Introduction to the Numerical Solution of the Navier-Stokes equations.

Module 12: Pressure recovery-schemes and energy visualization

Manel Soria

Assignatura: Aerodinàmica, Mecànica Orbital I Mecànica de Vol
Titulació: Master d'Enginyeria Aeronàutica
Curs: 1r
Escola Superior d'Enginyeries Industrial, Aeroespacial i Audiovisual de Terrassa (ESEIAAT)
Idioma: Anglès
Novembre 2021

UNIVERSITAT POLITÈCNICA
DE CATALUNYA
BARCELONATECH

Introduction to the Numerical Solution of the Navier-Stokes equations

Module 12. Pressure recovery-schemes and energy visualization

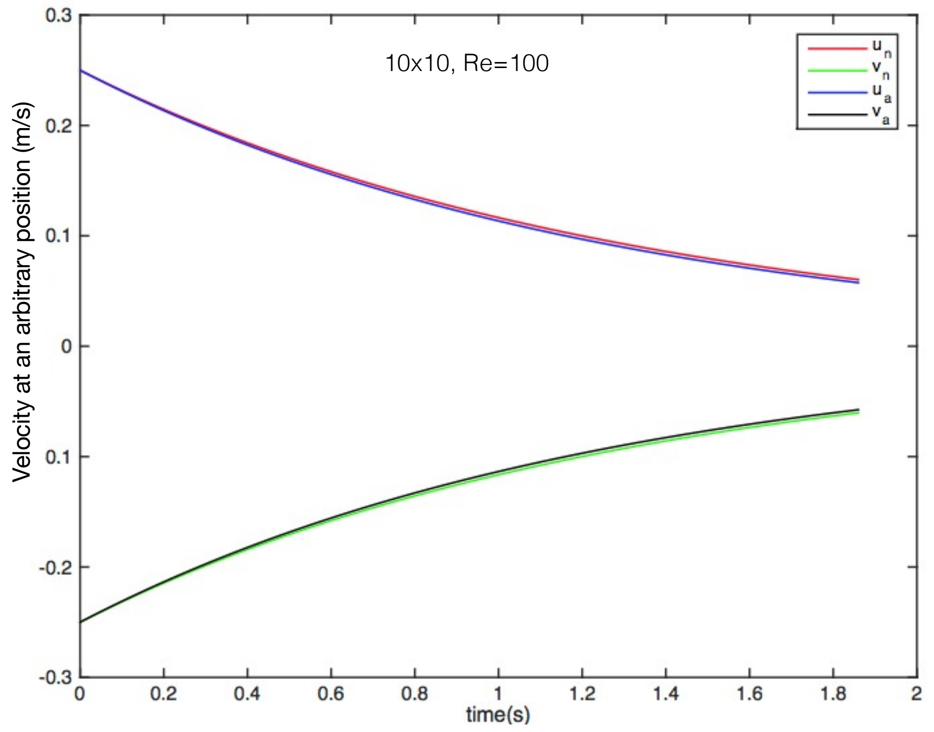
Manel Soria – manel.soria@upc.edu

MÀSTER UNIVERSITARI EN ENGINYERIA AERONÀUTICA
ESEIAAT

Module 12. Pressure recovery-schemes and energy visualization

13.1 Pressure recovery and comparison with analytic results

In last module we compared analytic and numeric velocities but not pressure.



The numeric pressure should also be similar to the analytic result, but there is an important difference that we have to take into consideration: the incompressible flow equations leave the pressure under determined to a constant.

If p is the solution of the equations, then $p + C$ also solve them. This under determination is inherited by the discrete Poisson equation as we saw in Module 9.

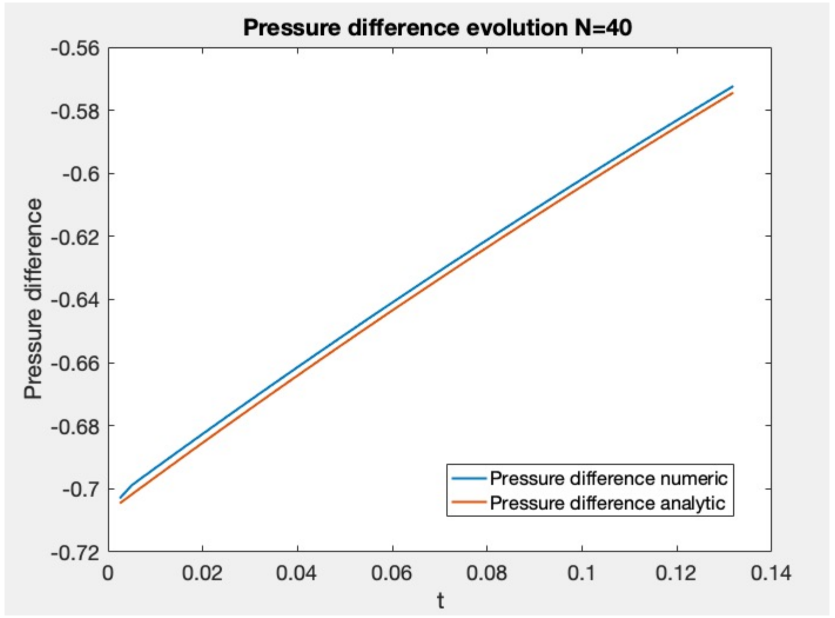
Let's see how can we obtain the numeric pressure and compare it with the analytic solution.

$$\boxed{\frac{\partial u_j}{\partial x_j} = 0 \qquad \frac{\partial u_i}{\partial t} + u_j \frac{\partial u_i}{\partial x_j} = \boxed{-\frac{1}{\rho}\frac{\partial p}{\partial x_i}} + \nu \frac{\partial^2 u_i}{\partial x_j \partial x_j}}$$

13.1 Pressure recovery and comparison with analytic results

Recall that the solution of the Poisson equation is not the pressure p but a pseudo-pressure $\tilde{p}$.
The numeric pressure can be obtained from Eq. 9.5.

$$p^{n+1} = \frac{\rho}{\Delta t}\tilde{p} \qquad (9.5)$$



Due to the indetermination, the **numerical pressure** has no relation with the **thermodynamic pressure**. For instance, it can be negative. Different codes *can produce* different pressure distributions, depending on how the Poisson equation has been solved. Moreover, the arbitrary constant can be different for each time step.

However, the difference between the pressure at two points *has to be the* same for all the codes and for the analytic solution. Thus, the best way to check our code is to compare the evolution of the pressure difference between two points.

In the plot, the pressure at an arbitrary mesh location (3,3) minus the pressure at an arbitrary reference location (13,7) has been represented, for both the numerical and the analytic solutions.

Recall that the analytic pressure has to be obtained, as a function of the position and instant (Eq. 2.2).

As the mesh density increases, the difference has to tend to zero.

CFD codes often give us different options for the "*numeric scheme*" to evaluate the convective term.
The approximation selected can have a large influence on the results obtained, as we will see in next slides.
Recall the discretization of the integral of the convective term in Module 2.

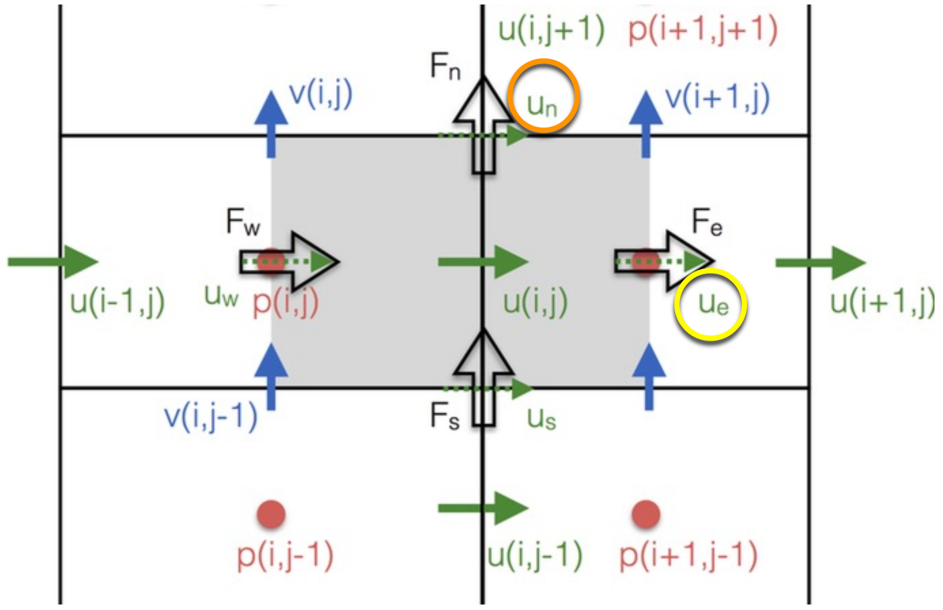
$$\int_V \nabla \cdot (u_1 \boldsymbol{u}) dV \approx u_e F_e - u_w F_w + u_n F_n - u_s F_s$$

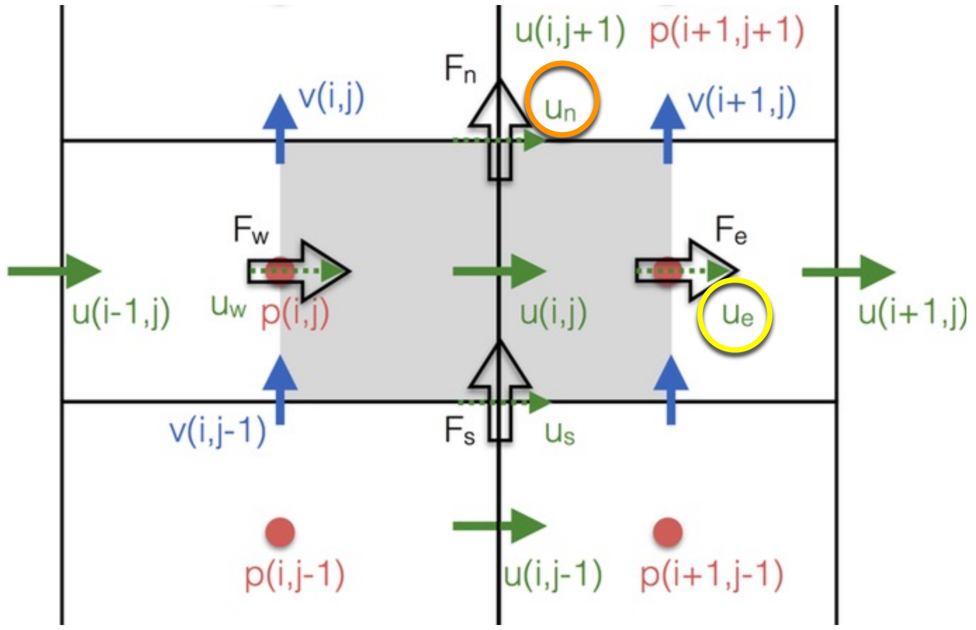
We choose a second order approximation scheme where each face velocity is the average of the adjacent nodal values:

$$u_e = \frac{u_{i+1,j} + u_{i,j}}{2}$$

$$u_n = \frac{u_{i,j+1} + u_{i,j}}{2}$$

Now, we will implement an **Upwind** scheme where the face velocity is assigned to upstream value.





In the ***upwind approximation***, the face velocity is assumed to be equal to the nodal value located upstream.

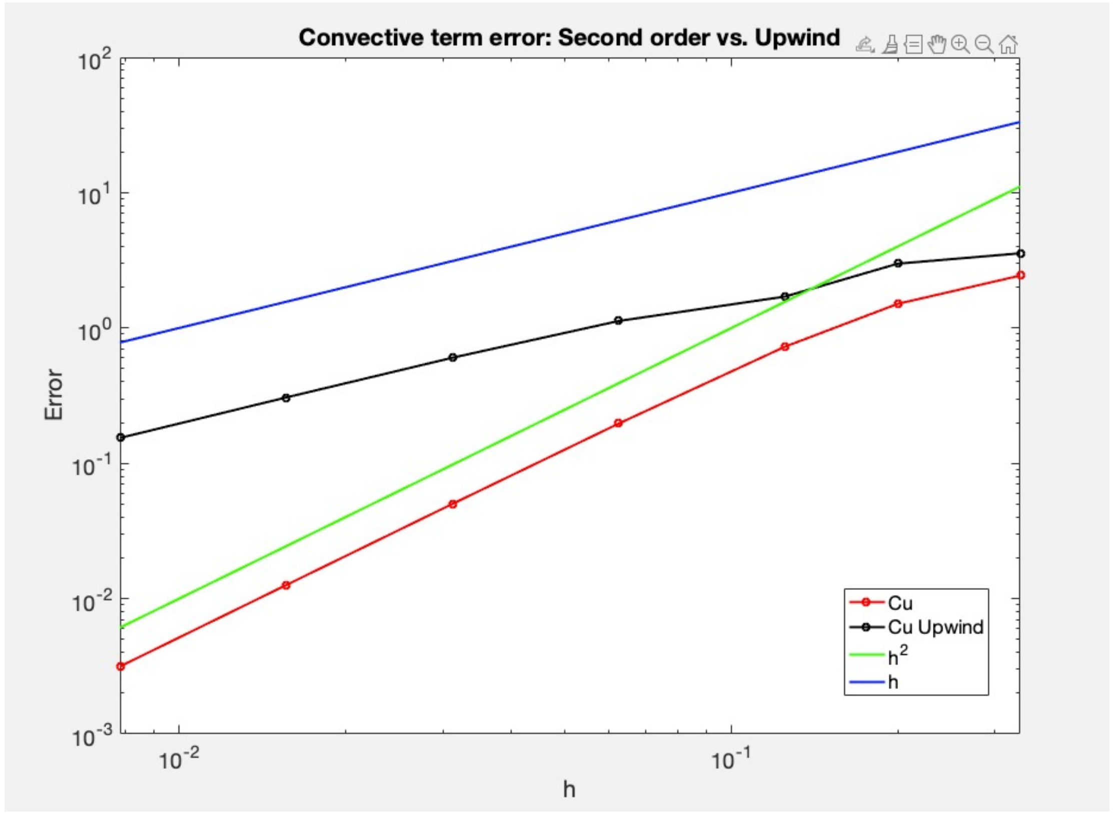
For instance, if $F_e > 0$ then $u_e = u(i,j)$, but if $F_e \leq 0$ (meaning that the flow is directed to the left), then $u_e = u(i + 1,j)$. The same holds for the velocities.

The flow terms are evaluated as before. The same holds for the diffusive term.

The difference in the code is small, but the results obtained are very different:

- They are less accurate (as upwind is a first order approximation).
- They dissipate kinetic energy, making the code more stable.

If the convergence error is analyzed as we did in M6, we see that the Upwind scheme is only first order accurate:

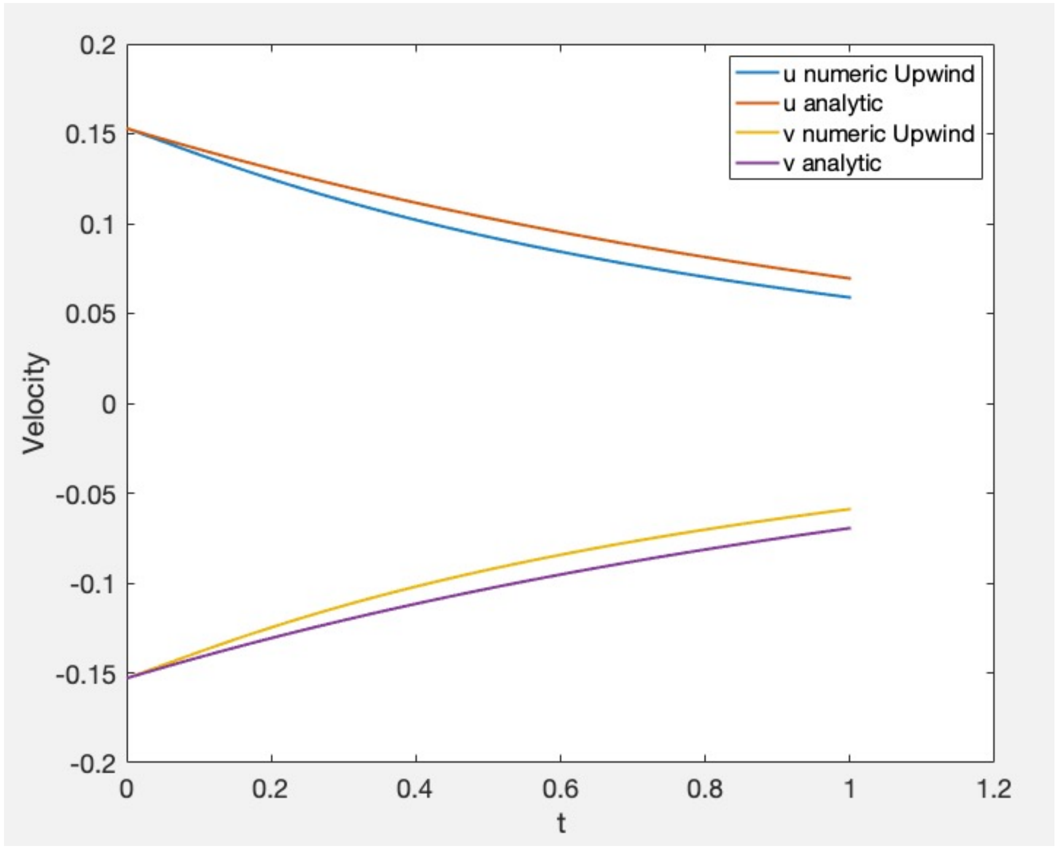


13.2 Numeric schemes and kinetic energy

When the integration of the Navier-Stokes is carried out with upwind, we can see that the error respect to the analytic solution is larger.

In this plot, both components of the velocity are compared with the analytic solution for N=60, at an arbitrary mesh position.

The numeric velocity decays faster than the analytic for both u and v and this is not a coincidence: the upwind scheme dissipates kinetic energy.



The specific kinetic energy of the fluid (per mass unit) is:

$$e = \frac{1}{2}|\mathbf{u}|^2 = \frac{1}{2}(u^2 + v^2)$$

It is a function of the position and the time. We are interested in the time evolution of the total kinetic energy K

$$K = \int_V e dV$$

It is only a function of time. For the Taylor solution (Eq. 2.2), the distribution of velocity as a function of time is known and thus we can obtain an expression for K.

$$\boxed{\begin{aligned} u &= FU_0 \cos\left(\frac{2\pi x}{L}\right)\sin\left(\frac{2\pi y}{L}\right) \\ v &= -FU_0 \sin\left(\frac{2\pi x}{L}\right)\cos\left(\frac{2\pi y}{L}\right) \qquad \text{[Eq. 2.2]} \\ F &= \mathrm{e}^{-\frac{8\nu t\pi^2}{L^2}} \end{aligned}}$$

In our domain,

$$K = \frac{1}{2}\int_0^L \int_0^L u^2 + v^2 dxdy$$

The integral can be done by hand as in the old times or using a symbolic calculation tool. With *Matlab*, once u, v and F have been defined (see module 2), we write:

```
% Let's find the total kinetic energy
e=(1/2)*(u^2+v^2);
K=int(int(e,x,0,L),y,0,L);
```

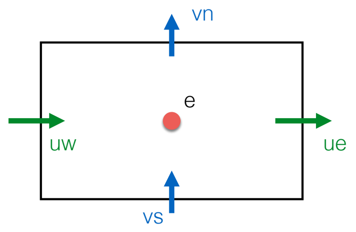
And we get the result:

$$K = \frac{L^2 {U_0}^2 \mathrm{e}^{-\frac{16\nu\pi^2 t}{L^2}}}{4}$$

The total kinetic energy decreases with time, as expected. The larger the viscosity, the faster it decays.

If a more detailed analysis is carried out, it can be seen that only the diffusive term in the momentum equation dissipates kinetic energy, but not the convective nor the pressure gradient terms.

Now, we will compute the numeric kinetic energy. For each main mesh control volume, it is usually evaluated as:



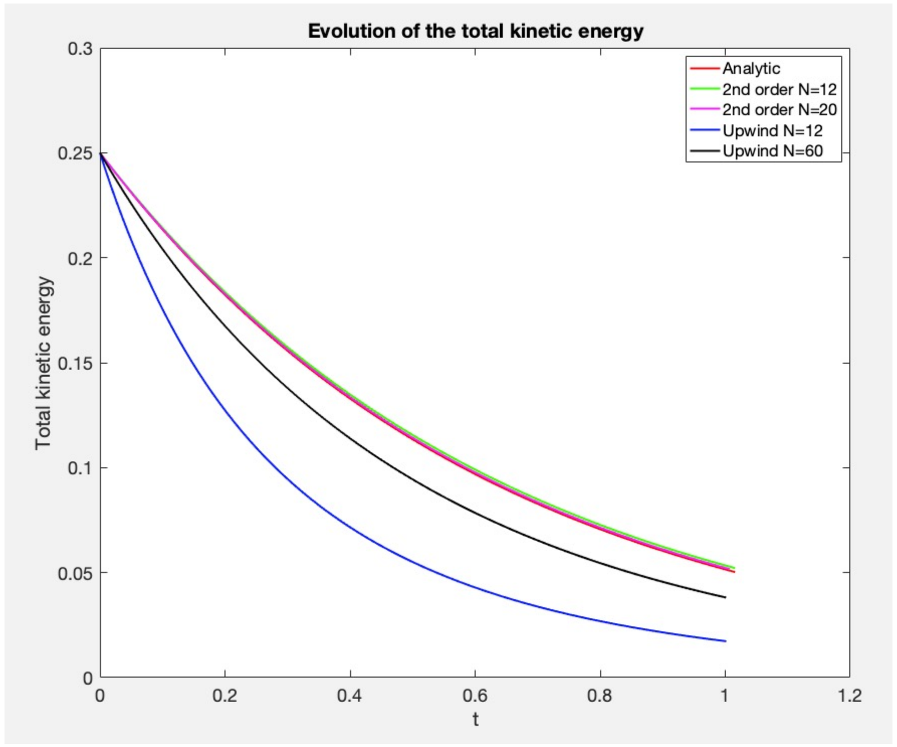
$$e = \frac{1}{2} V (u_e^2 + v_n^2)$$

Where V is the volume of the control volume.

It can also be defined in terms of the interpolated velocity :

$$e = \frac{1}{2} V \left[\left(\frac{u_e + u_w}{2} \right)^2 + \left(\frac{v_n + v_s}{2} \right)^2 \right]$$

However, the first definition is better to study the properties of the numerical schemes.



As it can be seen in the plot, our original 2nd order numerical scheme yields a very accurate value for the total kinetic energy, even for very coarse meshes. Actually, it can be proved that the convective term does not add or dissipate kinetic energy at all.

On the other hand, the upwind discretization, even for a quite fine mesh, underestimates the total kinetic energy. It can also be proved that the upwind dissipates kinetic energy.

Of course, for a sufficiently fine mesh, both schemes would converge to the analytic solution.

From the previous plots it would seem that upwind is always a bad idea. This is not true, there are good reasons to use it in many cases.

To conclude this section, keep in mind that there are many more numerical schemes for the convective term, and that this is a very important aspect of CFD.

Let's see how can we generate a graphic representation of our velocity fields.

Usually, special software is used to do so, such as the code *Paraview*. Then, the CFD codes only have to generate a data file in a format that can be read by the visualization software.

However, in the spirit of this course (building everything from scratch) we will develop some simple tools to visualize the flow field.

We will cover two approaches:

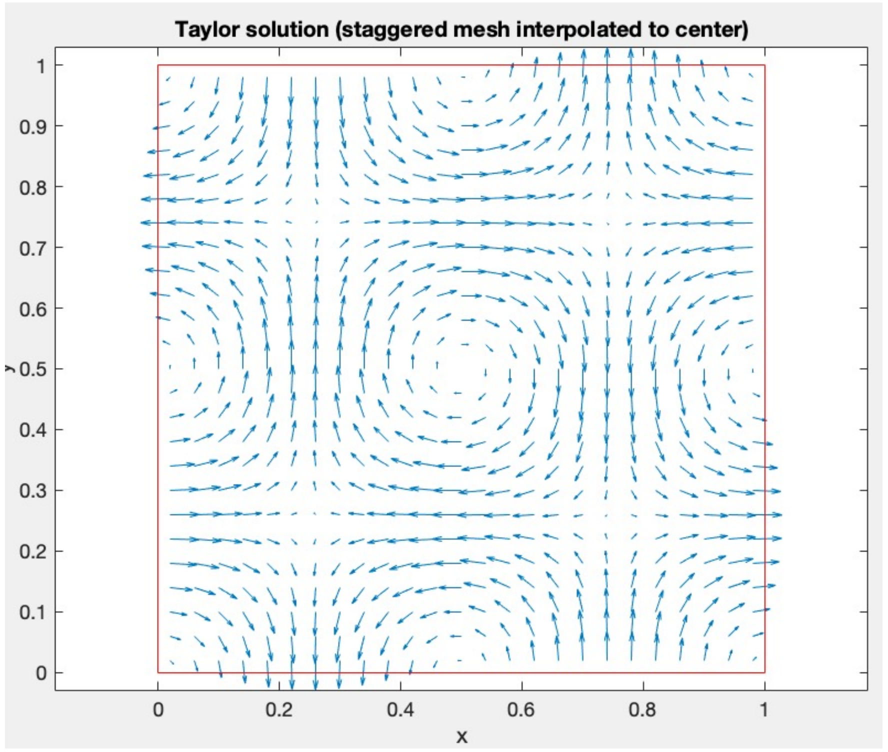
Quiver plot - A plot of the velocity vectors in the form of arrows

Streamline plot - A streamline is a curve tangent to the instantaneous velocity vector in each point.

The quiver plot is very simple: for each field position the velocity vector is plotted as an arrow. The scale of the arrows has to be adjusted to avoid them overlapping.

Before doing the plot, both components of the velocity field (that are staggered) have to be interpolated to the center of the cells.

In *Matlab*, the command *quiver* plots all the arrows together, but we have to be careful to order our data in the way it expects it.



Here you have an example of the Taylor flow (Eq. 2.2) result obtained for N=25.

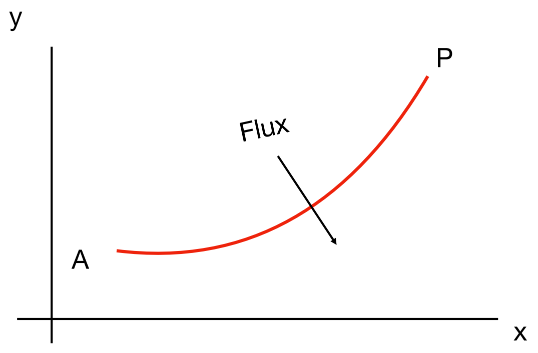
13.3 Visualization. Streamline

A streamline is a curve tangent to the instantaneous velocity vector in each point.

They should not to be confused with pathlines. A pathline is the trajectory that a very small particle inside the flow would follow. For a steady flow, pathlines are also streamlines.

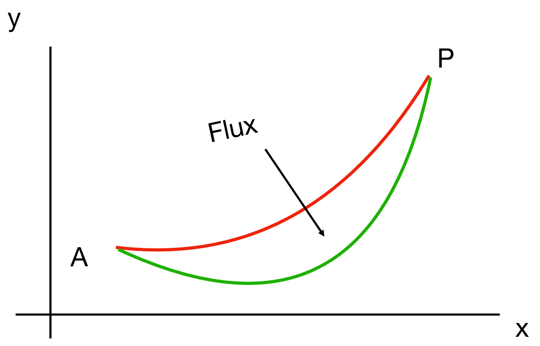
To generate the streamlines or pathlines, the most general way is to use numerical integration. However, for the special case of a two-dimensional flow, we can use the stream function to do so.

For a two-dimensional **incompressible** flow, the stream function ψ is defined as follows



Given two points A and P and a curve that joins them, the stream function is the volume flow through the curve, that can be calculated as the integral of the dot product between the velocity vector (u, v) and the normal to the curve element (dx, dy), that is $(dy, -dx)$:

$$\psi = \int_A^P udy - vdx$$



If the curve is changed from the red line to the green line (or any other line), the stream function remains constant, as the flux that crosses both lines has to be the same (otherwise mass would accumulate in the area between the lines, and this is not allowed in an incompressible flow).

If the point P is infinitesimally shifted $(\delta x, \delta y)$, this results in a change of the stream function. We can write this as:

$$\delta\psi = u\delta y - v\delta x$$

The differential of ψ is:

$$\delta\psi = \frac{\partial\psi}{\partial x}\delta x + \frac{\partial\psi}{\partial y}\delta y$$

Thus,

$$u = \frac{\partial\psi}{\partial y} \qquad v = -\frac{\partial\psi}{\partial x}$$

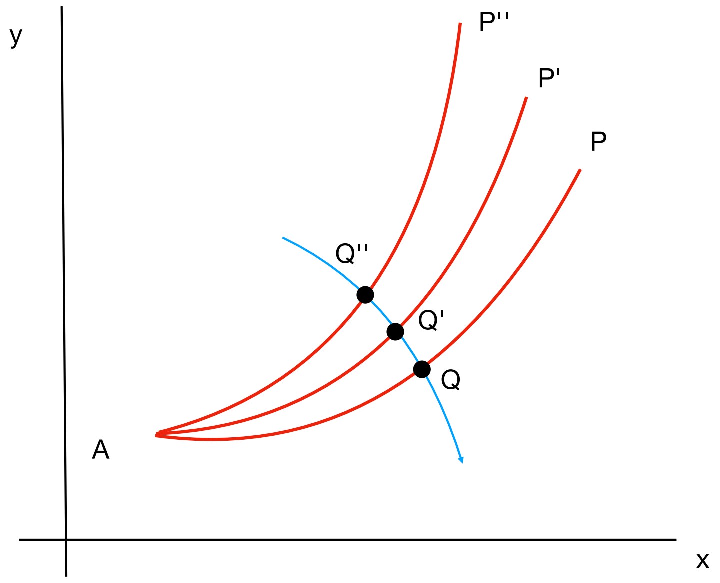
These expressions will be used to find the stream function.

Note that as $\frac{\partial^2\psi}{\partial x\partial y} = \frac{\partial^2\psi}{\partial y\partial x}$(Schwarz's theorem), we recover the incompressibility condition $\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} = 0.$

Recall that the stream function is only defined for **incompressible** flows.

The property of the stream function more relevant to us is the following:

The isovalue lines of the stream function are tangent to the flow velocity.



It is easy to see why: if the value of the stream function is the same at Q, Q', Q'', it means that the total flow crossing the red lines from A to Q, Q', Q'' is the same.

To satisfy this condition, the velocity has to be tangent to the blue line that joins all the points with the same value of ψ.

So, the streamlines (for a two-dimensional incompressible flow) are isovalues of the stream function and our method to represent them will be to evaluate first the stream function

Analytical evaluation of the stream function

In the rare situations where an analytical distribution of velocities is available, the stream function can be obtained by direct integration. We will consider a couple of examples before proceeding to the numerical integration that is often used.

Consider for instance a very simple case with a constant velocity field:

$$u = 1 \qquad v = 1$$

$$u = \frac{\partial \psi}{\partial y} \quad \longrightarrow \quad \psi_1 = \int u dy + f_1(x) + K_1 = y + f_1(x) + K_1$$

This is the first way to obtain the stream function, but there is another:

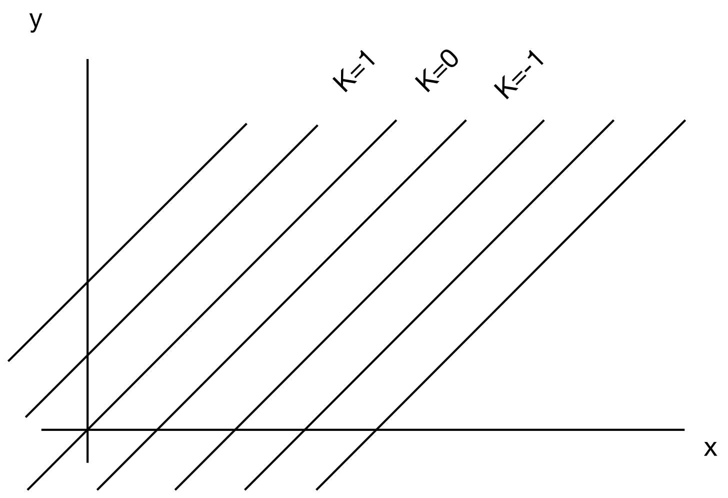
$$v = -\frac{\partial \psi}{\partial x} \quad \longrightarrow \quad \psi_2 = -\int v dx + f_2(y) + K_2 = -x + f_2(y) + K_2$$

Of course, both have to be equal:

$$\psi_1 = \psi_2 \quad \longrightarrow \quad \psi = y - x$$

Thus, the isovalue line of $\psi = K$ is $y = x + K$

13.3 Visualization. Streamline



Depending on the isovalue chosen, we get a different line.

All them are parallel to the velocity vector (1,1).

13.3 Visualization. Streamline

Consider now our initial conditions (Taylor solution for t=0, Eq. 2.2):

$$u = U_0 \cos\left(\frac{2\pi x}{L}\right) \sin\left(\frac{2\pi y}{L}\right)$$

$$v = -U_0 \sin\left(\frac{2\pi x}{L}\right) \cos\left(\frac{2\pi y}{L}\right)$$

From the previous expressions :

$$u = \frac{\partial \psi}{\partial y} \longrightarrow \psi_1 = \int u dy + f_1(x) + K_1 = -\frac{LU_0 \cos\left(\frac{2\pi x}{L}\right) \cos\left(\frac{2\pi y}{L}\right)}{2\pi} + f_1(x) + K_1$$

$$v = -\frac{\partial \psi}{\partial x} \longrightarrow \psi_2 = -\int v dx + f_2(y) + K_2 = -\frac{LU_0 \cos\left(\frac{2\pi x}{L}\right) \cos\left(\frac{2\pi y}{L}\right)}{2\pi} + f_2(y) + K_2$$

$$\boxed{\psi_1 = \psi_2}$$

$$\cancel{-\frac{LU_0\cos\left(\frac{2\pi x}{L}\right)\cos\left(\frac{2\pi y}{L}\right)}{2\pi}} + f_1(x) + K_1 = \cancel{-\frac{LU_0\cos\left(\frac{2\pi x}{L}\right)\cos\left(\frac{2\pi y}{L}\right)}{2\pi}} + f_2(y) + K_2$$

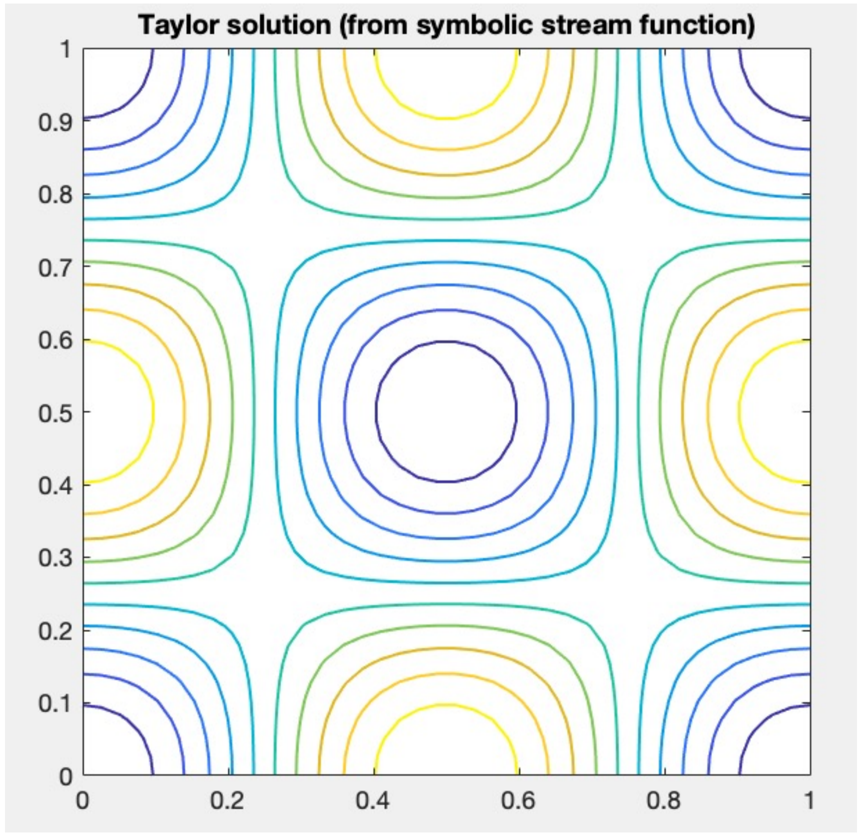
So, in this particular case, the functions are null and

$$\psi = -\frac{LU_0\cos\left(\frac{2\pi x}{L}\right)\cos\left(\frac{2\pi y}{L}\right)}{2\pi} + K$$

In order to plot isovalues of this function, we will use *Matlab*

```
clear
close all
% First, we will find the stream function
% using symbolic integration
syms x y t u v p F nu r L U0

u=U0*cos(2*pi*x/L)*sin(2*pi*y/L) ;
v=-U0*cos(2*pi*y/L)*sin(2*pi*x/L);

psi1 = int(u,y); % We integrate u respect x
psi2 = int(-v,x);

simplify(psi1-psi2) % We check that IN THIS
                    % PARTICULAR CASE, both are equal

psi=psi1;

% Now, we will evaluate numerically psi

N=40; % Mesh size for the isolines plot
cx=linspace(0,1,N);
cy=linspace(0,1,N);
[X,Y]=meshgrid(cx,cy);
                       .
                       .
                       .
```

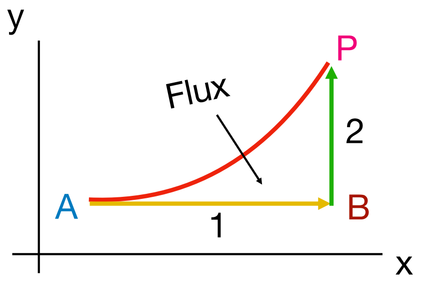
```
                       .
                       .
                       .
% The function meshgrid generates two matrices with x and
y coordinates
% For instance:
% [CX,CY]=meshgrid( [1,2],[3,4])
% CX =
%     1     2
%     1     2
% CY =
%     3     3
%     4     4

% define particular values for L and U0
psi = subs(psi,{L,U0},{1,1});

% find the numerical function
psiNumerical=matlabFunction(psi,'Vars',[x,y])

Z=psiNumerical(X,Y); % evaluate Z for each X,Y pair

contour(X,Y,Z,10,'LineWidth',2); % and finally, plot 10
                                 % isocontour values

set(gca,'FontSize',18) % choose a larger font

axis('equal')
```


Taylor solution (from symbolic stream function)
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
0.2
0.4
0.6
0.8
1

13.3 Visualization. Streamline

Usually, we don't have the analytic expression of our field and we have to compute the stream function numerically. To do so, we select a convenient integration path from an arbitrary point A where the stream function will be forced to be zero to point P (where P will be each of the nodes of the domain).

We chose A to be at the bottom left of the domain. Recall that due to the incompressibility condition, the path is arbitrary. First, we integrate along a horizontal line (1) and then along a vertical line (2) until we reach P:

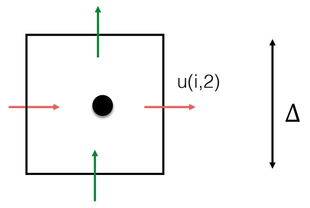


$$\psi = \int_A^P u dy - v dx$$

In part 1, the second term is zero: $\psi = \int_A^B - v dx$

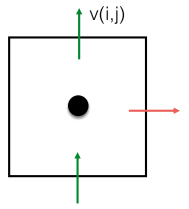
And in part 2, $\psi = \int_B^P u dy$

Part 1, the value of each column at the bottom row (j=2) is initialized:



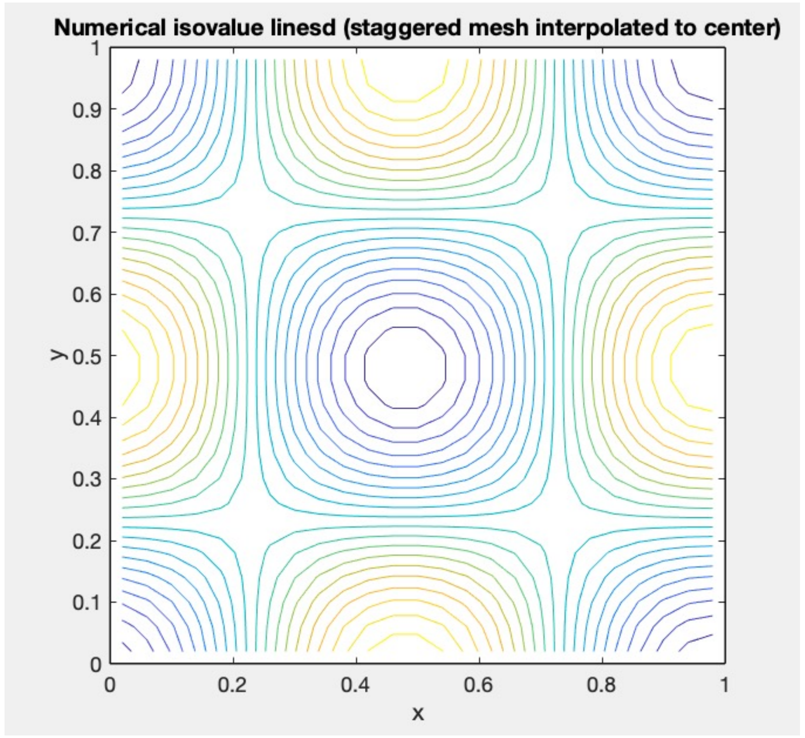
```
sf=zeros(size(u));
j=2;
for i=2:N+1
    sf(i,j)=sf(i-1,j)-delta*v(i,j);
end
```

Part 2, all the nodes along each column are updated:



```
for i=2:N+1
    for j=3:N+1
        sf(i,j)=sf(i,j-1)+delta*u(i,j);
    end
end
```

Finally, the stream function isovalues are plotted as before. Note that we have neglected the fact that each component of the velocity is in a different geometric position. The stream function has been assigned to the centred grid.


Numerical isovalue linesd (staggered mesh interpolated to center)
y
x