokes equations. We consider the steady-state Stokes equations

$$\begin{cases} -\Delta \boldsymbol{u} + \nabla p = \boldsymbol{f} & \text{in } \Omega, \\ -\nabla \cdot \boldsymbol{u} = g & \text{in } \Omega. \end{cases} \tag{1}$$

Here for the sake of simplicity, we fix the viscosity constant $\mu = 1$ and consider a two dimensional rectangular domain. Various boundary conditions will be provided during the discussion.

1. MAC SCHEME

Let $\boldsymbol{u} = (u, v)$ and $\boldsymbol{f} = (f_1, f_2)$. We rewrite the Stokes equations coordinate-wise

$$-\Delta u + \partial_x p = f_1, \tag{2}$$

$$-\Delta v + \partial_y p = f_2, \tag{3}$$

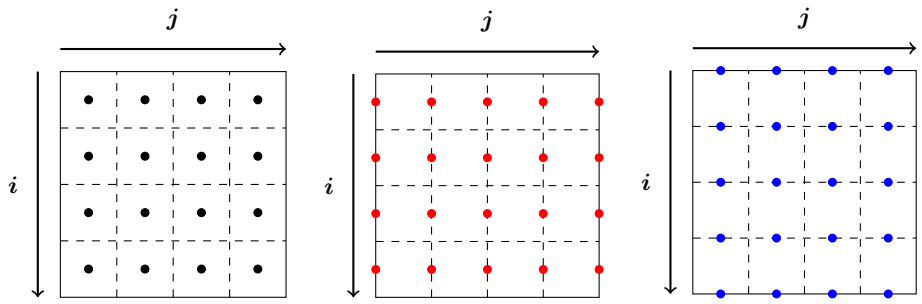
$$-\partial_x u - \partial_y v = g. \tag{4}$$

The domain $\Omega = (0,1)^2$ is decomposed into small squares with size h. We use two dimensional uniform grids for $\Omega = (0,1)^2$ as a typical setting. Generalization to domains composed by rectangles and to three dimensional domains composed by cubes is straightforward but with extra notation.

Standard central difference discretization of Δ and ∂_x, ∂_y at vertices of the uniform grid will not give a stable discretization of Stokes equations due to the failure of the discrete inf-sup condition. To see this, one can view the 5-point stencil as using $\mathcal{P}_1$ element for Laplacian operator and thus discretization at vertices is equivalent to use $\mathcal{P}_1 - \mathcal{P}_1$ unstable pair. Similarly changing pressure discretization to centers of cells corresponds to $\mathcal{P}_1 - \mathcal{P}_0$ which is still not stable. See Finite Element Methods for Stokes equation for discussion on the discrete inf-sup condition.

Date: March 23, 2022.

The idea of MAC, Marker and Cell, is to place the unknown of (u, v, p) in different locations. Specifically the pressure p is located in the center of each cell and the x-component velocity u on the middle points of vertical edges (red dots) and the y-component velocity v on middle points of horizontal edges; see Figure 1.

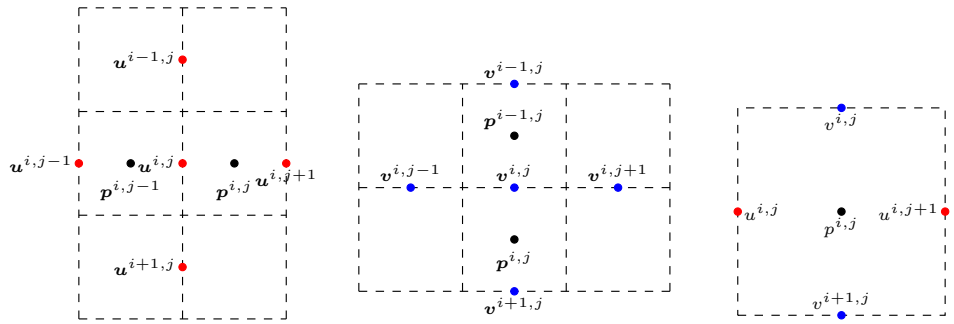


(a) p (b) u (c) v

FIGURE 1. Location and indices of (u, v, p) variables.

The MAC scheme is to discretize the x-coordinate momentum equation (2) at vertical edges, the y-coordinate momentum equation (3) at horizontal edges, and the continuity equation (4) at cell centers using central difference schemes.

Let us introduce the indices system consistent to the matrix, which is easier for the programming: i is the row index and j is the column index, running from $1:n$ or $1:n+1$, where n the number of cells in one direction. The pressure is then represented by a matrix `p(1:n,1:n)`, and the velocity is `u(1:n,1:n+1)`, and `v(1:n+1,1:n)`. One can easily write out the mapping from the index (i, j) to the coordinate (x_j, y_i) for different variables. Note that it is not consistent with the traditional indices system where i for x_i and j for y_j. The advtange to use the proposed index system is that once the mapping (algebraic to geometry mapping) is fixed, in almost all places of coding, we operate on the matrix and such index system is more intuitive to traverse in the matrix. See Programming of Finite Difference Methods for detailed discussion.



(A) equation for u **(B)** equation for v **(C)** equation for p

FIGURE 2. Indices of local stencils of MAC equations

Using this index system, the MAC scheme can be written as

$$\frac{4u^{i,j}-u^{i-1,j}-u^{i+1,j}-u^{i,j-1}-u^{i,j+1}}{h^2}+\frac{p^{i,j}-p^{i,j-1}}{h}=f_1^{i,j} \tag{5}$$

$$\frac{4v^{i,j}-v^{i-1,j}-v^{i+1,j}-v^{i,j-1}-v^{i,j+1}}{h^2}+\frac{p^{i-1,j}-p^{i,j}}{h}=f_2^{i,j} \tag{6}$$

$$-\frac{u^{i,j+1}-u^{i,j}}{h}-\frac{v^{i,j}-v^{i+1,j}}{h}=g^{i,j} \tag{7}$$

Since central difference schemes are used, it is easy to see that the above scheme has second order truncation error at interior nodes.

We then discuss discretization of boundary conditions. For Dirichlet boundary condition, one can impose it in one direction by fixing the value laying on the boundary and by extrapolation on the other direction. Let us take x-coordinate component velocity u as an example. On edges $x=0$ and $x=1$, the value is given by the boundary condition and no equation is discretized on these points. On edges $y=0$ and $y=1$, however, there is no unknowns of u on that edge and we need to modify the stencil at $y=h/2, 1-h/2$. As an example, consider the discretization at the index $(1,j)$. We introduce the ghost value at $y=1+h/2$, i.e. $u(0,j)$. Then we can discretize the momentum equation at $(1,j)$ using (5). The ghost value can be eliminated by the linear extrapolation, i.e, requiring $(u^{0,j}+u^{1,j})/2=u_D(x_j,1)$. Therefore the modified discretization (5) is

$$\frac{5u^{1,j}-u^{2,j}-u^{1,j-1}-u^{1,j+1}}{h^2}+\frac{p^{1,j}-p^{1,j-1}}{h}=f_1^{1,j}+\frac{2u_D(x_j,1)}{h^2}.$$

In short

$$(5,-1,-1,-1,-2) \tag{8}$$

is the stencil for u-unknowns near the horizontal boundaries.

We can also use the quadratic extrapolation, i.e., use $u^{1/2,j},u^{1,j},u^{2,j}$ to fit a quadratic function and evaluate at $u^{0,j}$ to get $u^{0,j}=-2u^{1,j}+\frac{1}{3}u^{2,j}+\frac{8}{3}u^{1/2,j}$. The modified boundary scheme is:

$$\frac{6u^{1,j}-\frac{4}{3}u^{2,j}-u^{1,j-1}-u^{1,j+1}}{h^2}+\frac{p^{1,j}-p^{1,j-1}}{h}=f_1^{1,j}+\frac{\frac{8}{3}u_D(x_j,1)}{h^2}.$$

and this near boundary stencil is denoted by

$$(6,-\frac{4}{3},-1,-1,-\frac{8}{3}). \tag{9}$$

The quadratic extrapolation will lead to a better accuracy. It will, however, destroy the symmetry of the matrix since the coefficient connecting $u^{1,j}$ to $u^{2,j}$ is $-4/3$ not -1.

For Neumann boundary condition $\partial u/\partial n|_{\partial\Omega}=g_N$, the ghost value will be eliminated by the central difference discretization $(u^{0,j}-u^{1,j})/h=g_N(x_j,1)$ and the modified stencil is

$$\frac{3u^{1,j}-u^{1,j-1}-u^{1,j+1}-u^{2,j}}{h^2}+\frac{p^{1,j}-p^{1,j-1}}{h}=f_1^{1,j}+\frac{g_N(x_j,1)}{h}.$$

Unlike the Dirichlet boundary condition, similar modification is needed for all grids points on or near the boundary edges and for points near corners two ghost degree of freedom (dof) should be introduced; see Finite Difference Methods.

We can write the discrete problem in the matrix form with familiar notation:

$$\begin{pmatrix} A & B^{\intercal} \\ B & 0 \end{pmatrix}\begin{pmatrix} \boldsymbol{u}_h \\ p_h \end{pmatrix}=\begin{pmatrix} \boldsymbol{f}_h \\ g_h \end{pmatrix}, \tag{10}$$

where $A = -\Delta, B = -\operatorname{div}$, and $B^\intercal = \operatorname{grad}$. But there is no need to form these matrices explicitly when implement the MAC scheme. See the matrix-free implementation in Programming of Finite Difference Methods.

Exercise 1.1. Verify the matrix $BB^\intercal$ is the standard 5-point stencil of Laplacian operator discretized at cell centers and with Neumann boundary conditions. Therefore the pressure is unique up to a constant.

2. Multigrid Methods for MAC Scheme

In this section we present a multigrid method for solving the MAC discretization of Stokes equations (10). We describe the so-called Distributive Gauss-Seidel (DGS) smoother developed by Brandt and Dinar [1] and prolongation and restriction operators and refer to Programming of MultiGrid Methods for the implementation of the multigrid cycles.

2.1. Distributive Gauss-Seidel Relaxation.

The standard relaxations, e.g., the Gauss-Seidel relaxation, are not applicable to the system (1), since the matrix is not diagonally dominant, and especially the zero block in the diagonal hampers the relaxation of the pressure. The idea of the distributive relaxation is to transform the principle operators to the main diagonal and apply the equation-wise decoupled relaxation.

Denote by

$$\mathcal{L} = \begin{pmatrix} A & B^\intercal \\ B & 0 \end{pmatrix}, \quad \text{and } \mathcal{M} = \begin{pmatrix} I & B^\intercal \\ 0 & -BB^\intercal \end{pmatrix}$$

then

$$\mathcal{L}\mathcal{M} = \begin{pmatrix} A & AB^\intercal - B^\intercal BB^\intercal \\ B & BB^\intercal \end{pmatrix} \approx \begin{pmatrix} A & 0 \\ B & BB^\intercal \end{pmatrix} := \widetilde{\mathcal{L}\mathcal{M}}.$$

By " $\approx$ " here we mean that the commutator $E = AB^\intercal - B^\intercal BB^\intercal$ is small or is of low rank. Thus it can be omitted when designing relaxation methods.

Let us justify that the commutator E is kind of 'small'. In the continuous level, with certain assumptions on the smoothness and boundary conditions, the $(1,2)$ block of $\mathcal{L}\mathcal{M}$ can be manipulate as

$$(-\Delta + \operatorname{grad}\ \operatorname{div})\operatorname{grad} = \operatorname{curl}\operatorname{curl}\operatorname{grad} = 0.$$

Here we use the identity

$$-\Delta = -\operatorname{grad}\ \operatorname{div} + \operatorname{curl}\operatorname{curl}, \tag{11}$$

which holds in H^{-1} topology, and the fact

$$\operatorname{curl}\operatorname{grad} = 0. \tag{12}$$

In short it can be summarized as

$$\Delta_u \operatorname{grad}_p = \operatorname{grad}_u \Delta_p. \tag{13}$$

Here we use subscript to indicate operators associated to velocity and pressure. Unfortunately (13) does not hold exactly for Stokes equations due to the boundary condition of velocity.

Exercise 2.1. Verify for MAC scheme, $E(i,j) = 0$ for interior indices $1 < i, j < n$. Thus the rank of E is at most $n^2 - (n-1)^2$.

Nevertheless, the matrix $\mathcal{M}(\widetilde{\mathcal{L}\mathcal{M}})^{-1}$ will be an good approximation of $\mathcal{L}^{-1} = \mathcal{M}(\mathcal{L}\mathcal{M})^{-1}$. It defines an iterative method for the original system

$$\boldsymbol{x}^{k+1} = \boldsymbol{x}^k + \mathcal{M}(\widetilde{\mathcal{L}\mathcal{M}})^{-1}(\boldsymbol{b} - \mathcal{L}\boldsymbol{x}^k), \tag{14}$$

where to simply the notation denoted by $\boldsymbol{x}^k := (u^k, p^k), \boldsymbol{b} := (f, g)$.

The so-called DGS smoother introduced by Brandt and Dinar [1] comes from consecutively Gauss-Seidel relaxation applied to the operator $\widetilde{\mathcal{L}\mathcal{M}}$. That is solving the transformed residual equation

$$\begin{pmatrix} A & 0 \\ B & A_p \end{pmatrix} \begin{pmatrix} e_u \\ e_p \end{pmatrix} = \begin{pmatrix} r_u \\ r_p \end{pmatrix}$$

approximately and update it through the distributive matrix $\mathcal{M}$. Let $\hat{A}, \hat{A}_p$ be approximation of A and $A_p = BB^\intercal$, respectively.

Algorithm (DGS) $[u^{k+1}, p^{k+1}] \leftarrow \texttt{DGS}(u^k, p^k, f, g)$

(1) Relax the momentum equation

$$u^{k+\frac{1}{2}} = u^k + \hat{A}^{-1}(f - Au^k - B^\intercal p^k),$$

(2) Relax the transformed mass equation

$$e_p = \hat{A}_p^{-1}(g - Bu^{k+\frac{1}{2}}).$$

(3) Distribute the correction to the original variables

$$u^{k+1} = u^{k+\frac{1}{2}} + B^\intercal e_p,$$
$$p^{k+1} = p^k - A_p e_p.$$

The update formulae of pressure can be also interpret as a Uzawa method using $A_p\hat{A}_p^{-1}$ as an approximation for the inverse of the Schur complement. From this point of view, the step (3) can be merged as $p^{k+1} = p^k + Bu^{k+1/2} - g$.

The name "distributive relaxation" is due to the fact that the correction e_p is distributed over u and p through the distributive matrix $\mathcal{M}$.

Remark 2.2. As the pressure is unique up to a constant, it is recommended to normalize p^{k+1} after the update. On the uniform grid, `p = p - mean(p)` will ensure $\int_\Omega p\,\mathrm{d}x = 0$. For non-uniform grids, weighted average with cell areas can be used.

2.2. **Matrix-free implementation.** In the DGS iteration, only algebraic manipulation not the geometric realization of the indices is needed. The neighboring indices of (i, j) for u, v, p can be found in Fig 2 in the previous section.

When implement the Gauss-Seidel iteration for finite difference methods, the direct update version is more efficient than the correction version. For example, the update for velocity `u(i,j)` for `i=2:n-1,j=2:n` is simply

```
1       u(i,j) = (h^2*f1(i,j) + u(i-1,j) + u(i+1,j) + u(i,j-1) + u(i,j+1)...
2               - h*(p(i,j) - p(i,j-1)))/4;
```

For boundary and near boundary subscript `i,j`, the stencil should be modified to impose the boundary condition. Red-black ordering can be used to vectorize the Gauss-Seidel iteration. See Programming of Finite Difference Methods.

After the update of velocity, we need to update the residual for the continuity equation, i.e. compute $r_p = g - Bu = g + \operatorname{div} u$ which can be done efficiently using the index relation of p and u, v; see again Fig. 2 (C).

Then we apply Gauss-Seidel iteration to solve the elliptic equation $A_p e_p = r_p$. Here recall that $A_p = BB^\intercal$ is the discrete Laplacian of pressure with Neumann boundary condition, i.e., $(4,-1,-1,-1,-1)$ for interior cells and $(3,-1,-1,-1)$ for boundary cells and $(2,-1,-1)$ for corners (dividing by h^2). One Gauss-Seidel iteration for interior cells will be

```
ep(i,j) = (h^2*rp(i,j) + ep(i-1,j) + ep(i+1,j) + ep(i,j-1) + ep(i,j+1))/4;
```

For boundary and corner cells, the stencil should be modified accordingly.

Then we should bring the correction `eq` to `u, v, p` through the distributive matrix. Note that $B^\intercal$ is just the discrete gradient operator. For example, the update of u will be

```
u(i,j) = u(i,j) + (ep(i,j) - ep(i,j-1))/h;
```

The update of pressure requires the computation of $A_p e_p$. Its matrix-free version is

```
p(i,j) = p(i,j) - (4*ep(i,j)-ep(i-1,j)-ep(i+1,j)-ep(i,j-1)-ep(i,j+1))/h^2;
```

Again for boundary and corner cells, the stencil should be modified accordingly.

2.3. **Transfer Operators.** At u- and v-grid points, we consider six-points restrictions, and at p-grid points, a four-point cell-centered restriction. In the finite difference method, the restriction is also applied to function values which is different with the finite element one in a scaling.

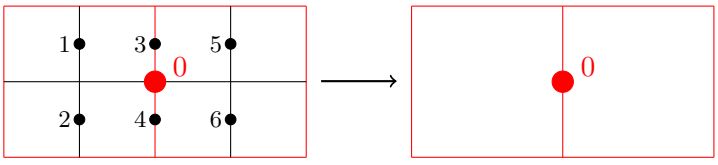
In stencil notation, the restriction operators are

$$R^u_{h,2h} = \frac{1}{8}\begin{pmatrix} 1 & 2 & 1 \\ & * & \\ 1 & 2 & 1 \end{pmatrix}, \quad R^v_{h,2h} = \frac{1}{8}\begin{pmatrix} 1 & & 1 \\ 2 & * & 2 \\ 1 & & 1 \end{pmatrix}, \quad R^p_{h,2h} = \frac{1}{4}\begin{pmatrix} 1 & & 1 \\ & * & \\ 1 & & 1 \end{pmatrix}.$$

Let us explain the restriction operator $R^u_{h,2h}$ in details. Look at Figure 3 for a vertical edge center degree of freedom. Point 0 is a degree of freedom for u on the coarse grid, and points $1,\cdots,6$ are degree of freedoms on the fine grid. Therefor the restriction at point 0 will be

$$u_0 = \frac{1}{8}(u_1 + u_2 + 2u_3 + 2u_4 + u_5 + u_6).$$

The explanation for the restriction operators $R^v_{h,2h}$ and $R^p_{h,2h}$ are similar. Note that $R^u_{h,2h}$ and $R^v_{h,2h}$ are only defined for interior edges. For Dirichlet boundary conditions, the correction and residual on the boundary edges are zero.



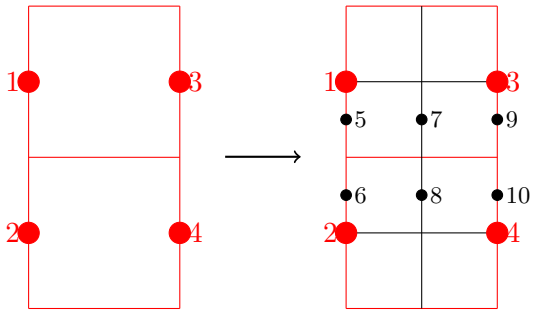
(A) Fine grid **(B)** Coarse grid

FIGURE 3. Sketch of restriction for vertical edge center dof 0 on coarse grid, (A) fine grid points $1,\cdots,6$ and coarse grid point 0 on fine grid. (B). coarse grid point 0.

For the prolongation operators, we typically apply bilinear interpolation of neighboring coarse-grid unknowns in the staggered grid. See Figure 4. To compute the values at points $5,\cdots,10$, we first calculate a bilinear function using coarse grid points $1,\cdots,4$, and the

value for fine grid points $5, \cdots, 10$ are just the evaluation of the bilinear function at these points. One can easily derive that:

$$u_5 = \frac{3}{4}u_1 + \frac{1}{4}u_2,\ u_6 = \frac{3}{4}u_2 + \frac{1}{4}u_1,$$
$$u_9 = \frac{3}{4}u_3 + \frac{1}{4}u_4,\ u_{10} = \frac{3}{4}u_4 + \frac{1}{4}u_3,$$
$$u_7 = \frac{1}{2}(u_5 + u_9),\ u_8 = \frac{1}{2}(u_6 + u_{10}).$$



(A) Coarse grid (B) Fine grid

FIGURE 4. Sketch of prolongation for vertical edge center dofs. (A) coarse grid points $1, \cdots, 4$ on coarse grid. (B). coarse grid points $1, \cdots, 4$ and fine grid points $5, \cdots, 10$ on fine grid.

Following the same approach, one can also derive formulae for computing horizontal edge center degree of freedoms on the fine grid from the coarse grid solution. Piecewise constant interpolation for the pressure p is used.

2.4. **Multigrid Cycles.** We formulate multigrid cycles below and explain afterwards.

```
function [u,v,p] = MG-MAC(u,v,p,f,g,J,m)
%% Direct update form of Multigrid Method
if J == 1  % coarsest level: solve by DGS iterations
    for i = 1:100
         [u,v,p] = DGS(u,v,p,f,g);
    end
end
% Presmoothing
for i = 1:m
    [u,v,p] = DGS(u,v,p,f,g);
end
% Form residual
[ru,rv,rp] = FormResidual(u,v,p,f,g,J);
% Restriction
[ruc,rvc,rpc] = Res(ru,rv,rp);
% Coarse grid correction
[euc,evc,epc] = MG-MAC(0,0,0,ruc,rvc,rpc,J-1,m);
if W-cycle
    [euc,evc,epc] = MG-MAC(euc,evc,epc,ruc,rvc,rpc,J-1,m);  % W-cycle
```

```
end
% Prolongation
[u,v,p] = [u,v,p] + Pro(euc,evc,epc);
% Postsmoothing
for i = 1:m
    [u,v,p] = DGS'(u,v,p,f,g);
end
```

Since we consider a square domain with uniform grids, one parameter `J` for level is sufficient. The mesh size will be $h = 1/2^J$ and the index range can be determined accordingly. All matrix-vector product can be implemented in matrix-free fashion.

The boundary condition of `u,v` can be imposed in the input and all later smoothing are applied to the interior nodes only. When form the residual and prolongate the correction, all boundary values of the residual or correction are set to zero.

In the post-smoothing, it is better to reserve the ordering of G-S iteration so that the operator is symmetric.

REFERENCES

[1] A. Brandt and N. Dinar. Multigrid solutions to elliptic flow problems. In S. Parter, editor, *Numerical Methods for Partial Differential Equations*, pages 53–147. Academic Press, New York, 1979. 4, 5

[2] F. Harlow and J. Welch. Numerical calculation of time-dependent viscous incompressible flow of fluid with free surface. *Physics of fluids*, 8(12):2182, 1965. 1