International Centre for
Numerical Methods in Engineering

Universitat Politècnica de Catalunya

Master of Science in Computational Mechanics

Finite Element in Fluids

Homework 7 - Unsteady Navier Stokes

Eugenio José Muttio Zavala

April 23, 2019

Submitted To:
Prof. Antonio Huerta
Prof. Matteo Giacomini
Prof. Pablo Saez

1 Unsteady Navier Stokes Problem

Problem Statement

Consider the unsteady Navier-Stokes equations and boundary conditions1.1 for incompressible fluid flows, in comparison with the steady version, a new term v_t in the momentum equation is added, which is the velocity rate with respect to the time.

$$\begin{cases} v_t - \nu\nabla^2\mathbf{v} + (\mathbf{v}\cdot\nabla)\mathbf{v} + \nabla p = \mathbf{b} & \text{in } \Omega x(0,T) \\ \nabla\cdot\mathbf{v} = 0 & \text{in } \Omega x(0,T) \\ \mathbf{v} = \mathbf{v}_D & \text{on } \Gamma_D x(0,T) \\ -p\mathbf{n}\cdot + \nu(\mathbf{n}\cdot\nabla)\mathbf{v} = \mathbf{t} & \text{on } \Gamma_N x(0,T) \\ \mathbf{v}(\mathbf{x},0) = \mathbf{v_0}(\mathbf{x}) & \text{in } \Omega \end{cases} \tag{1.1}$$

Where:

- $\mathbf{v}$ - is the velocity.
- p - is the pressure.
- ν - is kinematic viscosity.
- $\mathbf{b}$ - is a source term.
- $\mathbf{t}$ - is the tractions vector.

The weak form of the Navier-Stokes equations 1.1 in reduced notation is:

$$\begin{cases} (\mathbf{w},\mathbf{v}_t) + a(\mathbf{w},\mathbf{v}) + c(\mathbf{w},\mathbf{v},\mathbf{v}) + b(\mathbf{w},p) = (\mathbf{w},\mathbf{f}) + (\mathbf{w},\mathbf{t})_{\Gamma_N} \\ b(\mathbf{v},q) = 0 \end{cases} \tag{1.2}$$

Then, the finite element discretization of this weak form yields the system of semi-discrete equations:

$$\begin{cases} \mathbf{M}\dot{\mathbf{u}}(t) + [\mathbf{K} + \mathbf{C}(\mathbf{v}(t))]\mathbf{u}(t) + \mathbf{G}\mathbf{p}(t) = \mathbf{f}(t,\mathbf{v}(t)) \\ \mathbf{G}^{\mathbf{T}}\mathbf{u}(t) = \mathbf{h}(t) \\ \mathbf{u}(0) = \mathbf{v}_0 - \mathbf{v}_D(0) \end{cases} \tag{1.3}$$

where $\mathbf{M}$ is the standard finite element mass matrix defined as $\mathbf{M} = [mat\ N]^T[mat\ N]$. As theory mentions, the solution of a transient response can be done by suitable finite differences schemes such as the θ family methods. Now, as considering the semi-discrete equations stated above, a fully implicit scheme requires a nonlinear solution at each time - step, then a *semi-implicit* methodology where the convection matrix $\mathbf{C}(\mathbf{v}(t))$ and $\mathbf{f}(t,\mathbf{v}(t))$ are treated explicitly are preferred. Also, a time discretization of the Unsteady Navier-Stokes equations by means of a fractional-step procedure is explained next.

1.1 Cavity Flow

The cavity flow problem is the benchmark used to test the stabilization terms in this work. The figure 1.1 shows the geometry of the problem and includes the boundary conditions which presents zero velocities in each wall except for the upper wall, which has a magnitude of 1 in the positive horizontal direction. Because of that, the solution of the streamlines will behave as a vortex inside the domain. The pressure value is equal to zero in all the sides, but it should be considered that at the two upper corners (where the velocity boundary condition is applied) there will appear a singularity in pressure in each corner.

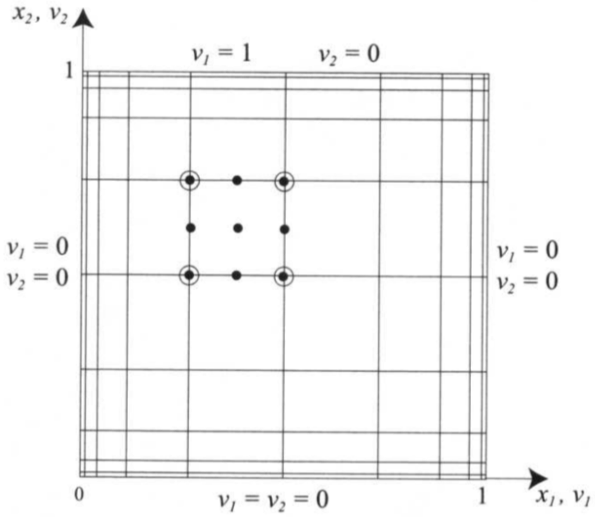


Figure 1.1: Cavity Flow Problem boundary conditions.

1.2 Chorin-Temam Projection Method

Fractional-step proposes a time advancement which is decomposed into a sequence of two or more steps, this approach to time integration allows us to alleviate the numerical difficulties related to the saddle point problem that arises from the variational form of the Navier-Stokes equations. The principle of projection method is to compute the velocity and pressure fields separately through the computation of an intermediate velocity. The Chorin-Temam projection method includes two basic steps as follows.

1st Step: This includes the viscous and convective in convective terms in the Navier-Stokes equations and (given the previous velocity field v^n) consist of finding an intermediate velocity field v^{n+1} such that:

$$\begin{cases} \frac{\mathbf{v}_{int}^{n+1}-\mathbf{v}^n}{\Delta t} + (\mathbf{v}^* \cdot \nabla)\mathbf{v}^{**} - \nu\nabla^2\mathbf{v}^{**} = \mathbf{f}^{n+1} & \text{in } \Omega \\ \mathbf{v}_{int}^{n+1} = \mathbf{v}_D^{n+1} & \text{on } \Gamma \end{cases} \tag{1.4}$$

where $\mathbf{v}^*$ and $\mathbf{v}^{**}$ must be chosen suitable for the treatment of the nonlinear convective term, possible options are:

- Explicit Euler $\mathbf{v}^*$=$\mathbf{v}^{**}$=$\mathbf{v}^n$
- Semi-Implicit $\mathbf{v}^*$=$\mathbf{v}^n$ and $\mathbf{v}^{**}$=$\mathbf{v}_{int}^{n+1}$
- Implicit Euler $\mathbf{v}^*$=$\mathbf{v}^{**}$=$\mathbf{v}_{int}^{n+1}$

For the semi-implicit and fully implicit cases, the algebraic system resulting from the finite element discretization is:

$$\mathbf{M}_1 \left(\frac{\mathbf{v}_{int}^{n+1} - \mathbf{v}^n}{\Delta t} \right) + (\mathbf{C}(\mathbf{v}^*) + \mathbf{K})\mathbf{v}_{\mathbf{int}}^{\mathbf{n+1}} = \mathbf{f}^{n+1} \tag{1.5}$$

2nd Step: This part determines the velocity $\mathbf{v}^{\mathbf{n+1}}$ and pressure $\mathbf{p}^{\mathbf{n+1}}$, by solving the next system:

$$\begin{cases} \frac{\mathbf{v}_{int}^{n+1} - \mathbf{v}^n}{\Delta t} + \nabla p^{n+1} = 0 & \text{in } \Omega \\ \nabla \cdot v^{n+1} = 0 & \text{in } \Omega \\ \mathbf{n} \cdot \mathbf{v}_{int}^{n+1} = \mathbf{n} \cdot \mathbf{v}_D^{n+1} & \text{on } \Gamma \end{cases} \tag{1.6}$$

The discrete equations emanating from the discretization of the weak form of the above equations induce the following system of algebraic equations:

$$\begin{cases} \mathbf{M}_2 \left(\frac{\mathbf{v}^{n+1} - \mathbf{v}_{int}^{n+1}}{\Delta t} \right) + \mathbf{G}\mathbf{p}^{n+1} = 0 \\ \mathbf{G}^{\mathbf{T}}\mathbf{v}^{n+1} = 0 \end{cases} \tag{1.7}$$

or equivantly,

$$\begin{bmatrix} \mathbf{M}_2/\Delta t & \mathbf{G} \\ \mathbf{G}^{\mathbf{T}} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{v}^{n+1} \\ \mathbf{p}^{n+1} \end{bmatrix} = \begin{bmatrix} \mathbf{M}_2 \mathbf{v}_{int}^{n+1}/\Delta t \\ \mathbf{0} \end{bmatrix} \tag{1.8}$$

Code Implementation 1

In order to solve the problem, the semi-implicit methodology is implemented inside a loop that computes at each time step, using the previous workclass implementation code in Matlab, it was easier to code the first and second steps required for the Chorim-Temam method as:

```
while step < nstep
    step = step +1;
    fprintf('Time Step = %d\n',step);
    C = ConvectionMatrix(X,T,referenceElement,velo);
    Cred = C(dofUnk,dofUnk);
    fred_n = fred - (K(dofUnk,dofDir)+C(dofUnk,dofDir))*valDir;

 %%% CHORIN-TEMAM %%%
 %%% 1st STEP %%%
        btot = dt*fred_n+Mred*veloVect(dofUnk);
        Atot = Mred+dt*(Cred+Kred);
        sol_1 = Atot \ btot;

 %%% CHORIN-TEMAM %%%
 %%% 2nd STEP %%%
        btot = [Mred*sol_1; f_q];
        Atot = [Mred Gred'*dt; Gred L];
        aux = Atot \ btot;

        veloInc = zeros(ndofV,1);
        veloInc(dofUnk) = aux(1:nunkV);
        presInc = aux(nunkV+1:end);
        velo = reshape(veloInc,2,[])';
        pres = presInc;
end
```

1.3 THETA FAMILY METHOD

Consider a semi-implicit discretization scheme based on the *Theta Method* formulation:

$$\frac{\Delta \mathbf{u}}{\Delta t} - \theta \Delta \mathbf{u}_t = \mathbf{u}_t^n \tag{1.9}$$

where the θ parameter defines the type of methodology employed. As can be seen in equation 1.9, if $\theta = 0$, then the increment of the solution $\left(\frac{\Delta \mathbf{u}}{\Delta t} = \frac{\mathbf{u}^{n+1} - \mathbf{u}^n}{\Delta t}\right)$ depends only in the "n" time step solution, where it said that the method is explicit. Then the opposite option is to use $\theta = 1$, now the method is considered as implicit. The selection of the θ parameter chosen rely in the stabilization of the solution, where is achievable for values between $0.5 \leq \theta \leq 1$. Moreover, a most accurate value can be obtained by using $\theta = 0.5$, which is known as **Crank-Nicolson**. In order to implement this, consider the transient weak form of Navier-Stokes 1.3, which can be re-arranged as:

$$\mathbf{u}_t = \frac{\mathbf{f} - (\mathbf{K} + \mathbf{C}(v))\mathbf{u} - \mathbf{G}\mathbf{p}}{M} \tag{1.10}$$

Substituting this equation in the expression 1.9, then

$$\frac{\Delta \mathbf{u}}{\Delta t} - \theta \left[\left(\frac{-(\mathbf{K} + \mathbf{C^{n+1}})\mathbf{u}^{n+1} - \mathbf{G}\mathbf{p}^{n+1}}{M} \right) - \left(\frac{-(\mathbf{K} + \mathbf{C^n})\mathbf{u}^n - \mathbf{G}\mathbf{p}^n}{M} \right) \right] = \frac{\mathbf{f} - (\mathbf{K} + \mathbf{C^n})\mathbf{u}^n - \mathbf{G}\mathbf{p}^n}{M} \tag{1.11}$$

As the formulation was obtained using the *Theta Method* in a *Semi-Implicit* scheme, the nonlinear convective term **C** is considered as linearized, and the above equation can be re-arranged as:

$$\frac{\Delta \mathbf{u}}{\Delta t} + \theta \left(\frac{(\mathbf{K} + \mathbf{C^n})\Delta \mathbf{u} + \mathbf{G}\Delta \mathbf{p}}{M} \right) = \frac{\mathbf{f} - (\mathbf{K} + \mathbf{C^n})\mathbf{u}^n - \mathbf{G}\mathbf{p}^n}{M} \tag{1.12}$$

To complete the above system, the divergence free condition must be imposed. Then the equations can be written as:

$$\begin{cases} (\mathbf{M} + \theta \Delta t(\mathbf{K} + \mathbf{C}^n))\Delta \mathbf{u} + \theta \Delta t \mathbf{G} \Delta \mathbf{p} = \Delta t(\mathbf{f} - (\mathbf{K} + \mathbf{C}^n)\mathbf{u}^n - \mathbf{G}\mathbf{p}^n) \\ \mathbf{G}^T \Delta \mathbf{u} = 0 \end{cases} \tag{1.13}$$

Code Implementation 2

The solution of the transient Navier-Stokes problem can be found using the time discretization by the θ methods. In order to implement this formulation into the Matlab code, first consider the equation 1.13 in matrix form as:

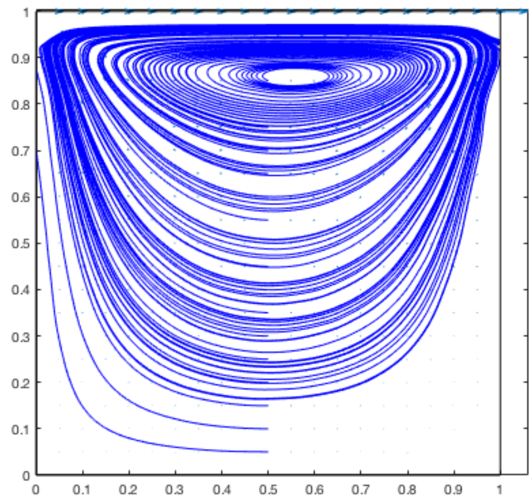
$$\begin{bmatrix} \mathbf{M}+\theta\Delta t(\mathbf{K}+\mathbf{C}^n) & \theta\Delta t\mathbf{G} \\ \mathbf{G}^T & \mathbf{0} \end{bmatrix} \begin{bmatrix} \Delta\mathbf{u} \\ \Delta\mathbf{p} \end{bmatrix} = \begin{bmatrix} \Delta t(\mathbf{f}-(\mathbf{K}+\mathbf{C}^n)\mathbf{u}^n-\mathbf{G}\mathbf{p}^n) \\ \mathbf{0} \end{bmatrix} \tag{1.14}$$

where the increment is obtained by solving the system in each time, and a posteriori it can be obtained the solution by adding the previous step. Consider the next lines of code:

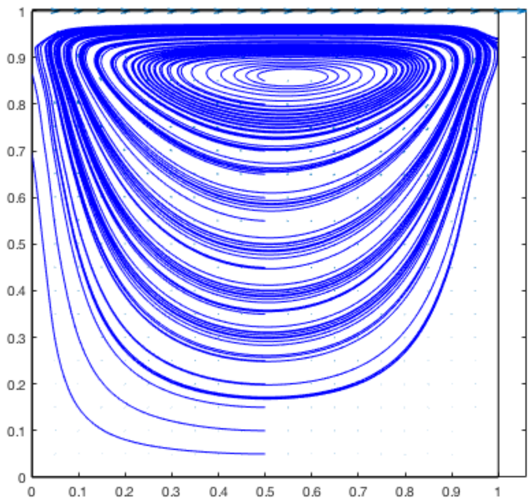
```
while step < nstep
    step = step +1;
    fprintf('Time Step = %d\n',step);
    C = ConvectionMatrix(X,T,referenceElement,velo);
    Cred = C(dofUnk,dofUnk);
    fred_n = fred - (K(dofUnk,dofDir)+C(dofUnk,dofDir))*valDir;

    %System of equations definition
    Amat = [Mred+theta*dt*(Kred+Cred)    dt*theta*Gred'
    Gred    L];
    ftot = [dt*(fred_n-(Kred+Cred)*veloVect(dofUnk)-Gred'*pres); f_q];

     % Solution of each increment
     solInc = Atot \ btot;

     % Solution updating at each time
     veloInc = zeros(ndofV,1);
     veloInc(dofUnk) = solInc(1:nunkV);
     presInc = solInc(nunkV+1:end);
     velo = velo + reshape(veloInc,2,[])';
     pres = pres + presInc;
end
```

1.4 Cavity Flow Problem - Solver Scheme Comparison

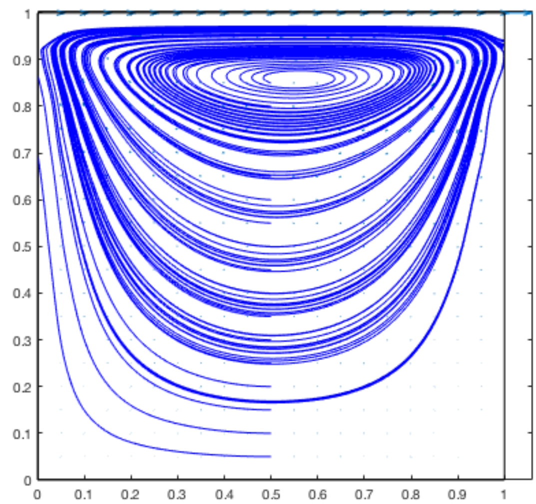
In order to compare both time discretization schemes, the cavity flow problem is solved using a 10x10 Q2Q1 quadrilateral discretization in the domain for the velocity and pressure. Consider a comparison in different times using two θ implicit solutions, one unstable ($\theta = 0.25$) and one stable ($\theta = 0.25$), and then a comparison between the stable and accurate solution given by Crank-Nicolson ($\theta = 0.5$) and the Chorin-Temam projection method:



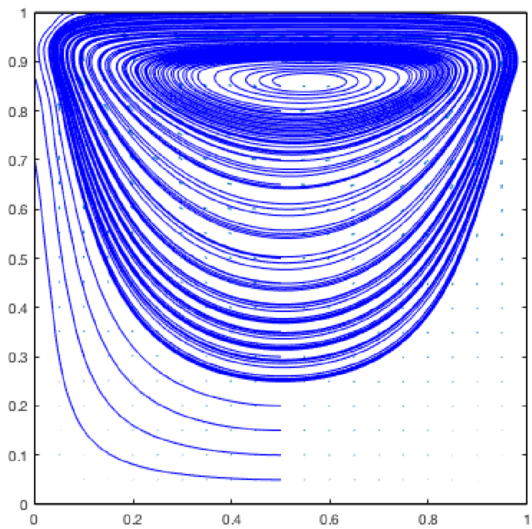
(a) $\theta = 0.25$



(b) $\theta = 1$

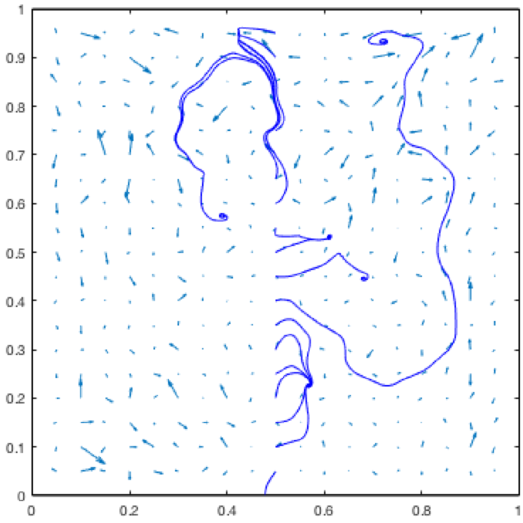


(c) $\theta = 0.5$

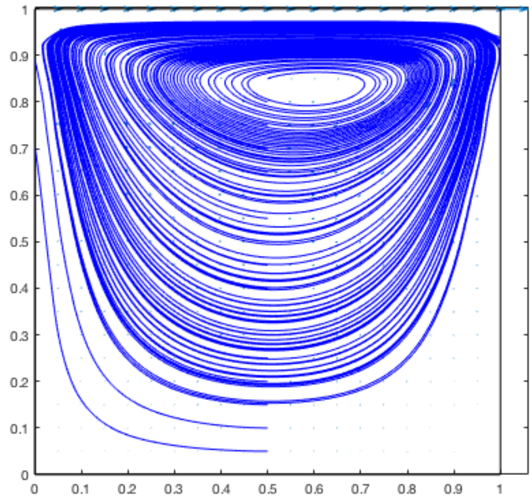


(d) Chorin-Temam

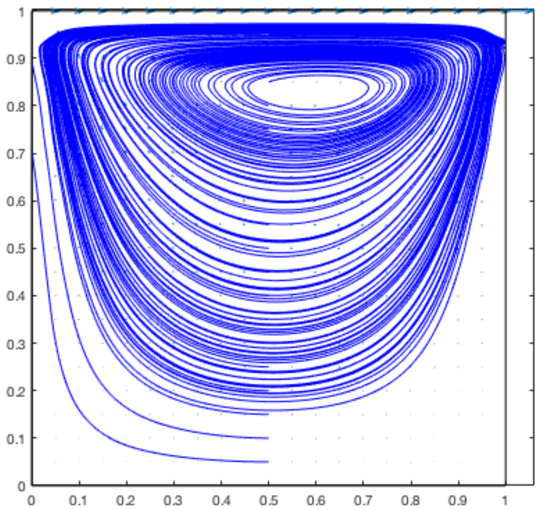
Figure 1.2: Time = 0.3 seconds. Comparison between a) unstable and b) stable θ method. Comparison between c) Crank-Nicolson and d) Chorin-Temam projection method.



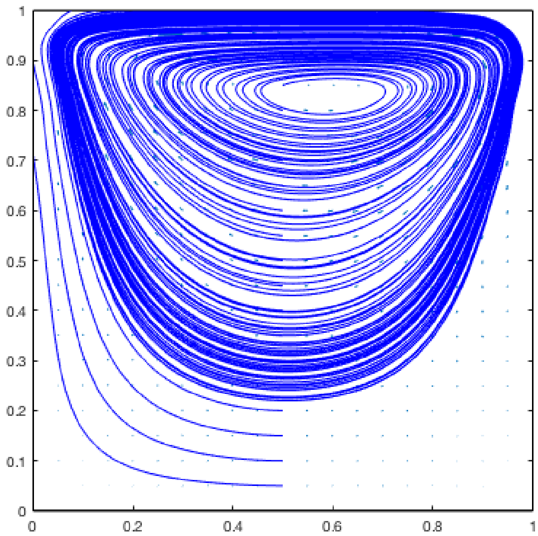
(a) $\theta = 0.25$



(b) $\theta = 1$

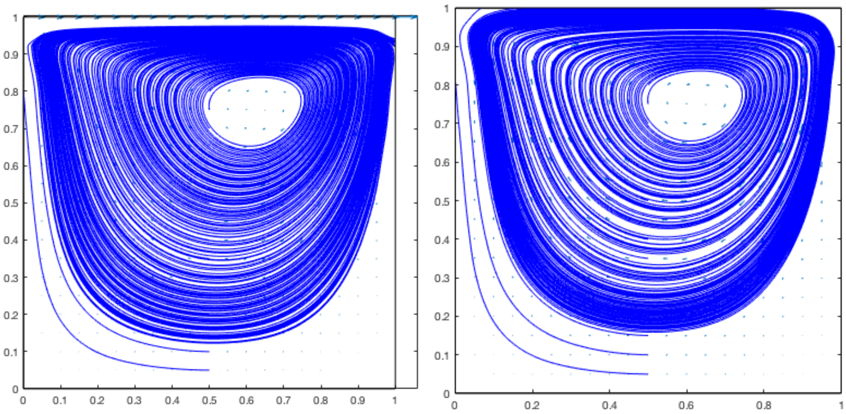


(c) $\theta = 0.5$



(d) Chorin-Temam

Figure 1.3: Time = 0.6 seconds. Comparison between a) unstable and b) stable θ method. Comparison between c) Crank-Nicolson and d) Chorin-Temam projection method.



(a) $\theta = 0.5$

(b) Chorin-Temam

Figure 1.4: Time = 3.0 seconds. Comparison between a) Crank-Nicolson and b) Chorin-Temam projection method.

1.5 Discussion

The first comparison that can discussed is the corresponding to the unstable and stable θ method. The graphs a) and b) of figure 1.2 present a 0.3 sec comparison, using implicit $\theta = 0.25$ and $\theta = 1$. Both parameters offer adequate results at this time. Now, consider the same pair of graphs but in figure 1.3 that corresponds to 0.6 sec, at this point the solution given by the unstable $\theta = 0.25$ is very remarkable, the solution is no longer useful. In contrast the implicit $\theta = 1$ continues working as expected. The second comparison done is between both time discretization methodologies. The graphs presented in figure 1.2 and 1.3 corresponding to 0.3 and 0.6 seconds respectively, are closer to each other. No main difference can be noticeable respect to stability, but the fluid trajectories are a little bit distinct. For example, the upper face of the domain where the velocity is constrained to one, the θ method leaves a boundary between the flow and the wall, in contrast to Chorim-Temam where the fluid is moving closely to the wall. Finally, the last test is carried out using only the accurate a) Crank-Nicolson and b) Chorim-Temam at t=3 seconds, shown in figure 1.4. The simulation obtained by both methodologies are stable but there is small difference in the trajectories, where the Crank-Nicolson shows lines almost equally spaced covering most part of the domain. But the Chorin-Temam presents a fluid that overlaps at the far from the center of vortex formed.

In conclusion, the motion of fluid is not a simple problem, moreover nonlinear terms and transient analysis that are necessary to obtain a simulation of the flow add difficulty to this work. Time-discretization have to be proposed but the computation is complex and expensive, then *Theta Method* and *Chorim-Temam* are different perspectives proposed to achieved this problem. Both methodologies are intelligent ways to avoid the computation of nonlinear terms of the Navier-Stokes problem, by linearizing the convection term to overcome the necessity of an iterative solver that most can increase the computer time computation.