

ng Potential

## Table of Contents

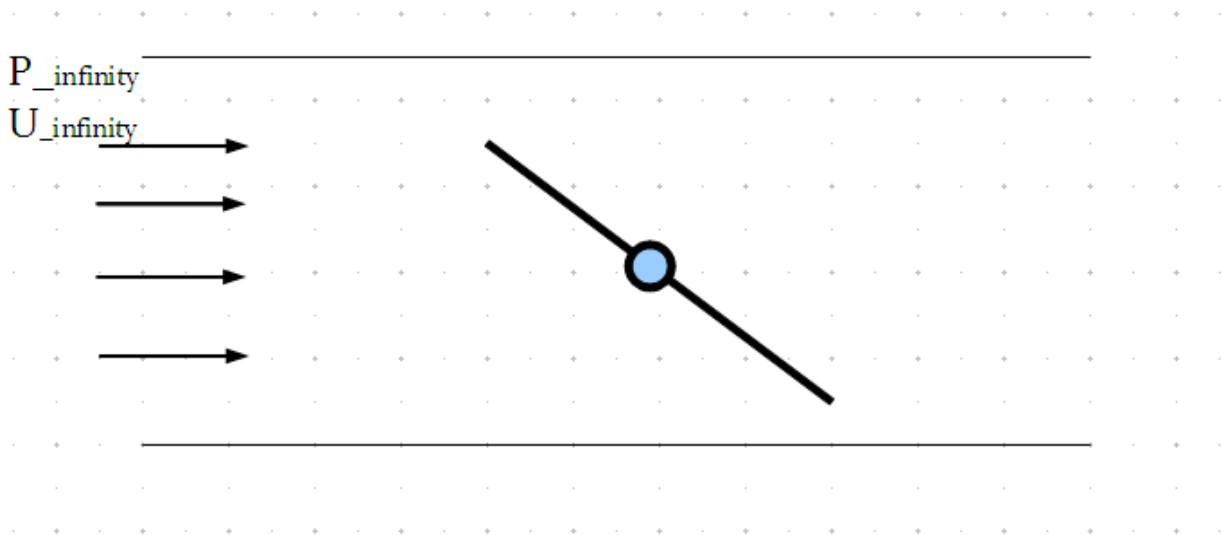
Abstract.....	2
Problem Description .....	2
Formulation.....	3
Assumptions & Consequences.....	3
Summary of Assumptions .....	4
Governing Equations.....	4
Boundary Conditions.....	5
Numerical Solution .....	6
Discretizing the Domain.....	6
Discretizing the Partial Differential Equation .....	7
Algorithm Overview .....	9
Computation of Velocities and Pressure .....	9
Results & Discussion .....	11
Appendix : MATLAB Program.....	15

## Abstract

Potential flow can sometimes be used as a good approximation for various problems. It can serve as a backup analytical solution which can be compared to, and one can intuitively relate with when solving otherwise complex flow problems. Potential flow does have its limitations and is accompanied with several assumptions and approximations that have to be explicitly stated and accounted for. Otherwise the solution would not make much sense.

## Problem Description

We are presented with the following description of the butterfly valve.



Given Information for the problem:

Far upstream,  $\vec{V} = U_{\infty}\vec{i} = \text{constant}$

$\rho_{air} = \text{constant}$

$P_{\infty} = \text{constant}$

The butterfly valve is placed at any angle in the channel.

Deliverables:

Flow field (information)

- $\psi$
- $\vec{V}$
- Pressure distribution

# Formulation

## Assumptions & Consequences

Since we use potential flow theory, we must incorporate various assumptions. The most important however is that flow is irrotational for potential flow. Since we are saying that there is no rotationality, this leads to being able to use formulas that assume irrotational flow, such as Bernoulli's equation between any 2 points. Before we discuss that, however, we must first list all the other critical assumptions.

We are assuming that this flow is steady. This assumption must be stated when using potential flow theory. Steady assumption gives way for the problem to be analyzed in a simpler manner, because the mass flow rate of flow coming in is equal to that going out over a given cross section. However, in cases where this is not true, one cannot assume steady flow and thus one cannot use Potential flow theory.

Since it is given in the problem statement that the density of air remains relatively constant, one can assume incompressible flow past the butterfly valve. Compressibility effects generally come into the picture when the velocities are close to 0.8 Mach. However, for all practical purposes, the flow is assumed to be well in the subsonic region and hence compressibility effects (though minor) are neglected.

One must assume that the problem is 2-dimensional, as modeled in the problem statement and required for simplistic analysis using potential flow.

Since the fluid is air, gravity (and gravitational effects) can be neglected since the density of air is relatively small, and the valve is not as large.

For purposes of analysis, the butterfly valve is assumed to be infinitely thin. This can never be the case in reality, but to simplify computations and get a good approximation, this assumption is used.

Finally, it is important to note that viscosity is neglected. The presence of viscosity would impose various conditions on the boundaries of the valve, which would require complex analysis (such as Navier Stokes). This is beyond the scope of potential theory, and hence viscosity must be neglected.

## Summary of Assumptions

- $\nabla \times \vec{V} = 0$  (Irrotational)
- $\mu = 0$
- Steady
- Incompressible,  $\rho_{air} = \text{constant}$
- Gravity neglected
- Infinitely thin butterfly valve

## Governing Equations

From the continuity equation in potential flow analysis, we have the following important equation for the stream function:

$$\nabla^2 \psi = 0 \quad \text{Equation [1a]}$$

When expanded, this results in the following:

$$\frac{\delta^2 \psi}{\delta x^2} + \frac{\delta^2 \psi}{\delta y^2} = 0 \quad \text{Equation[1b]}$$

When using potential flow, appropriate definitions need to be made. These are as follows:

$$u = \frac{\delta \psi}{\delta y} \quad \text{Equation[2]}$$

$$v = -\frac{\delta \psi}{\delta x} \quad \text{Equation[3]}$$

Now, one can integrate the freestream velocity from 0 to height  $y$ , from equation [2]. This would give us the following relationship:

$$\int_{y=0}^y U_{\infty} dy = \int_{\psi=0}^{\psi(y)} U_{\infty} dy$$
$$\therefore \psi(y) = U_{\infty} y \quad \text{Equation[4]}$$

Finally, one can apply Bernoulli's Equation (for any 2 points because of irrotational assumption)

$$\frac{P_{\infty}}{\rho_{\infty}} + \frac{v_{\infty}^2}{2} = \frac{P}{\rho_{\infty}} + \frac{v^2}{2} \quad \text{Equation[5]}$$

## Boundary Conditions

One may get a better understanding of the application of these governing equations, and then the boundary conditions by simply looking at the diagram (given). Figure [2] attempts to visualize the boundary conditions that are to be imposed for applying the governing equations

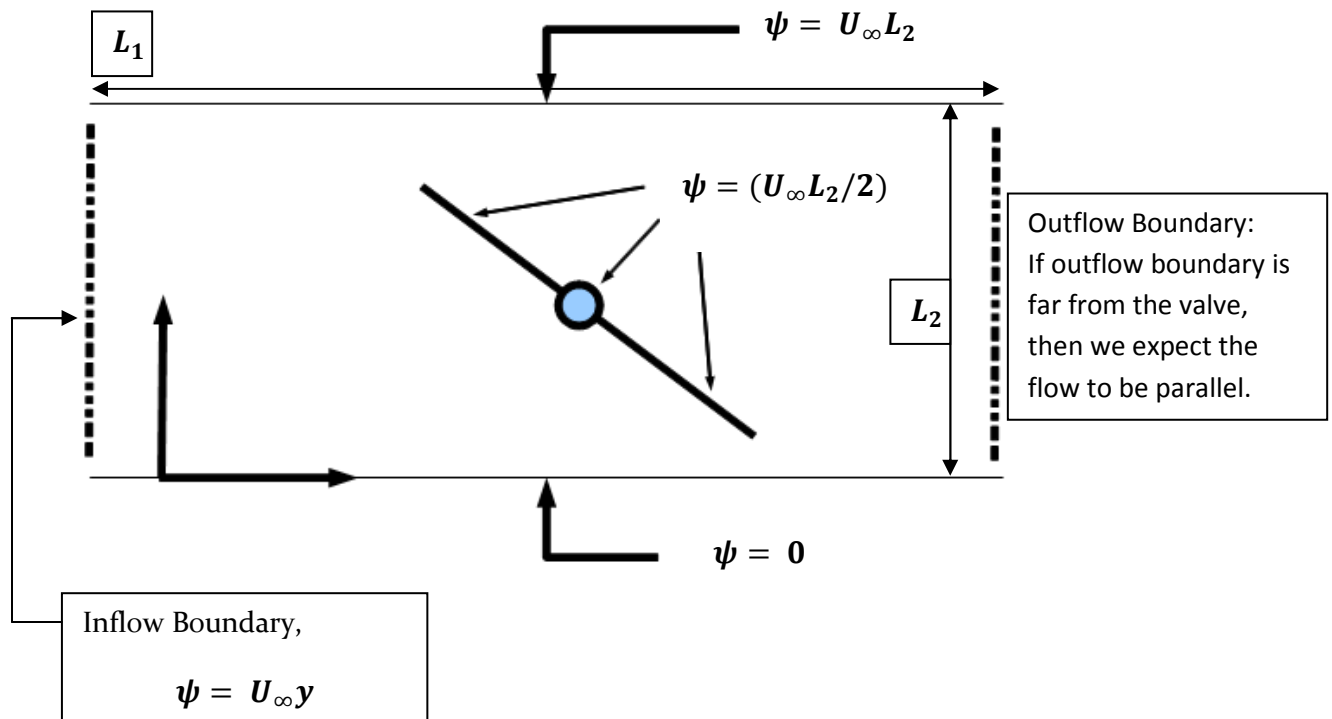


Figure [2] : Boundary Conditions

From equation [4], it is possible to evaluate the stream function at the midpoint of the valve. This shall indeed be the stream function all along the top and bottom surfaces of the valve as depicted in figure [2]. This leads to:

$$\psi = (U_\infty L_2 / 2) \text{ on the valve.}$$

Furthermore, at  $y = 0$  we have

$$\psi = 0$$

Also at  $y = L_2$ , we have  $\psi = U_\infty L_2$

We note that the length of the valve is also  $L_2$ , so that when the valve is perpendicular to the  $x$  axis, the flow is completely stopped.

## Numerical Solution

### Discretizing the Domain

Using potential flow, one can model this problem with the aid of a computer program. In order for us to do this, we must first approach this problem (including the governing equations discussed earlier) from a purely numerical standpoint. This will enable us to develop an algorithm that can be used towards simulating this problem on the computer.

Using the finite difference method approach, one can solve this problem using the following steps:

1<sup>st</sup> Step : Discretize the domain

2<sup>nd</sup> Step : Discretize the Partial Differential Equation

3<sup>rd</sup> Step: Solve resulting equations

In order to discretize the domain, one must generate a grid, along which the valve rests and the intersections of the grid would coincide with the grid. This process is shown in figure [3].

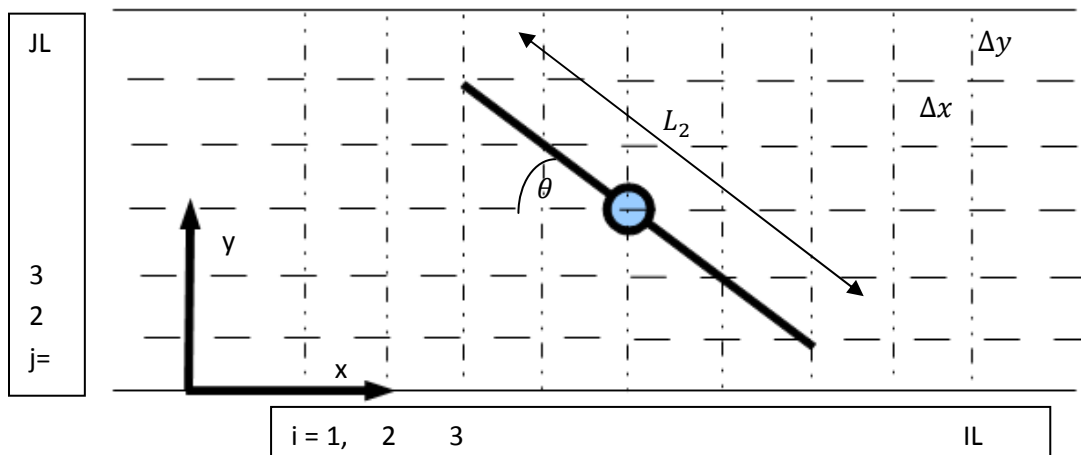


Figure [3] : Discretizing the domain: Grid generation

Now we derive the grid point separation relationship and illustrate explicitly how one can generate the above grid.

$$\Delta y = \frac{L_2}{JL-1}$$

From the grid drawn above, we get the equation the vertical spacing between the grids. The number of grids must be subtracted by 1 due to the fact that the zero j grid starts at 1.

To ensure that the valve meets the grid along the intersections, one must define  $\Delta x$  as follows, using the relationship:

$$\Delta x = \cot(\theta)\Delta y$$

This is rather evident from the sketch of the angle with respect to the grid points.

## Discretizing the Partial Differential Equation

### Gauss Siedel Method

The Gauss Siedel method employs the technique of solving the left hand side of the equation (matrix) by operating on the right hand side using previously known values for the same.

This is an iterative process, and is used in this scenario to find the stream function around the butterfly valve. This process would make the solution converge towards the end of the iteration, and further techniques will be employed to allow the solution to have the desired degree of accuracy.

The Taylor series approximation allows us to break a second order partial differential equation into finite iterations, to be able to numerically and iteratively converge to give a local approximation of the solution. This is particularly useful for this problem, and this technique combined with the G-S method is employed in order to find the solution. We can thus derive equation [6] from equation [1b], and this allows us to put the partial derivate into our iterative algorithm.

$$\frac{\psi_{i+1,j} - 2*\psi_{i,j} + \psi_{i-1,j}}{\Delta x^2} + \frac{\psi_{i,j+1} - 2*\psi_{i,j} + \psi_{i,j-1}}{\Delta y^2} = 0 \quad \text{Equation [6]}$$

One can then insert equation [6] into an algorithm to be able to solve for  $\psi_{i,j}$  and derive the stream function around the butterfly valve.

### Successive Over Relaxation

The SOR is used with Gauss Siedel in order to maintain accuracy, and reduce error to set tolerance limit. The SOR basically speeds this process up in order to save computational time. The iteration scheme used for this particular problem is discussed in the coding

section of the report. However, we note that the algorithm for general cases for using SOR is as follows. The parameter  $\omega$  is employed, and checks for convergence between the old matrix and new matrix by iterating through each new term and checking with the old one.

$N(x,y) = \omega N(x,y) + (1 - \omega) * M(x,y)$  where  $N(x,y)$  is the new value, and  $M(x,y)$  the old.

### ***Tolerance & Error***

In order for this algorithm to be complete, one must specify when to stop iterating. This is done by setting the tolerance factor, and basically saying that when the solution has converged to a point where the relative error is near the machine zero, then we can stop iterating. Machine zero is identified to be  $1 \times 10^{-7}$ , and this does mean that the solution obtained will have a certain degree of error. However, for engineering applications, this error needs to be critically set to as little as possible without hurting the time budget allotted for solving a particular computation. Hence, one uses the criteria of relative error, in conjunction with absolute error in order to check for convergence and stop iterating the algorithm.

$$\mathbf{Error} = |(x_i^{m+1} - x_i^m) / (x_i^{m+1})| \qquad \mathbf{Equation [7]}$$

A complete description of the algorithm, the numerical methods, and the process is described illustratively in figure [4].

## Algorithm Overview

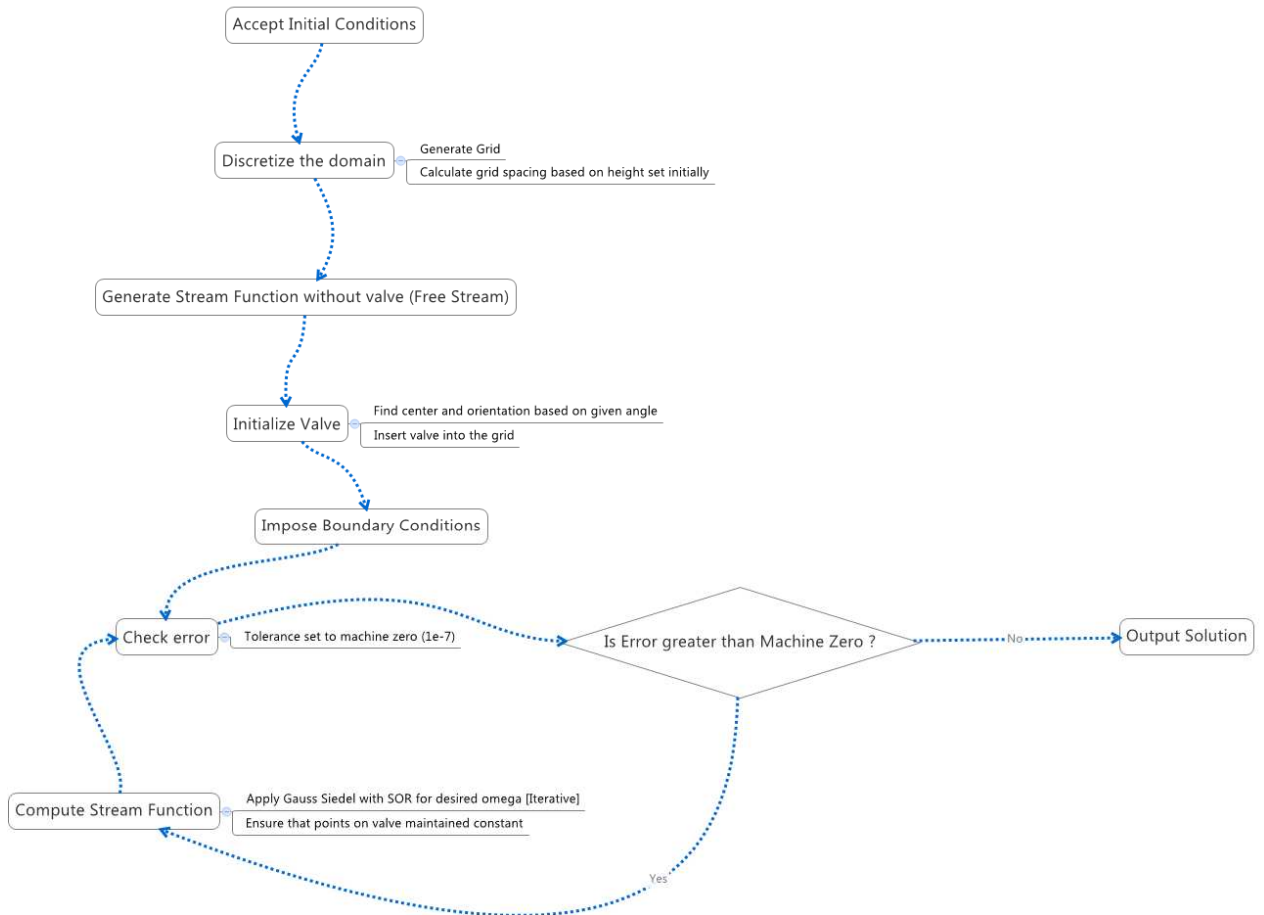


Figure [4] : Overview of the algorithm to get the desired stream function

## Computation of Velocities and Pressure

Using equations [2] and [3], one can write the numerical solution for the components of velocity as follows:

$$u_{i,j} = \frac{\psi_{i,j+1} - \psi_{i,j-1}}{2\Delta y}$$

Furthermore,

$$v_{i,j} = \frac{\psi_{i+1,j} - \psi_{i-1,j}}{2\Delta y}$$

This uses gives a reasonable approximation for the local flow field around the valve. For finding pressure, we make use of the fact that we are assuming irrotational flow, and thus Bernoulli's equations (irrotational) are valid. Thus, we employ equation 5:

$$\frac{P_\infty}{\rho_\infty} + \frac{V_\infty^2}{2} = \frac{P}{\rho_\infty} + \frac{V^2}{2} \quad \text{Equation[5, repeated]}$$

## Results & Discussion

First of all, assuming that the angle of rotation is  $\pi/4$  and the length  $L_2$  is arbitrary, we can generate the grid and get a stream function for free stream velocity of 19 (arbitrary). These inputs are used in order to explain the solution, they can be set to any desired value later on.

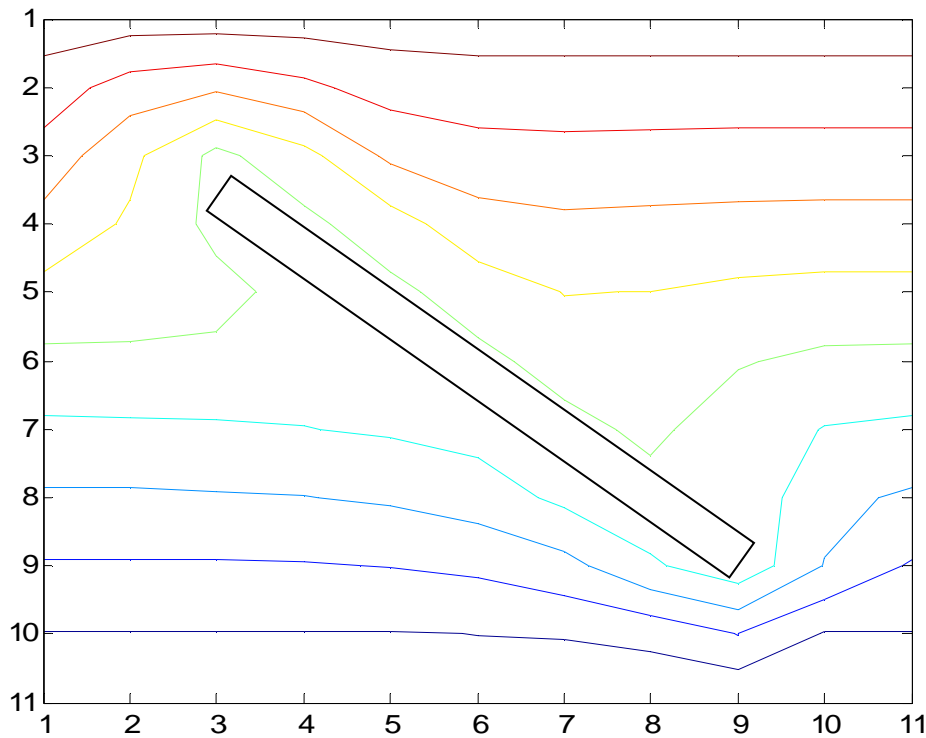
190	190	190	190	190	190	190	190	190	190	190
171	171	171	171	171	171	171	171	171	171	171
152	152	95	152	152	152	152	152	152	152	152
133	133	133	95	133	133	133	133	133	133	133
114	114	114	114	95	114	114	114	114	114	114
95	95	95	95	95	95	95	95	95	95	95
76	76	76	76	76	76	95	76	76	76	76
57	57	57	57	57	57	57	95	57	57	57
38	38	38	38	38	38	38	38	95	38	38
19	19	19	19	19	19	19	19	19	19	19
0	0	0	0	0	0	0	0	0	0	0

The valve is then set in place, and the new free stream function is computed with the valve inside. The basic form of this is as follows:

190.0000	190.0000	190.0000	190.0000	190.0000	190.0000	190.0000	190.0000	190.0000	190.0000	190.0000	190.0000
171.0000	150.6041	143.8055	155.4813	168.2092	171.4712	171.8049	171.4661	171.1576	171.0223	171.0000	171.0000
152.0000	124.8055	95.0000	113.4075	144.0868	152.4516	153.6804	153.0852	152.3979	152.0594	152.0000	152.0000
133.0000	117.4813	94.4875	95.0000	111.4465	132.5239	136.3957	135.4960	134.0017	133.1585	133.0000	133.0000
114.0000	111.2092	106.0868	92.4465	95.0000	109.3347	120.5593	119.6543	116.5127	114.4227	114.0000	114.0000
95.0000	95.4712	95.4516	94.5239	90.3347	95.0000	106.8371	107.5655	101.2779	96.1272	95.0000	95.0000
76.0000	76.8049	77.6804	79.3957	82.5593	87.8371	95.0000	103.2302	91.6138	79.0059	76.0000	76.0000
57.0000	57.4661	58.0852	59.4960	62.6543	69.5655	84.2302	95.0000	95.6309	65.0156	57.0000	57.0000
38.0000	38.1576	38.3979	39.0017	40.5127	44.2779	53.6138	76.6309	95.0000	59.3750	38.0000	38.0000
19.0000	19.0223	19.0594	19.1585	19.4227	20.1272	22.0059	27.0156	40.3750	19.0000	19.0000	19.0000
0	0	0	0	0	0	0	0	0	0	0	0

The valve can be seen numerically set in the matrix at the constant stream function values.

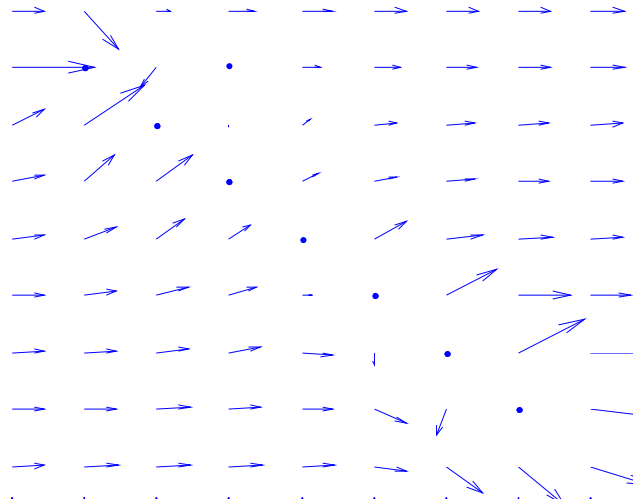
One can then take this stream function, and output it as a contour plot. This will be an attempt to visualize the streamlines and better understand the solution. Figure [5] shows this contour plot.



**Figure [ 5 ] : Contour Plot of the Stream Function and Valve**

From this stream function itself, one can identify the two stagnation points on this valve. The streamlines generated by the program concur with what we expect theoretically. Thus, it proves that the solution generated serves as a reasonable approximation, and till a certain point, it is indeed acceptable. One must remember that we have assumed irrotational. This will mean that beyond a certain point, we can expect that the solution will fail to work because in reality the flow should separate. This will never be the case with potential flow, and hence it is important to use other methods to get a more exact (& realistic) solution.

We can now proceed to see the vector diagram, figure[6] obtained for velocity for this particular problem, and then use this to get pressure everywhere (using Bernoulli's equation).



**Figure [6] : Vector Plot of Velocity**

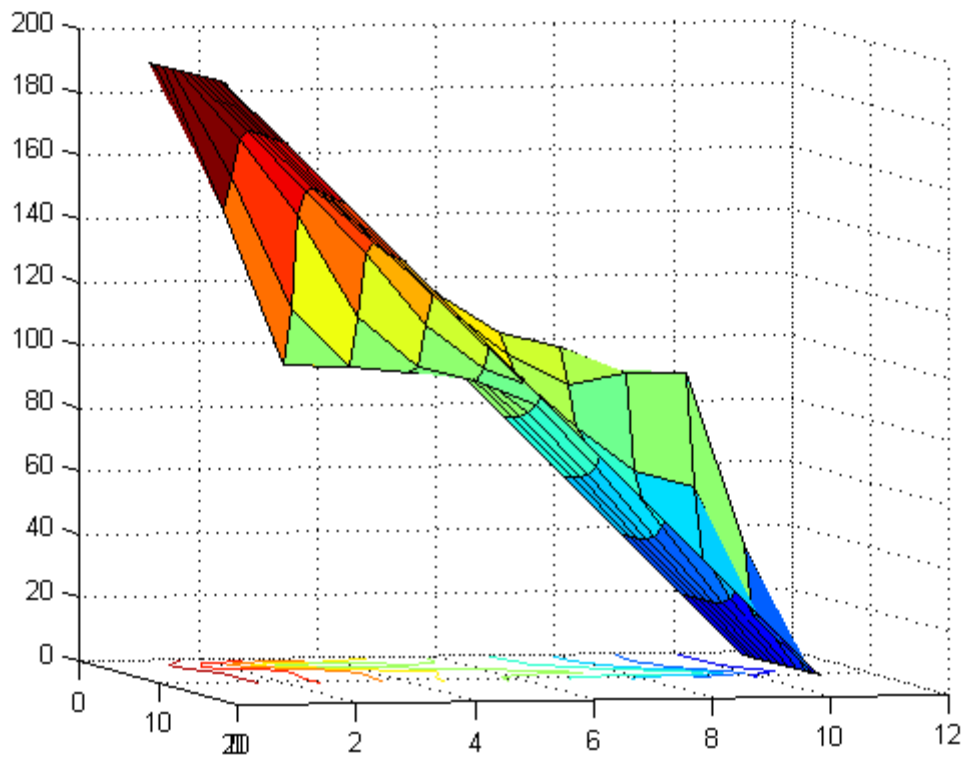
One can see the flow around the valve from this vector plot, and see how the flow goes around the butterfly valve. The stagnation points will have zero velocity, and so will the all the points along the physical valve itself (on the grid).

This can be used in conjunction with Bernoulli's equation in order to find the stagnation pressure(s).

The pressure around the valve can be seen in the pressure matrix obtained:

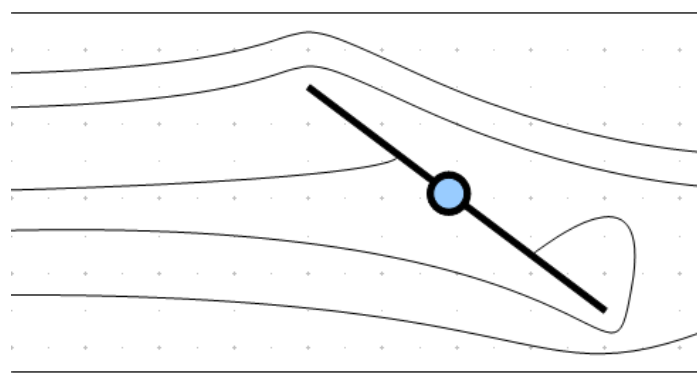
0.9956	0.9902	0.9945	0.9992	1.0000	1.0002	1.0001	1.0000	1.0000
0.9942	1.0018	0.9916	0.9974	1.0000	1.0003	1.0002	1.0001	1.0000
0.9979	0.9998	1.0018	0.9960	0.9995	1.0006	1.0005	1.0002	1.0000
1.0000	0.9995	1.0014	1.0018	0.9992	1.0009	1.0010	1.0005	1.0001
1.0001	1.0002	1.0003	1.0016	1.0018	1.0012	1.0016	1.0012	1.0003
1.0000	1.0001	1.0002	1.0003	1.0005	1.0018	1.0018	1.0018	1.0007
1.0000	1.0000	1.0001	1.0000	0.9996	0.9979	1.0018	1.0002	1.0015
1.0000	1.0000	1.0000	1.0000	0.9998	0.9984	0.9920	1.0018	0.9959
1.0000	1.0000	1.0000	1.0000	1.0000	0.9999	0.9994	0.9905	1.0000

Ofcourse, a surface plot of the stream function will better illustrate the stagnation points on the valve. This is shown in figure [7].



**Figure [7] : Surface Plot of Stream Function to visualize stagnation points and pressure distribution**

The program explains all the steps used to replicate the necessary outputs. However, the final diagram that summarizes this flow field is given below:



**Figure [8] : Potential Flow around Butterfly Valve**

## Appendix : MATLAB Program

```
%% Initializing constants

u_inf = 19; %Free Stream Velocity defined

theta = pi/4; %Defining the angle of rotation of the valve

L2 = 10; %Setting the height L2 of the pipe

JL = 11; %Setting grid points in the y direction

IL = 11; %Setting grid points in the x direction

OM = 1.5; %Setting omega to be desired tolerance value (used later)

P_inf = 100000; %Free Stream Pressure

row = 1; %Air Density = constant

%% Discretize Domain -> Grid Generation

deltay = L2/(JL-1); %Setting the Delta_y to be the height divided by the number of grid points

deltax = cot(theta)*deltay; %Using trigonometry, set deltax to relate to theta and deltay

valveJC = (JL)/2; %Set the middle y value of the valve to be at the center of grid

valveIC = (IL)/2; %Set middle x value of valve to be at grid center.

valve = zeros(JL); %Initialize a matrix of zeros for the valve

v = zeros(11)

for I = 1:IL

    for J = 1:JL

        XI1(J,I) = abs(u_inf*(J-JL)); %Set initial XI function

        u(I,J) = u_inf; %Set freestream velocity everywhere
```

```

end

end

%% Initialize Valve inside the generated grid

numy = (JL*sin(theta))/deltay; %Number of grid points in y direction covering the valve
numx = (JL*cos(theta))/deltax; %Number of grid points in x direction covering the valve

%Finding the top right corner of the valve to start
I = floor(valveIC - numx/2 + 2);
J = floor(valveJC - numy/2 + 2);

%Going from the top right corner to the bottom right corner of valve
while I ~= floor(valveIC + numx/2 + 1)
    while J ~= floor(valveJC + numy/2 + 1)

        XI1(I,J) = (u_inf*L2)/2; %Set uniform XI on the valve
        valve(I,J) = 1; %Generate points that represents the valve

        I = I + 1; %Increment i
        J = J + 1; %Increment j

    end
end

end

```

```
%% Finding Solution & Adjusting
```

```
%Initially set the new XI matrix to the old one
```

```
XI2 = XI1;
```

```
%Initialize error variable
```

```
error = 1 ;
```

```
%Initialize dummy matrix
```

```
XI4 = XI2;
```

```
%Machine zero is  $10^{-7}$ , hence we want to achieve that level of accuracy.
```

```
while error > 1e-7;
```

```
    XI2 = XI4; %Set dummy variable as new XI2
```

```
    for m = 1: 10000 %The higher this value, the more accurate XI will become
```

```
        for I = 2:IL-1
```

```
            for J = 2:JL-1
```

```
                %When the valve is not present
```

```
                if valve(I,J) == 0
```

```
                    %Gauss Siedel - Method for computation of XI
```

```
                    XI2(I,J) = 0.5*((XI2(I,J+1) + XI1(I,J-1))*deltax^2 + deltay^2*(XI2(I+1,J) + XI1(I-1,J)))/(deltax^2+deltay^2);
```

```
                    %Successive OverRelaxation technique to get accuracy
```

```
                    XI2(I,J) = OM*XI2(I,J) + (1-OM)*XI1(I,J);
```

```
                    %Tolerance level setting
```

```
                    error = abs((XI2(I,J) - XI1(I,J))/XI2(I,J))
```

```
                else
```

```

        end

    end

end

XI4 = XI2; %Replace with dummy matrix

end

end

%Finally set new XI matrix after final iteration

XI2 = XI4;

%% Computation of velocity, and pressure.

for I = 2:IL-1

    for J = 2: JL-1

        %When valve is not present

        if valve(I,J) == 0

            u(I,J) = (XI2(I,J+1) - XI1(I,J-1))/(2*deltay); %u velocity

            v(I,J) = (XI2(I+1,J) - XI1(I-1,J))/(2*deltax); %v velocity

            %Pressure found using Bernoulli between any 2 points

            P(I,J) = P_inf + row*u_inf^2/2 - row*((u(I,J))^2 + (v(I,J))^2)/2 ;

        else

            %On the valve itself

            u(I,J) = 0;

            v(I,J) = 0;

            P(I,J) = P_inf + row*u_inf^2/2;

        end

    end

end

```

```
end
```

```
end
```

```
end
```

```
%% Output Flow Field as Contour Plot
```

```
XI3 = XI2;
```

```
for I = 1:IL
```

```
    for J = 1:JL
```

```
        XI3(J,I) = XI2(I,J);
```

```
    end
```

```
end
```

```
% contour(XI3');
```