Table of Contents

Definitions

```
n = 100;
xc = 0;
yc = 0.6;
r = 0.3;
L = 2;
dx = L/(n-1);
grid = -L/2:dx:L/2;
[x,y] = meshgrid(grid,grid);
c = 1;
dt = 0.01*dx/c;
tfinal = 1.6;
timesteps = ceil(tfinal/dt);
```

Velocity Field

```
u=-cos(pi*(x+0.5)).*sin(3*pi/8*y);
v=sin(pi*(x+0.5)).*cos(3*pi/8*y);
```

Boundary Conditions

```
ip = [2:n,1];   % i+1 = [2,3,4,5,..., n-1, n, 1]
jp = ip;        % j+1 = [2,3,4,5,...n-1,n,1]
im = [n,1:n-1]; % i-1 = [n,1,2,3,4,..., n-2, n-1]
jm = im;        % j-1 = [n,1,2,3,4,..., n-2, n-1]
xv = grid;      % xv = [-L/2, -L/2 + dx,...,L/2-dx,L/2]
yv = grid;      % yv = [-L/2, -L/2 + dy,...,L/2-dy,L/2]
```

Define xT and yT on T

```
theta = 0 : 2*pi/(n-1) : 2*pi;
xT = r.*cos(theta) + xc;
yT = r.*sin(theta) + yc;
```

Initialize plotting variables

```
q = 1:n/20:n; % for quiver plot, showing all points is too crowded
xq = x(q,q); % x for quiver plot
yq = y(q,q); % y for quiver plot
```

```
uq =  u(q,q); % u for quiver plot
vq =  v(q,q); % v for quiver plot
xplot(:,1) = xT; % Initial xT
yplot(:,1) = yT; % Initial yT
```

Advection main loop

```
for t = 1:timesteps
    for istep = 1:n
        % Finds index of xv and yv which are less than or equal
        % to (xT,yT) at time -> t
        ix = find(xv<=xT(istep),1,'last');
        jx = find(yv<=yT(istep),1,'last');
        % Compute velocity at point (xT(istep),yT(istep))
        uab = ...
            1/((xv(ip(ix))-xv(ix))*(yv(jp(jx))-yv(jx)))*...
            (u(ix,jx)*(xv(ip(ix))-xT(istep))*(yv(jp(jx))-yT(istep)) + ...
            u(ip(ix),jx)* (xT(istep)-xv(ix)) * (yv(jp(jx)) - yT(istep)) + ...
            u(ix,jp(jx))*(xv(ip(ix))-xT(istep))*(yT(istep)-yv(jx)) + ...
            u(ip(ix),jp(jx))*(xT(istep)-xv(ix))*(yT(istep)-yv(jx)));
        vab = ...
            1/((xv(ip(ix))-xv(ix))*(yv(jp(jx))-yv(jx)))*...
            (v(ix,jx)*(xv(ip(ix))-xT(istep))*(yv(jp(jx))-yT(istep)) + ...
            v(ip(ix),jx)* (xT(istep)-xv(ix)) * (yv(jp(jx)) - yT(istep)) + ...
            v(ix,jp(jx))*(xv(ip(ix))-xT(istep))*(yT(istep)-yv(jx)) + ...
            v(ip(ix),jp(jx))*(xT(istep)-xv(ix))*(yT(istep)-yv(jx)));
        % Equation (2) -> dx/dt = u(x)
        xTn(istep) = xT(istep) + c*dt/dx*uab; % advect x
        yTn(istep) = yT(istep) + c*dt/dx*vab; % advect y
        % Save values for istep and t
        oxplot(istep,t) = xv(ix); % save x_i that defines our point
        oyplot(istep,t) = yv(jx); % save y_i that defines our point
        quplot(istep,t) = uab; % save uab used at time -> t
        qvplot(istep,t) = vab; % save vab used at time -> t
    end
    xT = xTn; % Save new coordinates
    yT = yTn; % Save new coordinates
    xplot(:,t+1) = xT; % Save for plot
    yplot(:,t+1) = yT; % Save for plot
    % Plot new coordinates
    figure(1);
    plot(oxplot(:,t),oyplot(:,t),'.',xplot(:,t),yplot(:,t),'.');
    hold on;
    quiver(oxplot(:,t),oyplot(:,t),quplot(:,t),qvplot(:,t));
    quiver(xq,yq,uq,vq);
    axis([-1,1,-1,1])
    axis('square')
    hold off;
    pause(0.001)
end
```

Published with MATLAB® R2013a