Lid Driven Cavity – Incompressible

Michael Nameika

1. (a) For the approximations of the derivatives for the nonlinear convection terms, consider the cells above with uniform grid spacing, $h := \Delta x = \Delta y$. Then in general, for a cell-centered approach, we have

$$\begin{aligned}\frac{\partial u^2}{\partial x} &\approx \frac{(u_e)^2 - (u_w)^2}{h}\\ \frac{\partial uv}{\partial x} &\approx \frac{u_e v_e - u_w v_w}{h}\\ \frac{\partial uv}{\partial y} &\approx \frac{u_n v_n - u_s v_s}{h}\\ \frac{\partial v^2}{\partial y} &\approx \frac{(v_n)^2 - (v_s)^2}{h}.\end{aligned}$$

Using a linear interpolant to determine u, v at the cell boundaries, we have

$$\begin{aligned}u_e &\approx \frac{u_E + u_P}{2}, \quad u_w \approx \frac{u_P + u_W}{2}\\ u_n &\approx \frac{u_N + u_P}{2}, \quad u_s \approx \frac{u_P + u_S}{2}\\ v_e &\approx \frac{v_E + V_P}{2}, \quad v_w \approx \frac{v_P + v_W}{2}\\ v_n &\approx \frac{v_N + v_P}{2}, \quad v_s \approx \frac{v_P + v_S}{2}.\end{aligned} \tag{1}$$

Putting this together, we have, in general,

$$\begin{aligned}N_x &\approx \frac{1}{4h}\left((u_e)^2 - (u_w)^2 + u_n v_n - u_s v_s\right)\\ N_y &\approx \frac{1}{4h}\left(u_e v_e - u_w v_w + (v_n)^2 - (v_s)^2\right).\end{aligned}$$

Now, on

(i) the interior nodes, we can apply (1) so that

$$\begin{aligned}N_x &\approx \frac{1}{4h}\left((u_E + u_P)^2 - (u_P + u_W)^2 - (u_N + u_P)(v_N + v_P) - (u_P + u_S)(v_P + v_S)\right)\\ N_y &\approx \frac{1}{4h}\left((u_E + u_P)(v_E + v_P) - (u_P + u_W)(v_P + v_W) + (v_N + v_P)^2 - (v_P + v_S)^2\right)\end{aligned}$$

(ii) the top boundary (minus the corners), we have $u_n = 1$, $v_n = 0$. Using this with (1), we have

$$\begin{aligned}N_x &\approx \frac{1}{4h}\left((u_E + u_P)^2 - (u_P + u_W)^2 - (u_P + u_S)(v_P + v_S)\right)\\ N_y &\approx \frac{1}{4h}\left((u_E + u_P)(v_E + v_P) - (u_P + u_E)(v_P + v_E) - (v_P - v_S)^2\right).\end{aligned}$$

(iii) the bottom left corner, from the boundary conditions, we have $u_w = v_w = u_s = v_s = 0$. Using this with (1) gives

$$\begin{aligned}N_x &\approx \frac{1}{4h}\left((u_E + u_P)^2 - (u_N + u_P)(v_N + v_P)\right)\\ N_y &\approx \frac{1}{4h}\left((u_E + u_P)(v_E + v_P) - (v_N + v_P)^2\right)\end{aligned}$$

(b) For the diffusion terms, we wish to discretize

$$L_x = \frac{1}{Re}\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right)$$
$$L_y = \frac{1}{Re}\left(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2}\right)$$

For the cell-centered finite volume discretization, we have

$$\begin{aligned}\frac{\partial^2 u}{\partial x^2} &\approx \frac{\partial}{\partial x}\left(\frac{\partial u}{\partial x}\right) \\ &= \frac{\left(\frac{\partial u}{\partial x}\right)_e - \left(\frac{\partial u}{\partial x}\right)_w}{h} \\ &\approx \frac{1}{h^2}\left(u_E - u_P - u_P + u_W\right) \\ &= \frac{u_E - 2u_P + u_W}{h^2}.\end{aligned}$$

Similarly,

$$\begin{aligned}\frac{\partial^2 u}{\partial y^2} &\approx \frac{u_N - 2u_P + u_S}{h^2} \\ \frac{\partial^2 v}{\partial x^2} &\approx \frac{v_E - 2v_P + v_W}{h^2}, \qquad \frac{\partial v^2}{\partial y^2} \approx \frac{v_N - 2v_P + v_S}{h^2}.\end{aligned}$$

Using this on

(i) the interior nodes, we have

$$L_x \approx \frac{1}{Re}\left(\frac{u_E + u_W + u_N + u_S - 4u_P}{h^2}\right)$$
$$L_y \approx \frac{1}{Re}\left(\frac{v_E + v_W + v_N + v_S - 4v_P}{h^2}\right).$$

(ii) the top boundary (minus the corners), from the boundary conditions we have $u_n = 1$, $v_n = 0$. Since our finite volume approximation only involves the nodal points, we make the assumption that the boundary conditions extend *out* of the computation region. That is, we assume $u_N = 1$ and $v_N = 0$. Using this, we have

$$L_x \approx \frac{1}{Re}\left(\frac{u_E + u_W + 1 + u_S - 4u_P}{h^2}\right)$$
$$L_y \approx \frac{1}{Re}\left(\frac{v_E + v_W + v_S - 4v_P}{h^2}\right).$$

(iii) the bottom left corner. Similar to (ii), we apply the boundary conditions to $u_E = v_E = v_S = u_S = 0$

$$L_x \approx \frac{1}{Re}\left(\frac{u_W + u_N - 4u_P}{h^2}\right)$$
$$L_y \approx \frac{1}{Re}\left(\frac{v_W + v_N - 4v_P}{h^2}\right).$$

(c) For the pressure Laplacian, we apply the same process as for diffusion terms and use Dirichlet zero boundary conditions. Doing so yields

(i) for the interior nodes,

$$\nabla^2 P \approx \frac{P_E + P_W + P_N + P_S - 4P_P}{h^2}.$$

(ii) for the top boundary (minus the corners),

$$\nabla^2 P \approx \frac{P_E + P_W + P_S - 4P_P}{h^2}.$$

(iii) for the bottom left corner,

$$\nabla^2 P \approx \frac{P_W + P_N - 4P_P}{h^2}.$$

(d) And finally, for the right-hand side of the pressure Poisson equation, we have

$$R = \frac{u_e - u_w + v_n - v_s}{h}.$$

Using (1), the above equation becomes

(i) for the interior nodes,

$$R = \frac{u_E - u_W + v_N - v_S}{h}$$

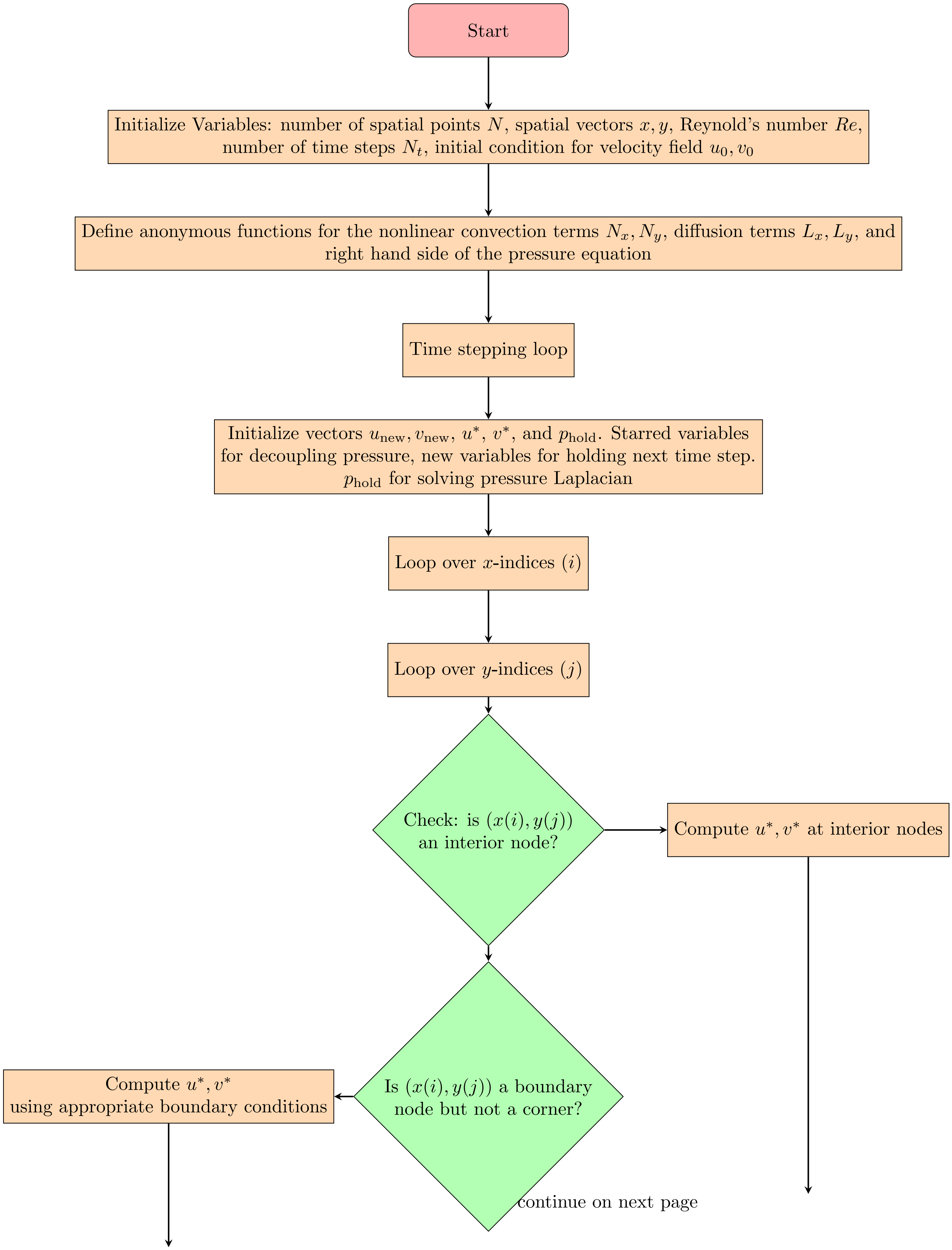
(ii) for the top boundary (minus the corners),

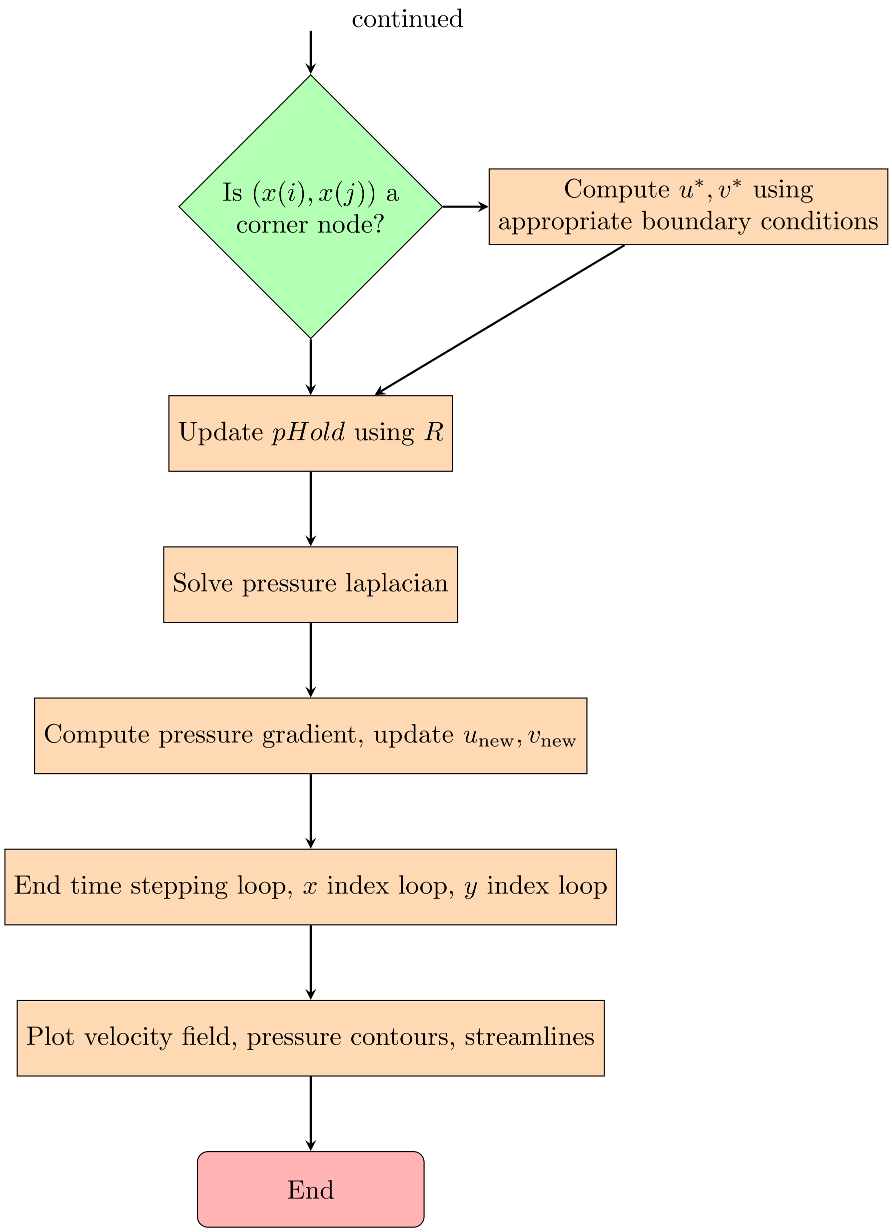
$$R = \frac{u_E - u_W - v_S}{2h}$$

(iii) for the bottom left corner,

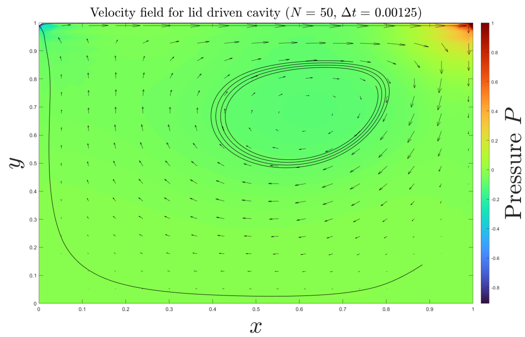
$$R = \frac{u_P + u_E + v_P + v_N}{2h}.$$

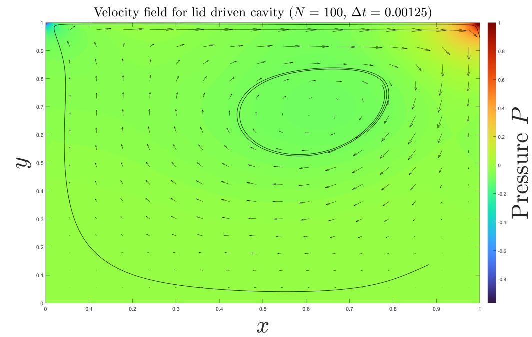
2. Flowchart

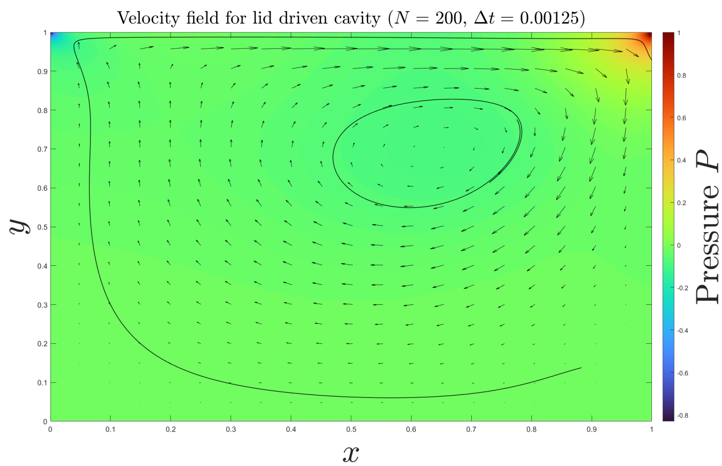

Start
Initialize Variables: number of spatial points N, spatial vectors x, y, Reynold's number Re, number of time steps N_t, initial condition for velocity field u_0, v_0
Define anonymous functions for the nonlinear convection terms N_x, N_y, diffusion terms L_x, L_y, and right hand side of the pressure equation
Time stepping loop
Initialize vectors $u_{\text{new}}, v_{\text{new}}$, u^*, v^*, and p_{hold}. Starred variables for decoupling pressure, new variables for holding next time step. p_{hold} for solving pressure Laplacian
Loop over x-indices (i)
Loop over y-indices (j)
Check: is $(x(i), y(j))$ an interior node?
Compute u^*, v^* at interior nodes
Is $(x(i), y(j))$ a boundary node but not a corner?
Compute u^*, v^* using appropriate boundary conditions
continue on next page


continued
Is $(x(i), x(j))$ a corner node?
Compute u^*, v^* using appropriate boundary conditions
Update $pHold$ using R
Solve pressure laplacian
Compute pressure gradient, update $u_{\text{new}}, v_{\text{new}}$
End time stepping loop, x index loop, y index loop
Plot velocity field, pressure contours, streamlines
End

3. Implementing the above Finite Volume approach for this problem, we find the following results for $N = 50, 100, 200$ at the nondimensional time $t = 20$:







Note that the pressure is lowest in the upper left corner and highest at the upper right corner. This intuitively makes sense since the upper left corner has a most amount of flow away from it and the upper right hand corner has the highest flow towards it. Note also that there is a cyclic region near the center, with small amounts of rotation throughout the rest of the cavity, as is displayed by the streamlines. Further notice that the streamline near the rotational center moves into a more stable orbit for a larger number of spatial points.

4. Code:

```
            %finite volume NS implementation
clear all; close all;

%number of spatial points
Nx = 200;
Ny = Nx;

%spatial grid spacing
h = 1/(Nx-2);
% h = 1/(Nx - 1);

%initial and final times
t0 = 0;
tEnd = 20;

%time step size
dt = 0.01/8;

%number of time steps
Nt = (tEnd - t0)/dt;

%Reynold's number
Re = 200;

%spatial coordinates
x = [0 h/2:h:1-h/2 1];
% x = linspace(0,1,Nx);
y = x;

%meshgrid plots for x and y for plotting purposes
[Xplot, Yplot] = meshgrid(x,y);

%initial conditions for velocity field
u0 = zeros(Nx,Ny);
u0(:,end) = 1;

v0 = zeros(Nx,Ny);


%Laplacian operator for solving pressure poisson equation
onesVec = ones(Nx,1);

%second derivative matrix
D2 = 1/h^2*spdiags([onesVec -2*onesVec onesVec], [-1 0 1], Nx, Ny);

I = speye(Nx,Nx);

%5 point laplacian matrix
Lop = kron(D2,I) + kron(I,D2);

%nonlinear convection terms
N_x = @(ue,uw,un,us,ve,vw,vn,vs) 1/h*( ue^2 - uw^2 + un*vn - us*vs );
```

```
N_y = @(ue,uw,un,us,ve,vw,vn,vs) 1/h*( ue*ve - uw*vw + vn^2 - vs^2 );

%diffusion terms
%use ghost cells for boundary conditions
L_x = @(uP,uE,uW,uN,uS) 1/(Re*h^2)*( uN + uW + uS + uE - 4*uP );
L_y = @(vP,vE,vW,vN,vS) 1/(Re*h^2)*( vN + vW + vS + vE - 4*vP );

%RHS of pressure equation
R = @(ue,uw,un,us,ve,vw,vn,vs) 1/h*( ue - uw + vn - vs );


%time stepping
for k = 1:Nt
    %initialize the starred variablesx
    unew = zeros(Nx,Ny);
    vnew = zeros(Nx,Ny);

    ustar = zeros(Nx,Ny);
    vstar = zeros(Nx,Ny);

    pHold = zeros(Nx,Ny);

    for i = 1:Nx
        for j = 1:Ny

            %interior nodes
            if (i > 1 && i < Nx && j > 1 && j < Ny)

                %interpolate nodes
                ue = (u0(i+1,j) + u0(i,j))/2;
                uw = (u0(i-1,j) + u0(i,j))/2;
                un = (u0(i,j+1) + u0(i,j))/2;
                us = (u0(i,j-1) + u0(i,j))/2;

                ve = (v0(i+1,j) + v0(i,j))/2;
                vw = (v0(i-1,j) + v0(i,j))/2;
                vn = (v0(i,j+1) + v0(i,j))/2;
                vs = (v0(i,j-1) + v0(i,j))/2;


                N1 = N_x(ue, uw, un, us, ve, vw, vn, vs);
                N2 = N_y(ue, uw, un, us, ve, vw, vn, vs);

                L1 = L_x(u0(i,j), u0(i+1,j), u0(i-1,j), u0(i,j+1), u0(
                   i,j-1));
                L2 = L_y(v0(i,j), v0(i+1,j), v0(i-1,j), v0(i,j+1), v0(
                   i,j-1));

                R1 = R(ue, uw, un, us, ve, vw, vn, vs);

                ustar(i,j) = u0(i,j) - dt*N1 + dt*L1;
                vstar(i,j) = v0(i,j) - dt*N2 + dt*L2;

            %Bottom boundary NO CORNERS
```

```
elseif (i > 1 && i < Nx && j == 1)
    ue = (u0(i+1,j) + u0(i,j))/2;
    uw = (u0(i-1,j) + u0(i,j))/2;
    un = (u0(i,j+1) + u0(i,j))/2;
    us = 0;

    ve = (v0(i+1,j) + v0(i,j))/2;
    vw = (v0(i-1,j) + v0(i,j))/2;
    vn = (v0(i,j+1) + v0(i,j))/2;
    vs = 0;


    N1 = N_x(ue, uw, un, us, ve, vw, vn, vs);
    N2 = N_y(ue, uw, un, us, ve, vw, vn, vs);

    L1 = L_x(u0(i,j), u0(i+1,j), u0(i-1,j), u0(i,j+1), 0);
    L2 = L_y(v0(i,j), v0(i+1,j), v0(i-1,j), v0(i,j+1), 0);

    R1 = R(ue, uw, un, us, ve, vw, vn, vs);

    ustar(i,j) = u0(i,j) - dt*N1 + dt*L1;
    vstar(i,j) = v0(i,j) - dt*N2 + dt*L2;

%top boundary NO CORNERS
elseif (i > 1 && i < Nx && j == Ny)
    ue = (u0(i+1,j) + u0(i,j))/2;
    uw = (u0(i-1,j) + u0(i,j))/2;
    un = 1;
    us = (u0(i,j-1) + u0(i,j))/2;

    ve = (v0(i+1,j) + v0(i,j))/2;
    vw = (v0(i-1,j) + v0(i,j))/2;
    vn = 0;
    vs = (v0(i,j-1) + v0(i,j))/2;


    N1 = N_x(ue, uw, un, us, ve, vw, vn, vs);
    N2 = N_y(ue, uw, un, us, ve, vw, vn, vs);

    L1 = L_x(u0(i,j), u0(i+1,j), u0(i-1,j), 1, u0(i,j-1));
    L2 = L_y(v0(i,j), v0(i+1,j), v0(i-1,j), 0, v0(i,j-1));

    R1 = R(ue, uw, un, us, ve, vw, vn, vs);

    ustar(i,j) = u0(i,j) - dt*N1 + dt*L1;
    vstar(i,j) = v0(i,j) - dt*N2 + dt*L2;

%left boundary NO CORNERS
elseif (j > 1 && j < Ny && i == 1)
    ue = (u0(i+1,j) + u0(i,j))/2;
    uw = 0;
    un = (u0(i,j+1) + u0(i,j))/2;
    us = (u0(i,j-1) + u0(i,j))/2;
```

```
        ve = (v0(i+1,j) + v0(i,j))/2;
        vw = 0;
        vn = (v0(i,j+1) + v0(i,j))/2;
        vs = (v0(i,j-1) + v0(i,j))/2;


        N1 = N_x(ue, uw, un, us, ve, vw, vn, vs);
        N2 = N_y(ue, uw, un, us, ve, vw, vn, vs);

        L1 = L_x(u0(i,j), u0(i+1,j), 0, u0(i,j+1), u0(i,j-1));
        L2 = L_y(v0(i,j), v0(i+1,j), 0, v0(i,j+1), v0(i,j-1));

        R1 = R(ue, uw, un, us, ve, vw, vn, vs);

        ustar(i,j) = u0(i,j) - dt*N1 + dt*L1;
        vstar(i,j) = v0(i,j) - dt*N2 + dt*L2;


    %right boundary NO CORNERS
    elseif (j > 1 && j < Ny && i == Nx)
        ue = 0;
        uw = (u0(i-1,j) + u0(i,j))/2;
        un = (u0(i,j+1) + u0(i,j))/2;
        us = (u0(i,j-1) + u0(i,j))/2;

        ve = 0;
        vw = (v0(i-1,j) + v0(i,j))/2;
        vn = (v0(i,j+1) + v0(i,j))/2;
        vs = (v0(i,j-1) + v0(i,j))/2;


        N1 = N_x(ue, uw, un, us, ve, vw, vn, vs);
        N2 = N_y(ue, uw, un, us, ve, vw, vn, vs);

        L1 = L_x(u0(i,j), 0, u0(i-1,j), u0(i,j+1), u0(i,j-1));
        L2 = L_y(v0(i,j), 0, v0(i-1,j), v0(i,j+1), v0(i,j-1));

        R1 = R(ue, uw, un, us, ve, vw, vn, vs);

        ustar(i,j) = u0(i,j) - dt*N1 + dt*L1;
        vstar(i,j) = v0(i,j) - dt*N2 + dt*L2;

    %bottom left corner
    elseif (i == 1 && j == 1)
        ue = (u0(i+1,j) + u0(i,j))/2;
        uw = 0;
        un = (u0(i,j+1) + u0(i,j))/2;
        us = 0;

        ve = (v0(i+1,j) + v0(i,j))/2;
        vw = 0;
        vn = (v0(i,j+1) + v0(i,j))/2;
        vs = 0;
```

```
            N1 = N_x(ue, uw, un, us, ve, vw, vn, vs);
            N2 = N_y(ue, uw, un, us, ve, vw, vn, vs);

            L1 = L_x(u0(i,j), u0(i+1,j), 0, u0(i,j+1), 0);
            L2 = L_y(v0(i,j), v0(i+1,j), 0, v0(i,j+1), 0);

            R1 = R(ue, uw, un, us, ve, vw, vn, vs);

            ustar(i,j) = u0(i,j) - dt*N1 + dt*L1;
            vstar(i,j) = v0(i,j) - dt*N2 + dt*L2;

        %top left corner
        elseif (i == 1 && j == Ny)
            ue = (u0(i+1,j) + u0(i,j))/2;
            uw = 0;
            un = 1;
            us = (u0(i,j-1) + u0(i,j))/2;

            ve = (v0(i+1,j) + v0(i,j))/2;
            vw = 0;
            vn = 0;
            vs = (v0(i,j-1) + v0(i,j))/2;


            N1 = N_x(ue, uw, un, us, ve, vw, vn, vs);
            N2 = N_y(ue, uw, un, us, ve, vw, vn, vs);

            L1 = L_x(u0(i,j), u0(i+1,j), 0, 1, u0(i,j-1));
            L2 = L_y(v0(i,j), v0(i+1,j), 0, 0, v0(i,j-1));

            R1 = R(ue, uw, un, us, ve, vw, vn, vs);

            ustar(i,j) = u0(i,j) - dt*N1 + dt*L1;
            vstar(i,j) = v0(i,j) - dt*N2 + dt*L2;

        %bottom right corner
        elseif (i == Nx && j == 1)
            ue = 0;
            uw = (u0(i-1,j) + u0(i,j))/2;
            un = (u0(i,j+1) + u0(i,j))/2;
            us = 0;

            ve = 0;
            vw = (v0(i-1,j) + v0(i,j))/2;
            vn = (v0(i,j+1) + v0(i,j))/2;
            vs = 0;


            N1 = N_x(ue, uw, un, us, ve, vw, vn, vs);
            N2 = N_y(ue, uw, un, us, ve, vw, vn, vs);

            L1 = L_x(u0(i,j), 0, u0(i-1,j), u0(i,j+1), 0);
            L2 = L_y(v0(i,j), 0, v0(i-1,j), v0(i,j+1), 0);
```

```
            R1 = R(ue, uw, un, us, ve, vw, vn, vs);

            ustar(i,j) = u0(i,j) - dt*N1 + dt*L1;
            vstar(i,j) = v0(i,j) - dt*N2 + dt*L2;

        %top right corner
        elseif (i == Nx && j == Ny)
            ue = 0;
            uw = (u0(i-1,j) + u0(i,j))/2;
            un = 1;
            us = (u0(i,j-1) + u0(i,j))/2;

            ve = 0;
            vw = (v0(i-1,j) + v0(i,j))/2;
            vn = 0;
            vs = (v0(i,j-1) + v0(i,j))/2;


            N1 = N_x(ue, uw, un, us, ve, vw, vn, vs);
            N2 = N_y(ue, uw, un, us, ve, vw, vn, vs);

            L1 = L_x(u0(i,j), 0, u0(i-1,j), 1, u0(i,j-1));
            L2 = L_y(v0(i,j), 0, v0(i-1,j), 0, v0(i,j-1));

            R1 = R(ue, uw, un, us, ve, vw, vn, vs);

            ustar(i,j) = u0(i,j) - dt*N1 + dt*L1;
            vstar(i,j) = v0(i,j) - dt*N2 + dt*L2;
        end



         %solve for the pressure
        pHold(i,j) = 1/dt*R(ue,uw,un,us,ve,vw,vn,vs);

    end %end y indexing for loop
end %end x indexing for loop


pHold = reshape(pHold, Nx*Ny, 1);

pHold = Lop\pHold;

pHold = reshape(pHold, Nx,Ny);

for i = 2:Nx-1
    for j = 2:Ny-1
        Px = 1/(2*h)*( pHold(i+1,j) - pHold(i-1,j) );
        Py = 1/(2*h)*( pHold(i,j+1) - pHold(i,j-1) );

        %update velocity field
        unew(i,j) = ustar(i,j) - dt*Px;
        vnew(i,j) = vstar(i,j) - dt*Py;
```

```
            end
        end

        %update velocity values
        u0 = unew;
        u0(:,end) = 1;

        v0 = vnew;

        %display progress of number of time steps
        prog = k/Nt

    end %end time stepping loop


    %fine grid
    NFine = Nx^2/4;
    xFine = linspace(0,1,NFine);
    yFine = xFine;

    XFine = Xplot;
    YFine = Yplot;
    pFine = pHold;
    % [XFine, YFine] = meshgrid(xFine, yFine);
    % pFine = interp2(Xplot,Yplot, pHold, XFine, YFine, 'spline');


    contourf(XFine, YFine, pFine', -1:0.01:1, 'LineStyle', 'none')
    hold on
    qspace = 10;

    quiver(Xplot(1:qspace:end,1:qspace:end), Yplot(1:qspace:end,1:qspace:
        end), u0(1:qspace:end,1:qspace:end)',v0(1:qspace:end,1:qspace:end)
        ', 1, 'k-', 'linewidth', 0.5)
    cb = colorbar();
    colormap('Turbo')
    % hold off

    axis([0 1 0 1])
    xlabel('$x$', 'fontsize', 45, 'interpreter', 'latex')
    ylabel('$y$', 'fontsize', 45, 'interpreter', 'latex')
    ylabel(cb, 'Pressure $P$', 'fontsize', 45, 'interpreter', 'latex')

    strmln1 = stream2(Xplot,Yplot, u0',v0', 0.7755,0.7755);
    strmln2 = stream2(Xplot,Yplot, u0', v0', 0.883, 0.1378);
    strmln3 = stream2(Xplot,Yplot, u0', v0', 0.7602, 0.413);
    strmln4 = stream2(Xplot,Yplot, u0', v0', 0.6786, 0.7092);

    strmPts = 2000;

    strmDbl1 = strmln1{1};
    strmDbl2 = strmln2{1};
    strmDbl3 = strmln3{1};
```

```
strmDbl4 = strmln4{1};

plot(strmDbl1(1:strmPts,1), strmDbl1(1:strmPts,2), 'k-', 'linewidth',
    0.8)
plot(strmDbl2(:,1), strmDbl2(:,2), 'k-', 'linewidth', 0.8)
% plot(strmDbl3(:,1), strmDbl3(:,2), 'k-', 'linewidth', 0.8)
% plot(strmDbl4(:,1), strmDbl4(:,2), 'k-', 'linewidth', 0.8)

hold off
title("Velocity field for lid driven cavity ($N = $ " + Nx + ", $\
    Delta t = $ " + dt + ")", 'fontsize', 25, 'interpreter', 'latex')
```