Universitat Politècnica de Catalunya

Master of Science in Computational Mechanics

Finite Elements in Fluids

Assignment 6
Incompressible Flow

Authors:
Carlos Eduardo Ribeiro Santa Cruz Mendoza

April 3, 2019

1 Stokes

The governing equations for a steady, isothermal and incompressible fluid flow are stated on Equation 1.1 and are regarded as the (simplified) Navier-Stokes equations.

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

The first equation, namely the balance of momentum, is composed respectively of the viscous term, the convective term, the pressure gradient and the external body forces. Flows under relatively low Reynolds number ($Re = \frac{vL}{\nu} < 1$, with L being a characteristic length) are represented by a predominance of the viscous forces over the convective forces. This allows Equation 1.1 to be further simplified by neglecting the convective term $(\boldsymbol{v}\cdot\nabla)\boldsymbol{v}$. Considering a problem with just Dirichlet boundary conditions, the simplified governing equations are given by Equation 1.2 and are referred to as Stokes equations.

$$\begin{cases} -\nu\nabla^2\boldsymbol{v} + \nabla p = \boldsymbol{b} & \text{in } \Omega \\ \nabla\cdot\boldsymbol{v} = 0 & \text{in } \Omega \\ \boldsymbol{v} = \boldsymbol{v}_D & \text{on } \Gamma \end{cases} \tag{1.2}$$

1.1 Cavity flow problem

The stokes equations can be applied to model a 2D flow inside a cavity, where the "lid" (at $y = 1$) has a velocity $v^* = 1$ while all the other walls have velocity zero. Thus, with the cavity completely filled by fluid and applying the no-slip condition, we can state the boundary conditions as depicted on Figure 1.1.

Given that we're dealing only with Dirichlet boundary conditions, only relative pressure fields can be obtained. Thus, a reference value of zero was assigned to the origin.

The problem and corresponding governing equations were discretized via standard Galerkin formulation, which solves the linear system presented on Equation 1.3.

$$\begin{bmatrix} \boldsymbol{K} & \boldsymbol{G}^T \\ -\boldsymbol{G} & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{v} \\ \boldsymbol{p} \end{bmatrix} = \begin{bmatrix} \boldsymbol{f} \\ \boldsymbol{0} \end{bmatrix} \tag{1.3}$$

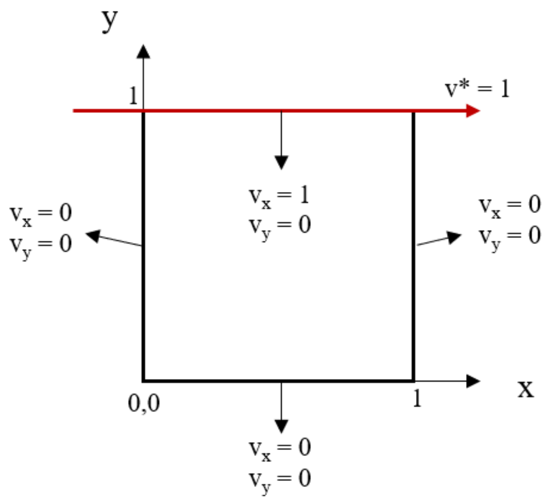
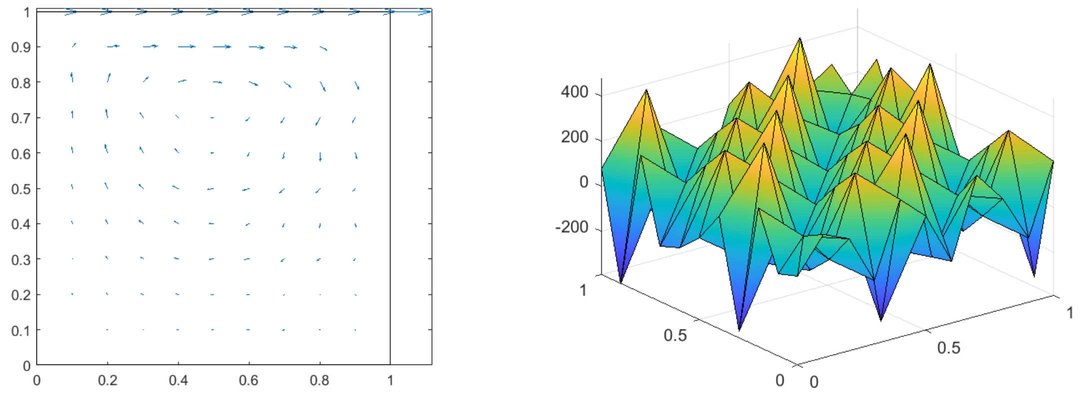


Figure 1.1: Domain and boundary conditions of the cavity problem

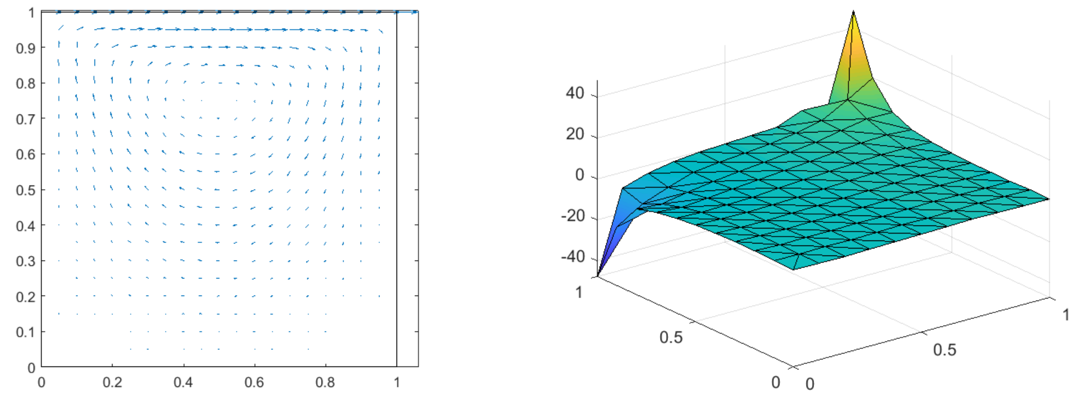
The domain was discretized in a regular 10x10 mesh, whereas different elements and interpolations for the velocity were tested. Triangular elements were tested with linear pressure interpolation and both linear and quadratic interpolation for velocity (P1P1 and P2P1, respectively) and quadrilateral elements likewise (Q1Q1 and Q2Q1).

The results for the aformentioned cases are given on Figures 1.2 and 1.3 for the triangular elements and on Figures 1.4 and 1.5 for the quadrilateral.



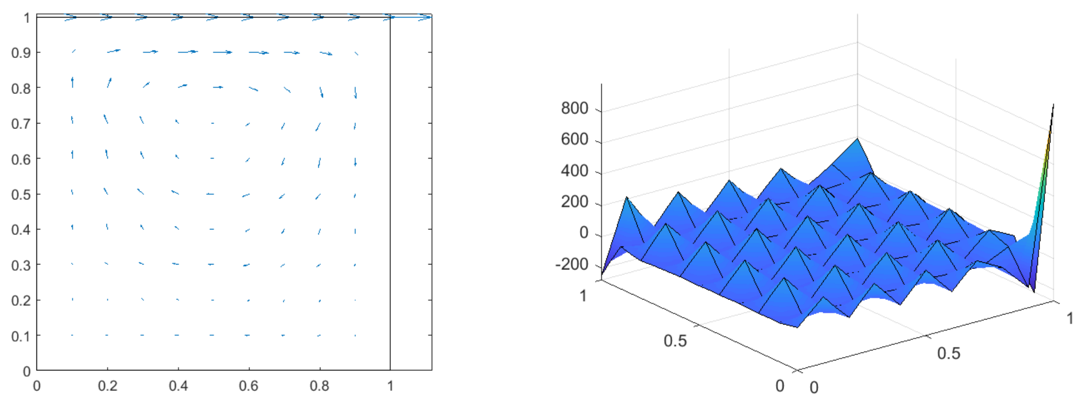
(a) (b)

Figure 1.2: Vector field (a) and pressure field (b) with P1P1 elements



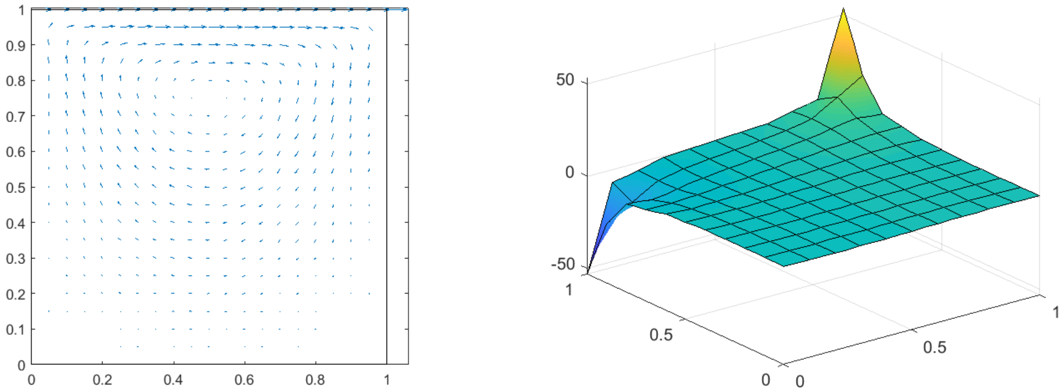
(a) (b)

Figure 1.3: Vector field (a) and pressure field (b) with P2P1 elements



(a) (b)

Figure 1.4: Vector field (a) and pressure field (b) with Q1Q1 elements



(a) (b)

Figure 1.5: Vector field (a) and pressure field (b) with Q2Q1 elements

It's evident that a linear interpolation of the velocity field has lead to significant instabilities on the pressure field, even though the velocity field itself has converged fairly well (with minor oscillations in respect to the stable solutions). This is explained by the so-called "LBB condition", which proves that the pressure and the velocity field are coupled in a way that the interpolation of one variable conditions the interpolation of the other. If these conditions are not met, the solution will be unstable. Among the unstable elements are P1P1 and Q1Q1, explaining the obtained instabilities.

However, there are methods to overcome the instabilities and avoid using elements with quadratic interpolation (higher computational cost). Similarly to the convection-diffusion problem, the SUPG stabilization technique can be applied, leading to the modified linear system shown on Equation 1.4.

$$\begin{bmatrix} \boldsymbol{K} + \bar{\boldsymbol{K}} & \boldsymbol{G}^T + \bar{\boldsymbol{G}}^T \\ -\boldsymbol{G} + \bar{\boldsymbol{G}} & 0 + \bar{\boldsymbol{L}} \end{bmatrix} \begin{bmatrix} \boldsymbol{v} \\ \boldsymbol{p} \end{bmatrix} = \begin{bmatrix} \boldsymbol{f} + \bar{\boldsymbol{f}}_w \\ \boldsymbol{0} + \bar{\boldsymbol{f}}_q \end{bmatrix} \tag{1.4}$$

Since it's relevant to apply this method only to linearly interpolated elements we can exclude the $\bar{\boldsymbol{K}}$, $\bar{\boldsymbol{G}}$ and $\bar{\boldsymbol{f}}_w$ terms, given that they contain second derivatives of the variables and are, thus, zero. Hence, only $\bar{\boldsymbol{L}}$ and $\bar{\boldsymbol{f}}_q$ need to be inserted to the code. Their weak form is given on Equations 1.5 and 1.6.

$$\bar{\boldsymbol{L}} = \sum_e \int_{\Omega_e} \tau_1 (\nabla q) \cdot (\nabla p) d\Omega_e \tag{1.5}$$

$$\bar{\boldsymbol{f}}_q = \sum_e \int_{\Omega_e} \tau_1 (\nabla q) \cdot (-\boldsymbol{f}) d\Omega_e \tag{1.6}$$

Where $tau_1 = \frac{h^2}{12\nu}$ (h is size of the element) and q is a scalar shape function.

The discretization of $\bar{\boldsymbol{L}}$ can be done as follows:

$$(\nabla q) \cdot (\nabla p) = \begin{bmatrix} \frac{\partial q}{\partial x} & \frac{\partial q}{\partial y} \end{bmatrix} \begin{bmatrix} \frac{\partial p}{\partial x} \\ \frac{\partial p}{\partial y} \end{bmatrix} = \frac{\partial q}{\partial x}\frac{\partial p}{\partial x} + \frac{\partial q}{\partial y}\frac{\partial p}{\partial y}$$

but we know that the pressure derivative is discretized as follows

$$\frac{\partial p}{\partial x} = \frac{\partial N_1}{\partial x}p_1 + \frac{\partial N_2}{\partial x}p_2 + ...$$

$$\frac{\partial p}{\partial x} = \begin{bmatrix} \frac{\partial N_1}{\partial x} & \frac{\partial N_2}{\partial x} & ... & \frac{\partial N_m}{\partial x} \end{bmatrix} \begin{bmatrix} p_1 \\ p_2 \\ \vdots \\ p_m \end{bmatrix}$$

The weight function is similarly discretized and, knowing that on the implementation the vector containing the derivatives of the shape function in respect to x is called $\boldsymbol{n}_x$, we can write:

$$(\nabla q) \cdot (\nabla p) = (\boldsymbol{n}_x \boldsymbol{q})^T (\boldsymbol{n}_x \boldsymbol{p}) + (\boldsymbol{n}_y \boldsymbol{q})^T (\boldsymbol{n}_y \boldsymbol{p})$$

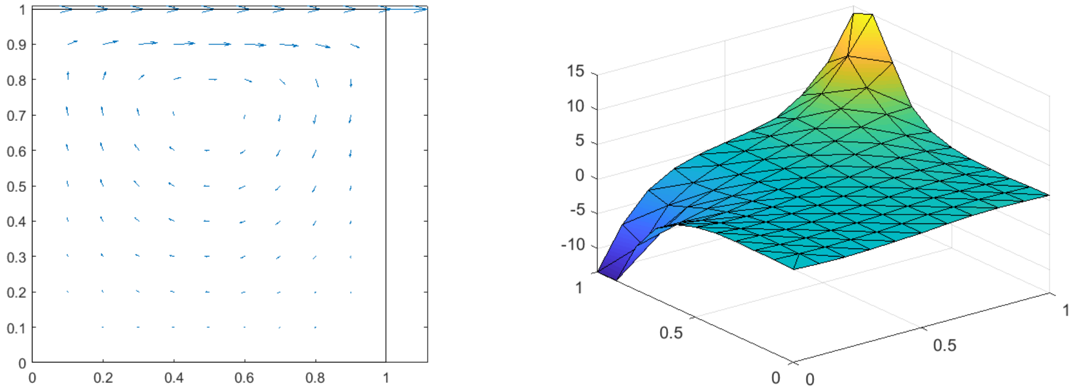
or, given that the equation holds for any weight function q:

$$(\nabla q) \cdot (\nabla p) = (\boldsymbol{n}_x^T \boldsymbol{n}_x + \boldsymbol{n}_y^T \boldsymbol{n}_y)\boldsymbol{p} \tag{1.7}$$

Following the same procedure $\bar{\boldsymbol{f}}_q$ is discretized as:

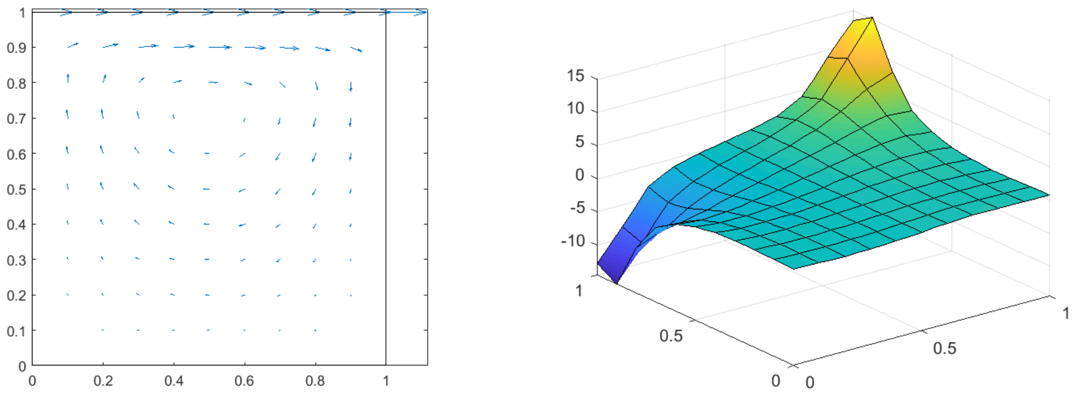
$$(\nabla q) \cdot (-\boldsymbol{f}) = - \begin{bmatrix} \boldsymbol{n}_x & \boldsymbol{n}_y \end{bmatrix} \begin{bmatrix} f_1 \\ f_2 \end{bmatrix} \tag{1.8}$$

Implementing Equations 1.7 and 1.8 to the code and adding them as seen on Equation 1.4 we are able to solve the modified linear system. The new results for the P1P1 and Q1Q1 elements are presented on Figures 1.6 and 1.7, respectively.



(a) (b)

Figure 1.6: Vector field (a) and pressure field (b) with P1P1 elements (SUPG)



(a) (b)

Figure 1.7: Vector field (a) and pressure field (b) with Q1Q1 elements (SUPG)

As we can see, the modified linear system given by the SUPG method provides stable results for the pressure field with a consistent formulation. This proves to be useful, given that the linear interpolation methods provided results with significant greater speed than the quadratic (0.097s against 0.421s for quadrilateral elements).

2 Navier-Stokes

The Navier-Stokes equations, as seen on Exercise 1, are repeated below.

$$\begin{cases} -\nu\nabla^2\boldsymbol{v} + (\boldsymbol{v}\cdot\nabla)\boldsymbol{v} + \nabla p = \boldsymbol{b} & \text{in } \Omega \\ \nabla\cdot\boldsymbol{v} = 0 & \text{in } \Omega \\ \boldsymbol{v} = \boldsymbol{v}_D & \text{on } \Gamma_D \\ \boldsymbol{n}\cdot\boldsymbol{\sigma} = \boldsymbol{t} & \text{on } \Gamma_N \end{cases} \tag{2.1}$$

When the convective term is relevant in respect to the viscous term, it must be implemented, with the expense of adding a nonlinearity to the problem. The Galerkin formulation takes now the form given in Equation 2.2

$$\begin{bmatrix} \boldsymbol{K} + \boldsymbol{C}(\boldsymbol{v}) & \boldsymbol{G}^T \\ -\boldsymbol{G} & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{v} \\ \boldsymbol{p} \end{bmatrix} = \begin{bmatrix} \boldsymbol{f} \\ \boldsymbol{0} \end{bmatrix} \Rightarrow \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \tag{2.2}$$

The convection matrix $\boldsymbol{C}(v)$ is formulated after the discretization of the convective term:

$$\boldsymbol{w}\cdot(\boldsymbol{v}\cdot\nabla)\boldsymbol{v} = \boldsymbol{w}\cdot\begin{bmatrix} v_x & v_y \end{bmatrix}\cdot\begin{bmatrix} \frac{\partial}{\partial x} \\ \frac{\partial}{\partial y} \end{bmatrix}\begin{bmatrix} v_x & v_y \end{bmatrix}$$

$$= \boldsymbol{w}\cdot(v_x\frac{\partial}{\partial x} + v_y\frac{\partial}{\partial y})\begin{bmatrix} v_x & v_y \end{bmatrix}$$

$$= \boldsymbol{w}\cdot(v_x\frac{\partial}{\partial x}\begin{bmatrix} v_x \\ v_y \end{bmatrix} + v_y\frac{\partial}{\partial y}\begin{bmatrix} v_x \\ v_y \end{bmatrix})$$

$$= \boldsymbol{w}(\begin{bmatrix} v_x & v_y \end{bmatrix}(\frac{\partial}{\partial x}\begin{bmatrix} v_x \\ v_y \end{bmatrix} + \frac{\partial}{\partial y}\begin{bmatrix} v_x \\ v_y \end{bmatrix}))$$

Thus, the Convection matrix will be given by:

$$\boldsymbol{C} = \boldsymbol{w}\cdot(\frac{\partial}{\partial x}\begin{bmatrix} v_x \\ v_y \end{bmatrix} + \frac{\partial}{\partial y}\begin{bmatrix} v_x \\ v_y \end{bmatrix})$$

or, in discretized form, knowing that the reshaped shape function vector and its derivatives are given by $\boldsymbol{N}$, $\boldsymbol{N}_x$, $\boldsymbol{N}_y$ and v_r is the reshaped velocity:

$$\boldsymbol{C} = \boldsymbol{N}(\boldsymbol{N}_x\boldsymbol{v}_r + \boldsymbol{N}_y\boldsymbol{v}_r)$$

2.1 Iterative methods

The non-linearity of the convection term requires an iterative method to solve it. The Picard's method uses an initial guess to calculate the residual and then assign a new guess:

$$\boldsymbol{A}(\boldsymbol{x}^k)\boldsymbol{x}^{k+1} = \boldsymbol{b}(\boldsymbol{x}^k)$$

$$res = \boldsymbol{b}(\boldsymbol{x}^k) - \boldsymbol{A}(\boldsymbol{x}^k)\boldsymbol{x}^k \Rightarrow \boldsymbol{A}^{-1}(\boldsymbol{x}^k)\boldsymbol{b}(\boldsymbol{x}^k) = \boldsymbol{x}^{k+1}$$

The method solves one linear system of equations per iteration k, converging linearly until the imposed tolerance.

The Newton-Raphson's method, on the other hand, has a quadratic convergence and defines (for the Navier-Stokes problem):

$$\boldsymbol{res} = \begin{bmatrix} \boldsymbol{K} + \boldsymbol{C})\boldsymbol{v} + \boldsymbol{G}^T p - f \\ \boldsymbol{G}\boldsymbol{v} \end{bmatrix}$$

and, via Taylor expansion

$$\boldsymbol{res}^{k+1} = \boldsymbol{res}^k + \boldsymbol{J}\Delta\boldsymbol{x}$$

where $\boldsymbol{J}$ is the jacobian matrix and $\Delta\boldsymbol{x}$ is the vector with the increment of the velocity and pressure fields in respect to the previous iteration. To obtain the field on the current iteration, $\boldsymbol{res}^{k+1}$ is assumed to be zero and, by inverting the jacobian, the increment can be found. The difficulty lies on calculating the Jacobian:

$$\boldsymbol{J} = \begin{bmatrix} \frac{\partial res_1}{\partial v} & \frac{\partial res_1}{\partial p} \\ \frac{\partial res_2}{\partial v} & \frac{\partial res_2}{\partial p} \end{bmatrix} = \begin{bmatrix} \boldsymbol{K} + \boldsymbol{C}(\boldsymbol{v}) + \frac{d\boldsymbol{C}}{d\boldsymbol{v}}\boldsymbol{v} & \boldsymbol{G}^T \\ -\boldsymbol{G} & 0 \end{bmatrix}$$

where the third term of J_{11}, which will be from now on called $\boldsymbol{C}_2$, it's still unknown. But from the definition of the convection matrix given before and applying the derivative in respect to the velocity vector (neglecting second order derivatives) we find:

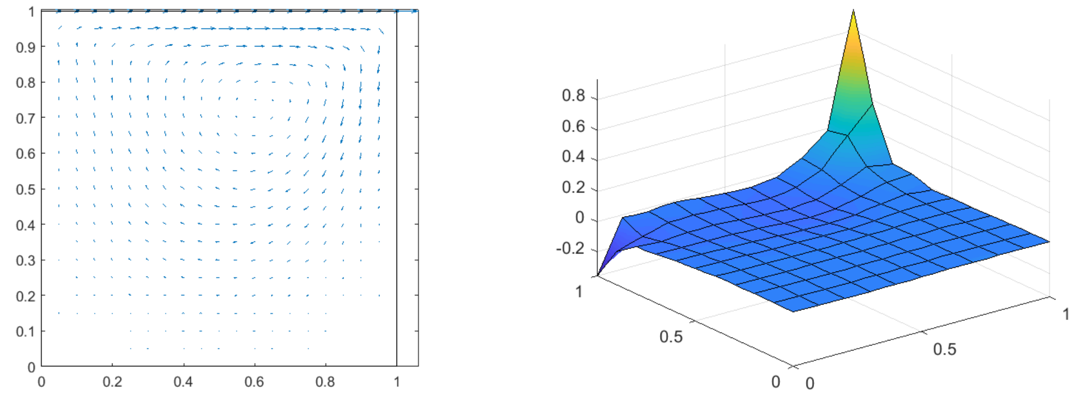
$$\boldsymbol{C}_2 = \boldsymbol{w} \cdot \begin{bmatrix} \frac{\partial v_x}{\partial x} & \frac{\partial v_x}{\partial y} \\ \frac{\partial v_y}{\partial x} & \frac{\partial v_y}{\partial y} \end{bmatrix} \begin{bmatrix} v_x & v_y \end{bmatrix} = \boldsymbol{w} \cdot (\begin{bmatrix} \frac{\partial}{\partial x} \\ \frac{\partial}{\partial y} \end{bmatrix} \begin{bmatrix} v_x & v_y \end{bmatrix})^T \cdot \boldsymbol{v}$$

which after discretization becomes:

$$\boldsymbol{C}_2 = \boldsymbol{N} \cdot (\begin{bmatrix} \boldsymbol{n}_x \\ \boldsymbol{n}_y \end{bmatrix} \boldsymbol{u}_e)^T \boldsymbol{N}$$

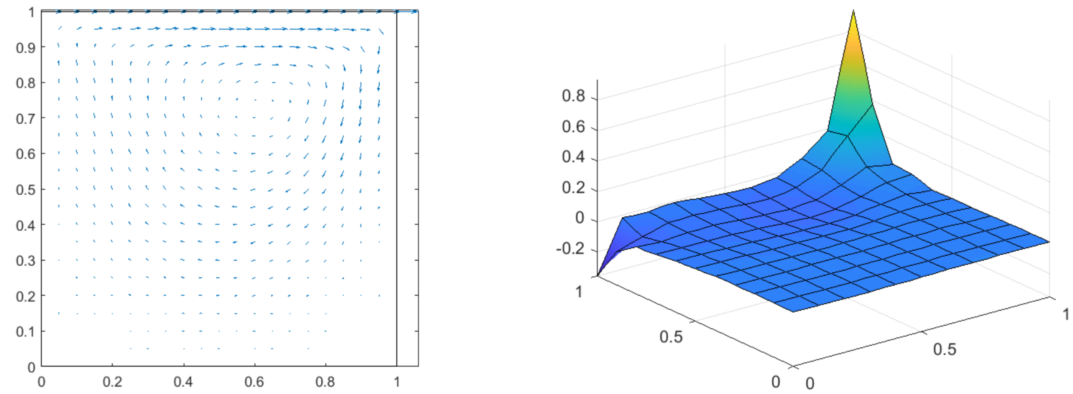
where u_e is the vector velocity.

After the implementation the solver was used with a mesh of 10x10, quadrilateral elements Q2Q1, a Reynolds number of 100 and tolerance of 10^{-12} for the convergence of the solution. The results are displayed below.



(a) (b)

Figure 2.1: Vector field (a) and pressure field (b) solved with Picard's method



(a) (b)

Figure 2.2: Vector field (a) and pressure field (b) solved with Newton-Raphson's method

The achieved results are virtually the same and consistent with the problem. However, the methods can be differed by their convergence rates, shown below:

Newton-raphson converged much faster, requiring only 5 iterations to reach a residual of four orders of magnitude lower than the tolerance. Whereas the Picard's method took four times the number of iterations and two times the computational cost (1,49s against 2,39s).

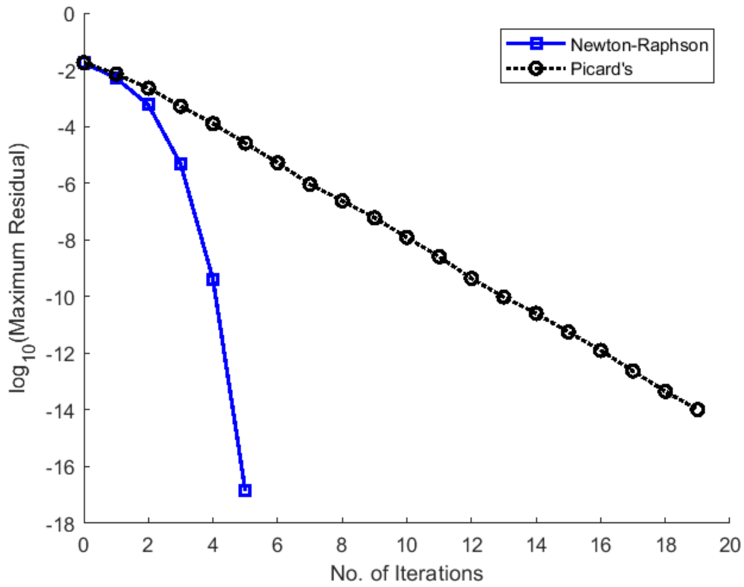


Figure 2.3: Convergence rate comparison