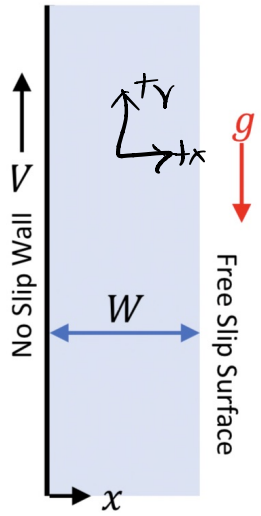
Sahib Azgar

ME456S Computational Fluid Mechanics

Due: 01/17/2024

Prof Allshouse

Problem 1 - Rising Wall: Consider a thin fluid film of width W, that is attached to an infinitely long vertical wall. The fluid falls down due to gravity. At $x = 0$ there is a no-slip wall that is moving upwards at a speed V. At $x = W$, there is a free-slip surface. Assume that there is no velocity in the x direction, there are no pressure gradients, and that the flow is steady.



a. Derive the velocity profile v as a function of x.

(Hand derivation)

mass balance

$$\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z} = 0$$

Given info $\frac{dv}{dy} = 0$

- $x=0$ no-slip wall moving upwards at speed V
- $x=W$, free slip surface
- no velocity x-direction
- no pressure gradients
- flow steady
- infinite long vertical wall
- free slip at $x=W$

linear momentum (no velocity x; infinite wall)

$$\rho\left(\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y} + w\frac{\partial u}{\partial z}\right) = -\frac{\partial P}{\partial x} + \rho g_x + \mu\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 u}{\partial z^2}\right)$$

$$\rho\left(\frac{\partial v}{\partial t} + u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y} + w\frac{\partial v}{\partial z}\right) = -\frac{\partial P}{\partial y} + \rho g_y + \mu\left(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2} + \frac{\partial^2 v}{\partial z^2}\right)$$

$$\rho\left(\frac{\partial w}{\partial t} + u\frac{\partial w}{\partial x} + v\frac{\partial w}{\partial y} + w\frac{\partial w}{\partial z}\right) = -\frac{\partial P}{\partial z} + \rho g_z + \mu\left(\frac{\partial^2 w}{\partial x^2} + \frac{\partial^2 w}{\partial y^2} + \frac{\partial^2 w}{\partial z^2}\right)$$ no z.

$$0 = \rho g_y + \mu\frac{\partial^2 v}{\partial x^2}$$

$$\mu\frac{\partial^2 v}{\partial x^2} = -\rho g_y$$

$$\int \frac{\partial^2 v}{\partial x^2}\,dx = \int \frac{-1}{\mu}\rho g_y\,dx$$

$$\int \frac{\partial v}{\partial x} = \int \frac{-1}{\mu}\rho g_y x + C_1$$

$$v(x) = -\frac{x^2}{2\mu}\rho g_y + C_1 x + C_2$$

$v_1(x=0) = V$

no slip wall: $\left.\frac{\partial v}{\partial x}\right|_{x=0} = 0$

$\tau(x=w) = 0 = \mu\frac{\partial v}{\partial x}$

$$\frac{\partial v}{\partial x} = \frac{1}{\mu}\rho g_y x + C_1$$

$$0 = \mu\frac{\partial v}{\partial x} = \mu\left(\frac{-\rho g_y w}{\mu} + C_1\right)$$

$\hookrightarrow C_1 = \frac{\rho g_y w}{\mu}$

$$v(0) = V = \frac{\rho g}{2\mu}(0)^2 + 0^2 + C_2$$

$\hookrightarrow C_2 = V$

$$\boxed{v(x) = \frac{-\rho g_y x^2}{2\mu} + \frac{\rho g_y w}{\mu} + V}$$

$\nu = \frac{\mu}{\rho}$

b. Calculate the volumetric flux Q of the fluid film.

(Hand derivation)

$$Q = \frac{\int_0^w v\,dx}{w} \rightarrow Q = \frac{\int_0^w \left(\frac{\rho g_y x}{\mu}\left(w - \frac{x}{2}\right) + V\right)dx}{w} = \frac{\frac{\rho g_y}{\mu}\int_0^w wx - \frac{x^2}{2}\,dx + \int_0^w V\,dx}{w}$$

$$Q = \frac{\rho g_y}{\mu}\left(\frac{wx^2}{2} - \frac{x^3}{6}\Big|_0^w\right) + \left(Vx\Big|_0^w\right) \rightarrow Q = \frac{\frac{\rho g_y}{\mu}\left(\frac{w^3}{2} - \frac{w^3}{6}\right) + Vw}{w}$$

$$\boxed{Q = \frac{\rho g_y w^2}{3\mu} + V}$$

c. Calculate the wall velocity V that would result in no net volume flux.

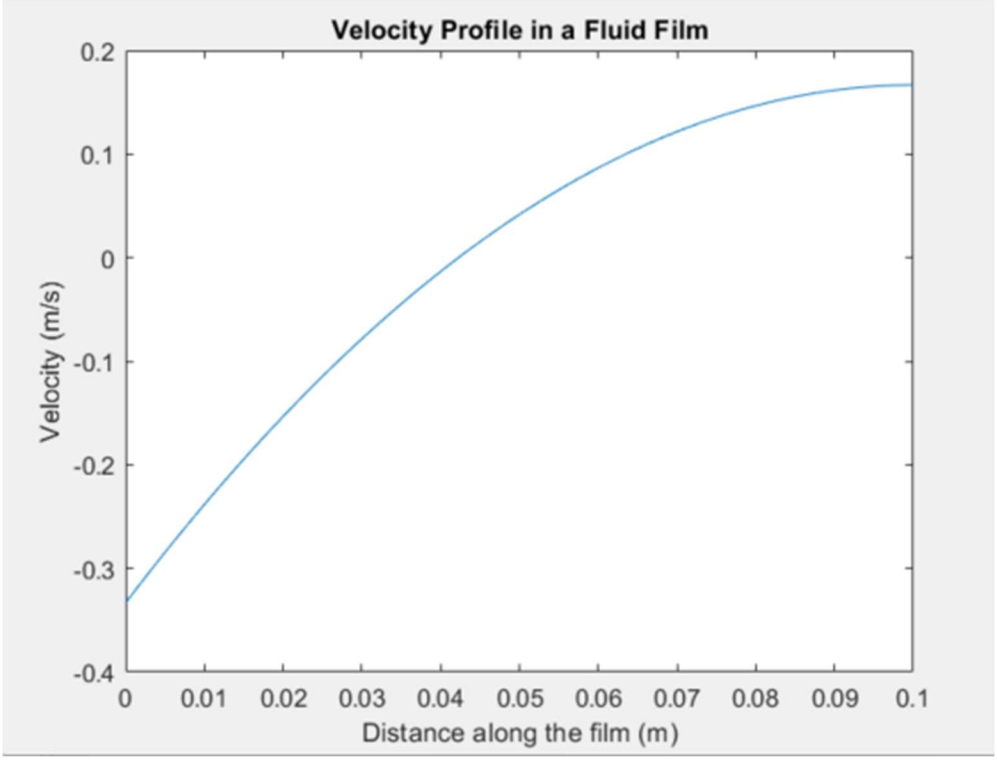
(Hand derivation)

$$Q = 0 = \frac{\rho g_y w^2}{3\mu} + V$$

$$\hookrightarrow \boxed{V = \frac{-\rho g_y w^2}{3\mu}}$$

d. If we let gravity be g=10, the viscosity be nu= 0.1, and the film thickness be W = 0.1. Determine the wall velocity needed to have zero net volume flux and plot the velocity profile *(Matlab script and plot)*

```
% Problem 1 Part d

clear; close all; clc;

% If we let gravity be g=10, the viscosity be nu= 0.1, and the film
% thickness be W = 0.1. Determine the wall velocity needed to have zero
% net volume flux and plot the velocity profile.

g = 10; % acceleration due to gravity (m/s^2)
viscosity = 0.1; % viscosity (Pa·s)
W = 0.1; % film thickness (m)
x = linspace(0,W,100); %smoother plot and to declare x variable
% Calculate wall velocity from Part C
wallVelocity =  -(g * W^2) / (3 * viscosity); % Wall velocity [m/s]

% Calculate the velocity profile from Part A
velocityProfile = wallVelocity + (g/viscosity) .* (W .* x - (.5 * x.^2)); %Velocity
Profile [m/s]

% Plot the velocity profile
plot(x, velocityProfile);
xlabel('Distance along the film (m)');
ylabel('Velocity (m/s)');
title('Velocity Profile in a Fluid Film');
```

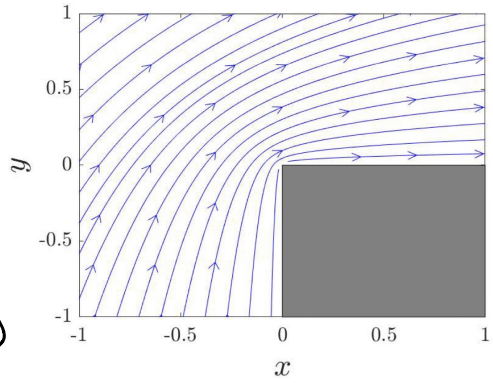


Problem 2 - Flow around a corner: The velocity field for a fluid flow around a corner can be defined by the stream function:

c. Complete the template file to reproduce the figure above. (In the template, you can play with the parameter n to change the corner angle.)

Tl
th

(Completed template and figure)



a. Derive the velocity components u_r and u_θ from $\psi(r,\theta)$. Don't forget to account for the use of polar coordinates!

(Hand derivation)

$u_r = \frac{1}{r}\frac{\partial \psi}{\partial \theta}$ $\quad u_\theta = \frac{-\partial \psi}{\partial r}$

$u_r = \frac{1}{r}\frac{\partial \psi}{\partial \theta}\left(Ar^n \sin(n\theta)\right)$

$u_r = \frac{1}{r} Ar^n \frac{\partial \psi}{\partial \theta}\left(\sin(n\theta)\right)$

$u_r = \frac{1}{r} Anr^n \cos(n\theta)$

$u_r = Anr^{n-1}\cos(n\theta)$ ← if plugging in $n=\frac{2}{3}$

$u_r = \frac{2}{3r^{1/3}} A\cos\left(\frac{2}{3}\theta\right)$

$u_\theta = \frac{-\partial \psi}{\partial r}\left(Ar^n \sin(n\theta)\right)$

$u_\theta = -\sin(n\theta)\frac{\partial \psi}{\partial r}\left(Ar^n\right)$

$u_\theta = -Anr^{n-1}\sin(n\theta)$

$u_\theta = -\frac{2}{3r^{1/3}} A\sin\left(\frac{2}{3}\theta\right)$

b. Given $x = r\cos(\theta)$, and $y = r\sin(\theta)$, relate u_x and u_y to u_r and u_θ.

(Hand derivation)

$x = r\cos\theta$

$u_x = \frac{-d\theta}{dt} r\sin\theta + \frac{dr}{dt}\cos\theta$

$y = r\sin\theta$

$u_y = \frac{d\theta}{dt} r\cos\theta + \frac{dr}{dt}\sin\theta$

$$u_x = -u_\theta \sin\theta + u_r \cos\theta$$

$$u_y = u_\theta \cos\theta + u_r \sin\theta$$

d. Is the flow irrotational? If so derive the velocity potential ϕ.

→ Part c two pages later

Plug in $n = \frac{2}{3}$

(Hand derivation)

Irrotational flow → $\vec{\nabla} \times \vec{u} = 2\vec{\omega}$

$u_r = \frac{2}{3r^{1/3}} A\cos\left(\frac{2}{3}\theta\right)$

$u_\theta = \frac{-2}{3r^{1/3}} A\sin\left(\frac{2}{3}\theta\right)$

$\vec{u} = \frac{2}{3r^{1/3}} A\cos\left(\frac{2}{3}\theta\right)\hat{r} - \frac{2}{3r^{1/3}} A\sin\left(\frac{2}{3}\theta\right)\frac{1}{r}\hat{\theta}$

$2\vec{\omega} = \frac{1}{r}\frac{\partial}{\partial r}(ru_\theta) - \frac{1}{r}\frac{\partial v_r}{v_\theta}$

$= \frac{1}{r}\frac{\partial}{\partial r}\left(r\left(-\frac{2}{3}r^{-1/3} A\sin\left(\frac{2}{3}\theta\right)\right)\right) - \frac{1}{r}\frac{\partial u_r}{\partial \theta}$

$= \frac{1}{r}\frac{\partial}{\partial r}\left(-\frac{2}{3}r^{2/3} A\sin\left(\frac{2}{3}\theta\right)\right) - \frac{1}{r}\frac{\partial}{\partial \theta}\left(\frac{2}{3r^{1/3}} A\cos\frac{2}{3}\theta\right)$

$= r^{-1}\left(\frac{-4}{9} Ar^{-1/3}\sin\left(\frac{2}{3}\theta\right)\right) - \frac{1}{r}\left[-\sin\frac{2}{3}\theta \cdot \frac{2}{3} \cdot \frac{2}{3r^{1/3}} A\right]$

$= \frac{-4}{9} Ar^{-4/3}\sin\left(\frac{2}{3}\theta\right) + \frac{2A}{3r^{4/3}} \cdot \frac{2}{3}\sin\left(\frac{2}{3}\theta\right)$

$= \frac{-4}{9} Ar^{-4/3}\sin\left(\frac{2}{3}\theta\right) + \frac{4}{9} Ar^{-4/3}\sin\left(\frac{2}{3}\theta\right) = 0$ → flow is irrotational

Proving irrotational without n value

$$\frac{1}{r}\frac{\partial(ru_\theta)}{\partial r} - \frac{1}{r}\frac{\partial u_r}{\partial \theta} = \omega$$

$$\frac{1}{r}\left[\frac{\partial(ru_\theta)}{\partial r} - \frac{\partial u_r}{\partial \theta}\right] = \omega$$

$$\frac{1}{r}\left[-An^2 r^{n-1}\sin(n\theta) - -An^2 r^{n-1}\sin(n\theta)\right]$$

$\frac{1}{r}(0) \to 0$ $\therefore$ Irrotational

$$\frac{\partial(ru_\theta)}{\partial r} = \frac{\partial\left(r \cdot -An r^{n-1}\sin(n\theta)\right)}{\partial r} = -An^2 r^{n-1}\sin(n\theta)$$

$$\frac{\partial u_r}{\partial \theta} = \frac{\partial(An r^{n-1}\cos(n\theta))}{\partial \theta} = -An^2 r^{n-1}\sin(n\theta)$$

Velocity Potential

$$\frac{\partial \phi}{\partial r} = u_r = An r^{n-1}\cos(n\theta)$$

$$\int \frac{\partial \phi}{\partial r} = \int An r^{n-1}\cos(n\theta)$$

$$\phi = An\cos(n\theta)\int r^{n-1}\,dr$$

$$\phi = An\cos(n\theta)\left(\frac{r^n}{n}\right) + C_1$$

$$\phi(r) = An r^n \cos(n\theta) + C_1$$

$$\frac{1}{r}\frac{d\phi}{d\theta} = u_\theta = -An r^{n-1}\sin(n\theta)$$

$$r \cdot \frac{1}{r}\frac{\partial \phi}{\partial \theta} = -An r^{n-1}\sin(n\theta) \cdot r$$

$$\int \frac{\partial \phi}{\partial \theta} = \int -An r^n \sin(n\theta)$$

$$\phi = -An r^n \int \sin(n\theta)\,d\theta$$

$$\phi(\theta) = An r^n \cos(n\theta) + C_2$$

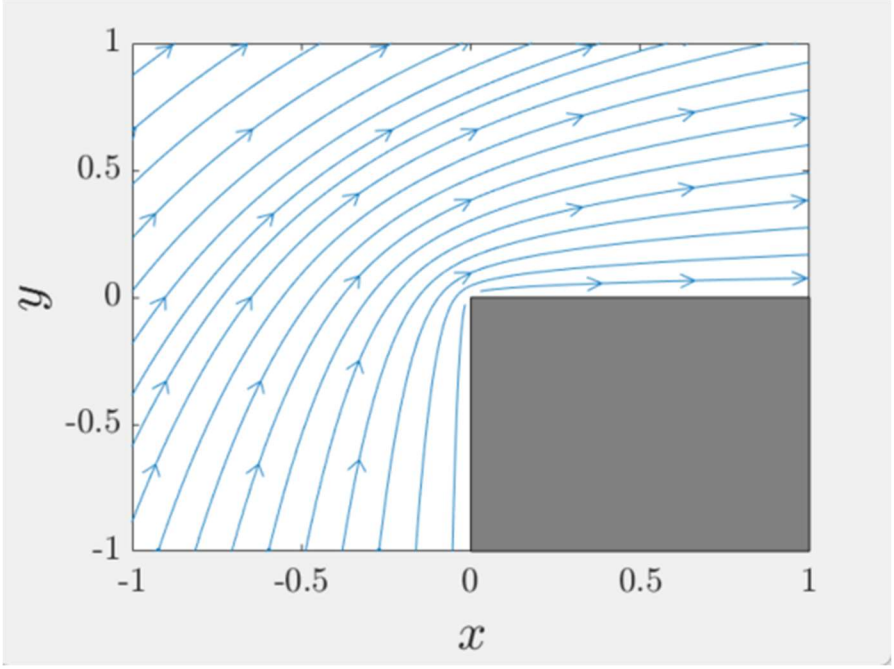
$$\boxed{\phi(r,\theta) = An r^n \cos(n\theta) + C}$$

Part c on next page.

c) Complete the template file to reproduce the figure. (In the template, you can play with the parameter n to change the corner angle).

```
function HW1P2C

%% Set potential flow parameters
A = 1;
n = 2/3;

%% Create the spatial range in Cartesian coordinates
x = -1.1:.025:1.1;
y = -1.1:.025:1.1;

%% Create a grid of points
[X,Y] = meshgrid(x,y);

%% Convert from Cartesian to Polar coordinates
r = sqrt(X.^2 + Y.^2);
theta = myatan(Y,X);

%% Calculate the polar coordinate velocities
v_r = A * n * r.^(n-1) .* cos(n * theta);
v_theta = -A * n * r.^(n-1) .* sin(n * theta);

%% Convert the polar coordinate velocities to Cartesian velocities
v_x = -v_theta.* sin(theta) + v_r.*cos(theta) ;
v_y = v_theta.* cos(theta) + v_r.* sin(theta);

%% Plot the results
streamslice(X,Y,v_x,v_y,'k')
axis([-1 1 -1 1])
hold on;
fill([0 1 1 0 0],[-1 -1 0 0 -1],[.5 .5 .5])
box on

%% Label the figure
set(gca,'FontSize',16,'TickLabelInterpreter','latex');
xlabel('$$x$$','FontSize',24,'Interpreter','latex');
ylabel('$$y$$','FontSize',24,'Interpreter','latex');

end

function theta=myatan(y,x)

theta=0*x;

theta(x>0) = atan(y(x>0)./x(x>0));
theta(y>=0 & x<0) = pi+atan(y(y>=0 & x<0)./x(y>=0 & x<0));
theta(y<0 & x<0)  =-pi+atan(y(y<0 & x<0)./x(y<0 & x<0));
theta(y>0 & x==0) = pi/2;
theta(y<0 & x==0) = -pi/2;
theta(theta<0) = theta(theta<0) + 2*pi;

end
```

Graph for Problem 2 Part C



Problem 3 - Julia Set: The Julia Set is an example of a map where the function is

$$z_{n+1} = F(z_n) = z_n^2 + C \qquad (1)$$

what I wrote is code added

where z and C are complex numbers. As you perform this map many times there are ultimately two fates for the map: it either moves away from the origin ("escapes") or remains close to the origin. What the Julia Set plots is how long it takes for a point to "escape." For this problem, you will write an algorithm that creates a Julia Set figure. There is a template file available on Canvas.

a. First, set the parameters for the particular Julia Set. $C = 0.4 + i0.1$. Note that the imaginary number i in Matlab is represented as `1i`. The maximum number of iterations that we will allow the map to take is 100. This prevents the code from stalling at any one point. Finally, we will set a threshold for determining if a mapping has escaped. In our case, that value will be 4.

```
C = 0.4 + 0.1i
maxIter = 100
escapeValue = 4
```

b. Now we are going to create the range of values we will consider for the real and imaginary parts of $z = x + iy$. To do this you will be using the function `linspace`. `linspace` is a function that creates a vector of equally spaced values between perscribed end points. To learn what the inputs of the function should be type `>> help linspace`. The range of values you should test are $-1 < x < 1$ and $-1.5 < y < 1.5$. You should have 1000 x-values and 1500 y-values.

```
linspace
x_values = linspace(-1,1,1000)
Y_values = linspace(-1.5,1.5,1500)
```

c. Create the variable `count`, which will contain the number of iterations the map takes before escaping for a particular position. You will be performing the mapping for each combination of x and y values, so we are going to create a matrix of zeros to contain the count for each combination. Using the function `zeros(N,M)` you can create a matrix of `N`-rows and `M`-columns. Assuming we want to have each row represent a unique y-value and each column a unique x-value create the appropriate sized `count` variable.

```
Count = zeros(length(Y), length(x));
for I = 1:length(Y)
    for J = 1:length(x)
```

d. We are using the indexes I and J to track which value of Y and X we are using, respectively. Create the initial position $z = x + i * y$.

```
z = x(J) + 1i*Y(I);
```

e. Before starting the mapping, we are going to require that the magnitude of the complex number be less than the escape value. To do this we will use an `if` command. If the magnitude of z is less than the escape value, you will perform the mapping. Otherwise, you will set the count `n` to zero. To find the magnitude of a complex number we use the absolute value function `abs`.

```
if abs < escapeValue
    n = 1;
n=0;
```

f. Finally, we are going to create a `while` loop that runs until either the magnitude of the updated value of `z` is greater than the escape value or the number of iterations taken is greater than the maximum number of iterations.

```
while abs(z) < escapeValue && n < maxIter
    z = z^2 + C;
    n = n+1;
end
```

Problem 3 Entire Code and Result

```
clear; close all; clc;

%% a) Set the parameters for the Julia Set

C = 0.4+ 0.1i; % Constant added to the mapping z_n+1 = z_n^2 + C
maxIter = 100; % Maximum number of iterations considered for a point to escape
escapeValue = 4; % The value above which we consider the mapping of a point to have
escaped

%% b) Create the set of points where we will evaluate the mapping

X = linspace(-1,1,1000);
Y = linspace(-1.5,1.5,1500);

%% c) Create the variable that will track the number of mappings required to escape

count = zeros(length(Y),length(X));

% Loop through each Y value
for I = 1:length(Y)

    % Loop through each X value
    for J = 1:length(X)

        %% d) Create the initial position (z = x + i*y)
        z = X(J) + 1i*Y(I);

        %% e) Create the if condition
        if abs(z) < escapeValue
            n = 1; % Mapping iteration counter
            %% f) Create the while condition
            while abs(z) < escapeValue && n <= maxIter
                z = z^2 + C; % Apply the mapping to update the position
                n = n + 1; % Increase the number of iterations taken
            end

        else
            n = 0; %Set the count n to zero.
        end

        % Store the number of iterations taken in the count variable
        count(I,J) = n;
    end
end

%% Plot the results
imagesc(X,Y,log(count));
colormap jet
xlabel('$$x$$','FontSize',16,'Interpreter','latex')
ylabel('$$y$$','FontSize',16,'Interpreter','latex')
set(gca,'FontSize',14)
```

Graph Result after Code