UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Escola Superior d'Enginyeries Industrial,
Aeroespacial i Audiovisual de Terrassa

BACHELOR THESIS

Bachelor's degree in Aerospace Vehicle Engineering

Heat and Mass Transfer Technological Center

Study for the numerical resolution of the Navier-Stokes equations using the Fractional Step Method

REPORT

Polytechnic University of Catalonia
ESEIAAT

Written by:
Anna Feliubadaló Rubio
Call: 1

Director:
Carlos David Pérez Segarra

Co - Director:
Asensio Oliva Llena

Delivery Date:
10th June 2018

Agraïments

Voldria agrair el suport que he rebut per part del CTTC (Centre Tecnològic de Transferència de Calor i Massa) per haver-me donat la oportunitat de dur a terme aquest estudi i per brindar-me tot el suport tècnic necessari.

En especial, voldria expressar la meva enorme gratitud al Carlos David Pérez Segarra, el director d'aquest estudi, per la seva ajuda, la seva orientació i guia pacient, per les seves crítiques, per la seva ajuda per tal de mantenir el meu progrés amb ritme, pel seu assessorament incondicional i per estar sempre disposat a atendre'm. També m'agradaria donar les gràcies al Jorge Chiva, per la seva immensa paciència per revisar i corregir els meus codis.

D'altra banda, vull donar les gràcies a totes aquelles persones que m'han ajudat a arribar fins aquí: a tots els Bergants, per fer-me gaudir de l'etapa universitària i per ensenyar-me tantíssimes coses. Però sobretot... gràcies a la meva família i al Pau, per fer el camí més amè.

Contents

List of Figures

List of Tables

Glossary

2D Two-Dimensional. 12

CDS Central Difference Scheme. 40

CFD Computational Fluid Dynamics. 14

CFL Courant Friedrichs Lewy. 78

CTTC Heat and mass transfer technological center. 11

CV Control Volume. 14

EDS Exponential Difference Scheme. 40

FSM Fractional Step Method. 67

FVM Finite Volume Method. 14

HH The Helmholtz-Hodge Theorem. 68

NS Navier-Stokes. 13

P Péclet non-dimensional number. 54

PLDS Power Law Difference Scheme. 43

QUICK Quadratic Upstream Interpolation for Convective Kinematics. 40

Re Reynolds non-dimensional number. 72

TDMA Tri-Diagonal Matrix Algorithm. 17

UDS Upwind Difference Scheme. 40

Chapter 1

Introduction

1.1 Resume

This work is denominated *Study for the numerical resolution of the Navier-Stokes equations using the Fractional Step Method*. It is divided into 7 Chapters and 1 Annex.

Chapter 1 presents the objectives, scope, requirements and background of the work and, additionally, the state of the art. Chapter 2 is dedicated to the introduction to numerical methods, while Chapter 3 covers the analysis of conduction heat transfer in solids. Chapter 4 covers different resolution methodologies of the Convection-Diffusion equation. An introduction to the numerical resolution of the Navier-Stokes equations through the Fractional Step Method is presented in Chapter 5. Finally, the environmental impact and some conclusions are presented in Chapter 6 and 7, respectively.

In addition, one annex is presented. It includes additional graphs and the own developed computational codes.

1.2 Aim

The main objective of this work is to perform a study of the resolution of fundamental equations of fluid dynamics and heat and mass transfer. Furthermore, the goal of this study is also to extend and consolidate the knowledge in thermodynamics, heat transfer, Computational Fluid Dynamics, etc. as well as their applications in the aerospace engineering field.

The basic methodology of the study is the numerical simulation of the phenomenology of fluid dynamics and heat transfer. Reference cases will be proposed in order to verify the developed code and the obtained solutions. Whenever necessary, the mathematical models will be validated based on the results, experimental data or simulations obtained by the Heat and Mass Transfer Technological Center (CTTC).

In a second phase, and depending on the skills developed, an application will be developed in a specific field.

1.3 Scope

The study includes the following tasks and objectives:

- Study of the state of the art of numerical simulation in the field of fluid dynamics and heat and mass transfer.

- Understanding and implementation of different numerical schemes and solvers.
- Study and discretization of conduction heat transfer problems under steady and unsteady conditions.
- Study and discretization of the generic convection - diffusion equation.
- Solve reference cases in order to verify the developed codes and the numerical solutions obtained.
- Implementation of a code to solve the Navier-Stokes equations.
- Validation and verification of the codes: enough accuracy, good programming style...
- Possible application to a selected engineering problem.
- Writing of possible improvements and future expansion.
- Writing a project planning and budget.

1.4 Requirements

The basic requirement is the development of well verified codes for the resolution of the different cases. Furthermore, the validation and optimization of the codes is required. Other requirements for the study are listed below:

- The programming language used to develop the code will be *Matlab*.
- The mathematical method used to solve the differential equations is the FVM.
- The domain and the geometries will be two-dimensional (2D).

1.5 Background

Engineers in the past had to rely on analytical skills to solve significant engineering problems and thus, had to undergo a rigorous training on mathematics. During the last decades, the ready availability of high speed computers has had a big impact on engineering practice. Numerical simulation has come to play an important role in the analysis and design of engineering processes.

For many heat transfer problems it is not possible to obtain a solution by means of analytic techniques. Instead, solving them requires the use of numerical methods, which in many cases allow such problems to be solved quickly. In addition, it is easily to see the effect of changes in parameters when modelling a problem numerically.

Furthermore, nowadays, industrial companies invest large amounts of money and time on prototyping to simulate and test their products in order to improve and optimize them. These products are tested in laboratories with high-technical instruments to measure different physical properties. These procedures to test products are, almost always, too expensive for businesses. For that reason, CFD programs are increasing in strength as they are a cheaper option for companies.

Due to the reasons presented above, the developing of this study is justified by the need of learning the methods used to solve problems involving heat and mass transfer, understand the physics behind their formulation and being able to model real cases.

1.6 State of the art

The Navier-Stokes (NS) equations are a mathematical model aimed at describing the motion of an incompressible viscous fluid, like many common ones as, for instance, water, oil and, under certain circumstances, also air. They were introduced in 1822 by the french engineer Claude Louis Marie Henri Navier and re-obtained, by different arguments, by George Gabriel Stokes in 1845. We refer the reader to the paper by Olivier Darrigol [11], for a detailed analysis of the history of the NS equations.

Even though, the interest in fluid dynamics goes back to the oldest applications of fluids in engineering. History can show us examples of the application of fluid dynamics [12]:

- Already in the Roman Empire with the construction of machines and hydraulic systems: the aqueducts.
- During the first century B.C., the engineer and architect Vitrubio invented the horizontal hydraulic wheel, which revolutionized the technique of grinding grain.
- After Archimedes, it took more than 1600 years before the next significant scientific breakthrough occurred, due to Leonardo da Vinci, who provided the first equation of mass conservation and developed multiple hydraulic and aerodynamic mechanisms.

Among all these formulations, the NS equations stand out, a set of partial non-linear derivative equations responsible for describing the movement of a viscous newtonian fluid. The analytical solutions to these equations are few and for very specific cases. Nowadays, the great advantage is the possibility of combining different techniques to solve the NS equations by means of numerical methods in order to solve more complex problems.

In this area, and with the technological advances in the computing field, it was possible to develop programming codes based on numerical resolution, known as CFD, capable of solving the mass and heat transfer with relative ease and also the movement of a fluid in laminar regime case. The actual case of study and main branch of research and computational expenditure will be, probably, the movement of a fluid in turbulent regime.

Chapter 2

Introduction to numerical methods

Computational Fluid Dynamics (CFD) carries out solving flows and related phenomena which can be described by partial differential equations and, commonly, they cannot be solved analytically. The techniques for solving differential equations based on numerical methods were developed before programmable computers existed. During World War II, it was common to find rooms of people, usually women, working on mechanical calculators to numerically solve systems of differential equations for military calculations [18]. Nowadays, the availability of computers and softwares have had a huge impact on engineering practice and numerical simulation has an important role in the resolution of engineering problems.

A numerical method is a tool designed to solve numerical problems. This science tries to obtain an approximated solution by using the discretization concept and method to approximate differential equations by a system of algebraic equations. These discretized equations are applied to small domains in space in order to obtain results at discrete locations in space over time. We have different methods to convert our equations into different discrete algebraic equations. The most important approach methods are the Finite Difference, the Finite Volume and the Finite Element methods. The chosen method for this study will be the Finite Volume Method.

2.1 Finite Volume Method

The Finite Volume Method (`FVM`) is a numerical technique which transforms the partial differential equations representing conservation laws over differential volumes into discrete algebraic equations over finite volumes (or elements or cells) [7].

2.1.1 Domain discretization

The first step in the solution process is the discretization or division of the geometric domain into finite control volumes (CV). The geometric discretization of the physical domain results in a mesh on which the conservation equations are eventually solved. This requires the subdivision of the domain into discrete cells or elements that completely fill the computational domain to yield a grid or mesh system [7].

The FVM structural discretization of the domain for Cartesian Coordinate system is shown in Figure 2.1. The different control volumes have a $\Delta x \cdot \Delta y$ dimension and they are classified according to their neighbouring nodes: The central node is named P and the neighbouring ones are called N (north), S (south), E (east) and W (west) nodes.

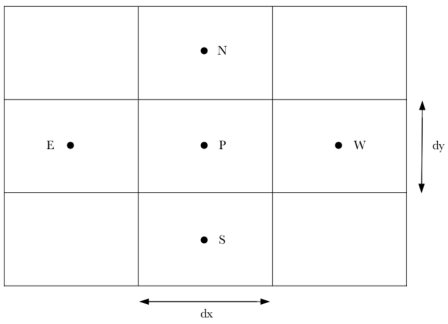


Figure 2.1: Nomenclature of FVM.

The CV faces n, s, e and w are placed just between the grid point P and its neighbours.

There are different kinds of spatial meshes:

1. Structured mesh: This mesh is characterized by a regular connectivity. Structured meshes allow programming codes more efficiently than unstructured meshes, although they can only be used for geometrically simple solution domains.
   - Uniform mesh: All elements have the same dimensions and are equally separated.
   - Non-Uniform mesh: The distances between elements are different.
   - Orthogonal
   - Non-Orthogonal
2. Unstructured mesh: This mesh is characterized by irregular connectivity.
3. Hybrid mesh: It contains structured and unstructured portions.

The meshes used in the thesis will be orthogonal, and uniform or non-uniform, depending on each problem.

2.1.2 Temporal discretization schemes

The partial differential equations are transformed into algebraic equations by integrating them over each discrete element.

When the general discretization equation is obtained, the term β, which is a weighting factor between 0 and 1, can be added. *"For certain specific values of the factor β, the discretization equation reduces to different temporal discretization schemes"* [14]. In particular:

- $\beta = 0$ leads to the explicit scheme
- $\beta = 1$ leads to the fully implicit scheme
- $\beta = 0.5$ leads to the Crank-Nicolson scheme

As it is said in Reference [14], the different values of β can be interpreted in terms of the variations of $T_P \sim t$.

Explicit Scheme ($\beta = 0$)

The finite differences are taken at the beginning of the step. This scheme assumes that the old value of T_P^0 prevails throughout the entire time step except at time $t + \Delta t$. With this scheme it is not necessary the resolution of a non-trivial system of equations at each step of time.

This resolution system is of first order in time. The step criteria is given by numerical, not physical, conditions. To estimate the time interval, a value smaller than a characteristic dimension of the mesh should be taken.

Fully Implicit Scheme ($\beta = 1$)

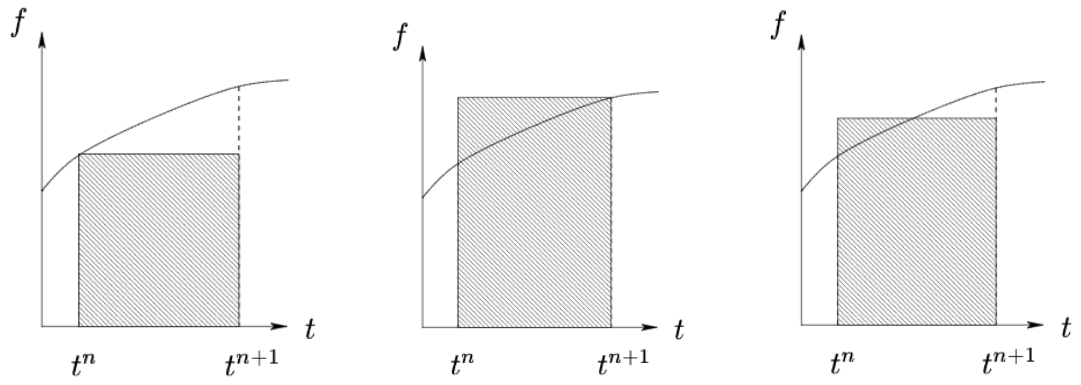
The finite differences are taken at the end of the step. The resolution implies the knowledge of the unknowns and this results in an iterative process with a larger number of calculations per time step. However, this is balanced by the fact that the process is stable for all sizes of time step and thus, it is unconditionally stable.

This scheme postulates that, at time t, T_P suddenly drops from T_P^0 to T_P^1 and stays with this value over the whole time step. It involves a first order error and requires a solver of equations since the unknown parameter needs to be known in order to proceed with the calculation.

Crank-Nicolson Scheme ($\beta = 0.5$)

Assumes a linear variation of T_P. This solution is very accurate and unconditionally stable. It requires a solver since the calculated temperature and the new unknown variable are required in order to do calculations.

As an advantage over the other schemes, the error committed using this scheme is a second order error.



(a) Schema of explicit method. (b) Schema of implicit method. (c) Schema of CN method.

Figure 2.2: Temporal discretization schemes [17].

2.2 Solvers

Once the volume discretization is done and an appropriate finite difference equation is written for each node, it is possible to determine the temperature distribution. The initial problem is reduced to solving a system of linear algebraic equations. Numerous methods are available for this purpose and are classified as:

- Direct methods: This methods involve a predetermined number of arithmetic operations and their use is adequate when the number of equations is small.
- Iterative methods: This methods use an initial guess to generate a sequence of improving approximate solutions for a class of problems. Although it is not possible to predetermine the required number of arithmetic operations, iterative methods are characterized by reduced computer requirements and are especially appropriate when the number of equations is large.

In this section we consider the Tri-Diagonal Matrix Algorithm (TDMA) and the Gauss-Seidel method as examples of the direct and iterative methods, respectively.

2.2.1 Tri - Diagonal Matrix Algorithm

The TDMA method is a direct resolution algorithm. Consider a system of N algebraic equations generated by finite differences that correspond to N unknown temperatures. We can write:

$$Ax = B$$

Where A is the coefficient matrix, x is the unknowns vector and B is the vector corresponding to independent terms.

In some applications, we find systems where A is a square matrix and its elements are all zero except those of the main diagonal and some of the parallels to this diagonal. A particular case of these matrices are tri-diagonal matrices, those whose all non-zeros coefficients align themselves along three diagonals of the matrix: the main diagonal and both adjacent to it.

We can now suppose a mesh with N aligned nodes, were node 1 and N denote the boundary points. The discretization equations can be written as:

$$a_i \cdot T_i = b_i \cdot T_{i+1} + c_i \cdot T_{i-1} + d_i \tag{2.1}$$

As Equation 2.1 shows, the temperature T_i is related to the neighboring temperature T_{i+1} and T_{i-1}. In order to explain the TDMA method in a simply way, we will start from the discretized equation for a one-dimensional temperature calculation case:

$$a_P \cdot T_P = a_E \cdot T_{i+1} + a_W \cdot T_{i-1} + b_P \tag{2.2}$$

If we group terms and develop the equation we see that we can write the equation as:

$$T_i = P_i \cdot T_{i+1} + R_i \tag{2.3}$$

where

$$P_i = \frac{a_{Ei}}{a_{Pi} - a_{wi} \cdot P_{i-1}}$$

and

$$R_i = \frac{b_{Pi} + a_{Wi} \cdot R_{i-1}}{a_{Pi} - a_{wi} \cdot P_{i-1}}$$

2.2.2 Gauss - Seidel method

The Gauss - Seidel method is an iterative solver. With this method, the values of the variable are calculated by visiting each grid point in a certain order. It is based on the trial and error method and though, the algorithm must include a step in which an initial guess value is assigned to all temperatures. Starting with an arbitrary guess, we are able to approach the correct solution of the equations. Once the variable for the whole domain has been calculated, the obtained value at each point is compared with the assumed value (or last calculated) and the calculation is repeated until this difference is as small as the established precision.

2.2.3 Line-by-Line method

The line-by-line method is an iterative resolution algorithm that combines both previous methods. We need to choose a grid line (either x or y-direction) and assume that the temperatures along the neighboring lines are known (guessed or last calculated values) and solve the chosen line by TDMA. Once the line is solved, results are compared with the initial guess values, as in the Gauss-Seidel algorithm.

The required equation for solving horizontal lines with vertical sweeping direction with line by line method is:

$$a_P T_P = a_E T_E + a_W T_W + b_P^{**} \tag{2.4}$$

where:

$$b_P^{**} = a_N T_N + a_S T_S + b_P \tag{2.5}$$

Applying the TDMA method as many times as rows has the domain, we obtain the following P and R parameters:

$$P(i,j) = \frac{a_E(i,j)}{a_P(i,j) - a_w(i,j) \cdot P(i-1,j)} \tag{2.6}$$

$$R(i,j) = \frac{b_P(i,j)^{**} + a_W(i,j) \cdot R(i-1,j)}{a_P(i,j) - a_w(i,j) \cdot P(i-1,j)} \tag{2.7}$$

2.2.4 The relaxation factor

We can improve the convergence of the iterative algorithm by means of the relaxation method. This method allows us to speed up or to slow down the changes from one iteration to the next one (over-relaxation and under-relaxation, respectively). It consists on introducing a parameter, f_R, that can accelerate the process. Once we have calculated the temperature field by means of the Gauss-Seidel method, we modify the value with a linear combination of the results of the previous and current iteration:

$$T_{new}(i,j) = T_{old}(i,j) + f_R \cdot [T_{new}(i,j) - T_{old}(i,j)] \tag{2.8}$$

Where f_R can take the values between 0 and 2:

- For $0 < f_R < 1$ it is called under-relaxation.
- For $f_R > 1$ it is called over-relaxation. This method is reduced to the Gauss-Seidel method if $f_R = 1$.

Chapter 3

Conduction heat transfer

Heat is the form of energy that can be transferred from one system to another as a result of temperature difference (see reference [19]). Heat can be transferred in three different modes: conduction, convection and radiation.

The objective of this chapter is to make a first approach to the conduction heat transfer and to study and understand the physical phenomena involved on it.

3.1 Mathematical formulation

Conduction is the transfer of energy from the more energetic particles of a substance to the less energetic ones as a result of interactions between them.

When discretizing the conduction heat transfer equations, we must differentiate between one-dimensional and two-dimensional cases, as well as between steady and transient states.

One - dimensional steady conduction

We begin applying the first principle of thermodynamics to all nodes. The energy balance tells us that the heat that enters the CV through face w less the heat that goes out through face e, plus the heat generated inside must be zero, since we are considering steady state.

$$Q = Q_w - Q_e + Q_{VP} \quad = \quad 0$$

According to the Fourier's law, we know that:

$$-\lambda \left.\frac{dT}{dx}\right|_w S_w + \lambda \left.\frac{dT}{dx}\right|_e S_e + q_{VP} V_P \quad = \quad 0$$

In order to approximate the value of derivatives, we use a development in Taylor series of the second order. If we replace the result of this development, we obtain:

$$-\lambda \frac{T_P - T_W}{\Delta x} S_w + \lambda \frac{T_E - T_P}{\Delta x} S_e + q_{VP} V_P \quad = \quad 0$$

If we develop the previous expressions, we see that all the equations have the same general form:

$$a_P \cdot T_P = a_W \cdot T_W + a_E \cdot T_E + b_P$$

where

$$a_I = \frac{\lambda_i S_i}{\Delta x}$$

The same procedure can be applied in the y-direction with both north and south nodes.

Two - dimensional transient conduction

To set the two-dimensional equation we can not only apply the first principle of thermodynamics as in one-dimensional cases, we must add the transitory term. We start from the energy conservation equation (equation 3.1):

$$\frac{\partial}{\partial t}\int \rho u dV = -\int \overrightarrow{q}\cdot\overrightarrow{n}\,dS \tag{3.1}$$

This equation must be discretized establishing a mesh and the position of nodes within it. A distribution of centered nodes and an implicit resolution scheme will be used.

In addition, thermophysical properties will be assumed as constant. Keeping this in mind and starting from equation 3.1 we obtain:

$$\rho\frac{\partial}{\partial t}\int u dV = -\int \overrightarrow{q}\cdot\overrightarrow{n}\,dS \tag{3.2}$$

$$\overline{u} = \frac{1}{V_P}\int u dV \tag{3.3}$$

Considering that the average energy can be written as in equation 3.3 and also that:

$$-\int \overrightarrow{q}\cdot\overrightarrow{n}\,dS = \sum Q \tag{3.4}$$

we obtain:

$$\rho_P \cdot V_p \cdot \frac{\partial \overline{u_P}}{\partial t} = \sum Q \tag{3.5}$$

where the Q term corresponds to all conduction heat transfer flows that enter and leave the CV. Figure 3.1 shows the four heat flows associated with an internal cell.

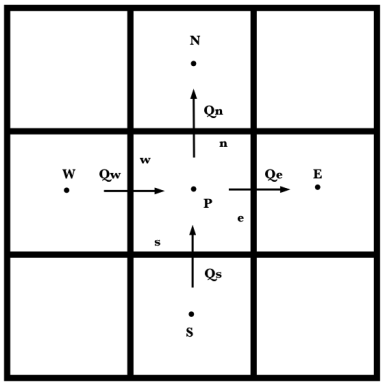


Figure 3.1: Heat flows associtated with an internal cell.

With this equation we must now proceed with the resolution of the time integral.

$$\int_{t^n}^{t^{n+1}} \rho_P \cdot V_p \cdot \frac{\partial \overline{u_P}}{\partial t} dt = \int_{t^n}^{t^{n+1}} \sum Q dt \tag{3.6}$$

$$\rho_P V_p \int_{t^n}^{t^{n+1}} \frac{\partial \overline{u_P}}{\partial t} dt = \left[\beta \cdot \left(\sum Q \right)^{n+1} + (1-\beta) \cdot \left(\sum Q \right)^{n} \right] \Delta t \tag{3.7}$$

$$\rho_P V_p \left(\overline{u}_P^{n+1} - \overline{u}_P^{n} \right) = \left[\beta \cdot \left(\sum Q \right)^{n+1} + (1-\beta) \cdot \left(\sum Q \right)^{n} \right] \Delta t \tag{3.8}$$

We apply the definition of internal energy to this equation to obtain:

$$\rho_P c_P V_p \left(T_P^{n+1} - T_P^{n} \right) = \left[\beta \cdot \left(\sum Q \right)^{n+1} + (1-\beta) \cdot \left(\sum Q \right)^{n} \right] \Delta t \tag{3.9}$$

For an implicit resolution scheme ($\beta = 1$), the equation will result as:

$$\rho_P c_P V_p \left(T_P^{n+1} - T_P^{n} \right) = \Delta t \cdot \left(\sum Q \right)^{n+1} \tag{3.10}$$

$$\frac{\rho_P c_P V_p}{\Delta t} \left(T_P^{n+1} - T_P^{n} \right) = (Q_s - Q_n + Q_w - Q_e)^{n+1} \tag{3.11}$$

From the Fourier's conduction law we obtain:

$$Q = -\lambda \frac{dT}{dx} S \tag{3.12}$$

Developing a second order approximation in equation 3.12 we obtain an expression for a general node, I:

$$\frac{dT}{dx} \simeq \frac{T_I - T_P}{dPI} \tag{3.13}$$

Applying this approximation to the energy equation we obtain:

$$\frac{\rho_P c_P V_p}{\Delta t} \left(T_P^{n+1} - T_P^{n} \right) = (-\lambda_s \frac{T_P - T_S}{dPS} S_s + \lambda_n \frac{T_N - T_P}{dPN} S_n - \lambda_w \frac{T_P - T_W}{dPW} S_w + \lambda_e \frac{T_E - T_P}{dPE} S_e)^{n+1} \tag{3.14}$$

Grouping terms, we obtain the final discretization equation:

$$\boxed{a_P T_P^{n+1} = (a_N T_N + a_S T_S + a_E T_E + a_W T_W + b_P)^{n+1}}$$

where: $T_N = T(i, j+1)$, $T_S = T(i, j-1)$, $T_E = T(i+1, j)$ and $T_W = T(i-1, j)$.

3.2 A Two-Dimensional Transient Conduction Problem

The main goal of this section is to develop a code for the resolution of a 2D transient conduction problem. To do this, numerical methods must be applied with the ultimate goal of achieving a solid basis of its methodology and to be able to expand it in future cases of more complexity. Therefore, the laws that govern the conduction heat transfer in solid bodies will be presented, as well as the procedure to be followed for its numerical resolution. See Reference [4] for the whole explanation of the problem.

3.2.1 Problem definition

A long rod is composed of four different materials (M1 to M4), represented with different colours in the figure below. All lines are parallel to the coordinate axis. The coordinates of the points P0 to P3 are given in Table 1. The properties of the materials are given in Table 2. Each of the four sides of the rod interacts with the surrounding in a different manner, as described in Table 3. The initial temperature field is T = 8.00 °C.

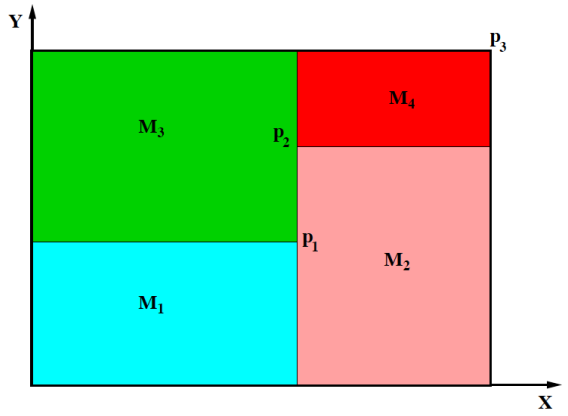


Figure 3.2: General schema of the proposed problem. Figure taken from [4].

Table 3.1: Problem coordinates

	x [m]	y [m]
P0	0.00	0.00
P1	0.50	0.40
P2	0.50	0.70
P3	1.10	0.80

Table 3.2: Physical properties.

	$\rho[kg/m^3]$	$c_p[J/kgK]$	$\lambda[W/mK]$
M1	1500.00	750.00	170.00
M2	1600.00	770.00	140.00
M3	1900.00	810.00	200.00
M4	2500.00	930.00	140.00

Table 3.3: Boundary conditions

Cavity wall	**Boundary condition**
Bottom	Isotherm at T = 23.00 °C
Top	Uniform $Q_{flow} = 60.00 W/m$
Left	In contact with a fluid at $T_G = 33.00$ °C and $\alpha = 9.00 W/m^2 K$
Right	Uniform temperature $T = 8.00 + 0.005 \cdot t$ °C (where t is the time in seconds)

The final aim is to obtain the temperature values at each point of the domain for each moment of time by means of a code in *Matlab* language. In order to facilitate the representation of results, the explanation will be focused on the temperature evolution of the following domain points: (0.65, 0.56) and (0.74, 0.72).

3.2.2 Resolution hypothesis

In order to be able to solve the exercise, many hypothesis have been formulated:

1. Two-dimensional study of the case, so that the equations do not consider the contribution of depth.
2. Transient study of the equations.
3. Constant thermophysical properties.
4. Implicit resolution scheme.
5. Different mesh density for each region.
6. Mesh scheme: Centered nodes.

3.2.3 Spatial discretization

In contrast to an analytic solution, which allows the determination of the temperature at any point of interest in a medium, a numerical solution allows to determine the temperature only in discrete points. The first step in any numerical analysis is, therefore, to select these points. The point of reference is usually called a nodal point. The nodal points are designated by a numerical scheme that is designated by the indices (i , j).

Consider the two - dimensional body that is divided into increments in the x and y directions, as shown in Figure 3.3. This rectangular mesh has computational cells of different sizes. The width and height of cell (i , j) are denoted by Δx_i and Δy_i, respectively. The temperature in the midpoint of cell (i,j) at the considered time is denoted by $T_{i,j}$.

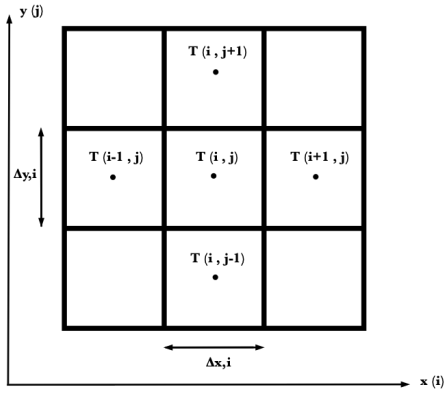


Figure 3.3: Choice of indices for the cells in the computational mesh.

We divide the domain of each region using the node centered method: We divide the total volume into N finite volumes aligned along the thickness given. Each CV will have the given height and the separation between them will be Δx_i. The next step is to locate the nodes in the middle of each CV. We proceed in the same way in the y - direction. The final scheme obtained can be seen in Figure 3.4.

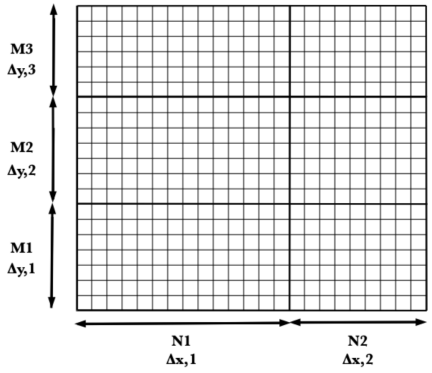


Figure 3.4: Division of domain with different mesh densities.

Once the volume has been discretized, we write N, S, E and W to refer to the nodes that remain above, below, to the right and left, respectively. The node we are dealing with is denoted as node P. In the same way, we will use n, s, e, w to refer to the surfaces of the CV, as we can see in Figure 2.1.

3.2.4 Calculation of coefficients

The values of the different coefficients can be obtained by grouping the terms of the previous equation. However, there is a certain number of nodes with defined boundary conditions and thus, they acquire special values that must be calculated separately, as follows.

1. Nodes of the bottom side: We can apply the *Dirichlet* boundary condition. All nodes are at a constant temperature of 23 ^{o}C. In the developed code, the a_I coefficients corresponding to these nodes are ignored and the temperature is imposed in the b_P coefficient. Thus, these nodes are eliminated from the iterative resolution process.

$$\begin{cases} a_N = 0 \\ a_S = 0 \\ a_E = 0 \\ a_W = 0 \\ a_P = 1 \\ b_P = T_{bottom} \end{cases}$$

2. Nodes of the right side: We can apply the *Dirichlet* boundary condition. All nodes are at a constant temperature. In the developed code, the a_I coefficients corresponding to these nodes are ignored and the temperature is imposed in the b_P coefficient ($b_P = T_{right}$). Thus, these nodes are eliminated from the iterative resolution process.

3. Nodes of the top side: We can apply the *Neumann* boundary condition. The nodes of this strip have a uniform incident heat flux on their upper face. Since the nodes of the north region have no influence, we impose $a_N = 0$.

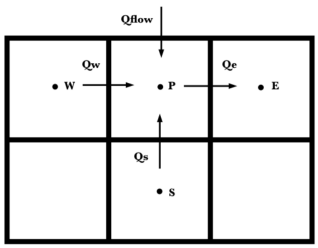


Figure 3.5: Discretization of the top side nodes.

$$a_N = 0$$
$$a_E = \frac{\lambda_e S_e}{dPE}$$
$$a_S = \frac{\lambda_s S_s}{dPS}$$
$$a_W = \frac{\lambda_w S_w}{dPW}$$

$$a_P = \frac{\rho_P c_P V_p}{\Delta t} + a_N + a_S + a_E + a_W$$

$$b_P = \frac{\rho_P c_P V_p}{\Delta t} \cdot T_P^n + Q_{flow} S_{flow}$$

4. Nodes of the left side: Since the nodes of this region do not have any node on the west side, we impose that $a_W = 0$. The other coefficients will obtain the original values.

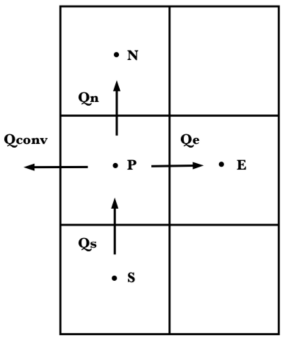


Figure 3.6: Discretization of the left side nodes.

$$a_N = \frac{\lambda_n S_n}{dPN}$$
$$a_E = \frac{\lambda_e S_e}{dPE}$$
$$a_S = \frac{\lambda_s S_s}{dPS}$$
$$a_W = 0$$

$$a_P = \frac{\rho_P c_P V_p}{\Delta t} + a_N + a_S + a_E + a_W + \alpha_G S_G$$

$$b_P = \frac{\rho_P c_P V_p}{\Delta t} \cdot T_P^n + \alpha_G T_G S_G$$

5. Interior nodes: They are the general nodes of the study domain, in which only heat transfer by conduction occurs.

$$a_N = \frac{\lambda_n S_n}{dPN}$$
$$a_E = \frac{\lambda_e S_e}{dPE}$$
$$a_S = \frac{\lambda_s S_s}{dPS}$$
$$a_W = \frac{\lambda_w S_w}{dPW}$$

$$a_P = \frac{\rho_P c_P V_p}{\Delta t} + a_N + a_S + a_E + a_W$$

$$b_P = \frac{\rho_P c_P V_p}{\Delta t} \cdot T_P^n$$

3.2.5 Resolution algorithm

The resolution algorithm on which the code is based to solve the equations is the following:

1. Introduction of the input data
   - Physical data: Data such as problem coordinates, physical properties, boundary conditions, initial conditions, etc.
   - Numerical data: Mesh scheme, time step, convergence criteria, etc.
2. Previous calculations: The code includes many functions in order to do some previous calculations such as to generate the mesh, to assign the corresponding physical properties to all generated CVs, to calculate the harmonic mean, etc.
3. Initial temperature map: This step includes a function created to assign initial values of temperature to all the points of the domain. The initial temperature for this problem is 8^{o}C.
4. Evaluation of the constant discretization coefficients: The coefficients that do not depend on temperature or time are calculated. These coefficients are a_N, a_S, a_E and a_W.
5. Calculation of the next time step: Once we have given an initial temperature, we calculate the new temperature field in $t^{n+1} = t^n + \Delta t$.
6. Evaluation of the discretization coefficients: We evaluate the b_P coefficients at each node (i , j).
7. Solve the set of equations: We solve the equation

$$a_P T_P^{n+1} = (a_N T_N + a_S T_S + a_E T_E + a_W T_W + b_P)^{n+1}$$

   point by point (Gauss - Seidel method).
8. Convergence: At this point, the difference between the temperature values at the current time and the previous one will be checked at each point (i , j) of the 2D domain. We need to ask the code if $max|T^{n+1} - T^{*(n+1)})| < \delta$. If the temperature meets this criteria, the code will stop calculating and will go to the next step of the algorithm. If not, the code will return to step number 6.
9. New time step. At this point, the code checks if there is need to calculate the temperature in a new time. If the answer is *Yes*, we go to step 5 with $t^n = t^{n+1}$ and $T^n(i,j) = T^{n+1}(i,j)$. If not, we go to next step [1].
10. Final calculations such as the temperature at the specific domain points (0.65, 0.56) and (0.74, 0.72).
11. Print results: We plot the evolution of the temperature through time, a color map to see the different isotherms along the rod, etc.
12. End

[1] Note that the denomination $t = n + 1$ is only valid when time - step is 1 s. The correct denomination is $t = n + \Delta t$.

3.2.6 Results

After the code is executed, it is validated with the temperature of the two coordinate points to make sure that everything is correct. The obtained results for the temperature field at t = 5000 seconds are shown below in Figure 3.7 and can be compared with the expected temperature field in Figure 3.8, provided by the *CTTC* in Reference [4].

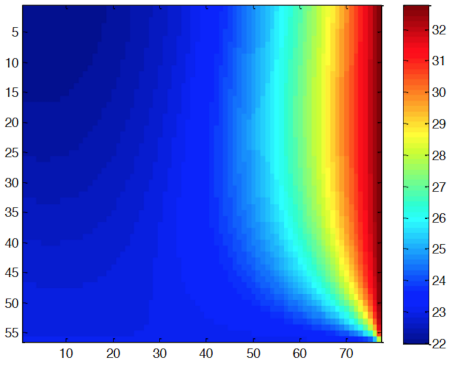


Figure 3.7: Obtained heat conduction solution at 5000 seconds.

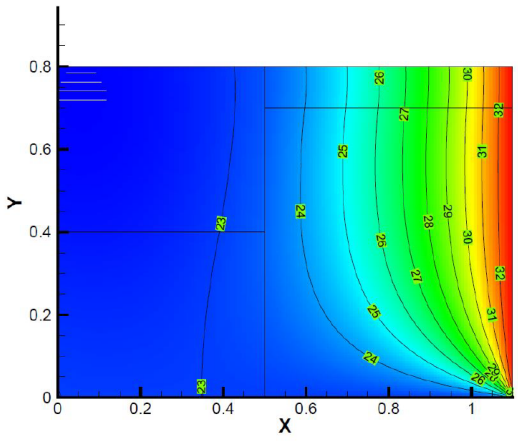


Figure 3.8: Expected heat conduction solution at 5000 seconds.

Figures 3.9 to 3.12 show the evolution of temperatures at the first and second requested points, respectively, through time analyzed with different mesh sizes.

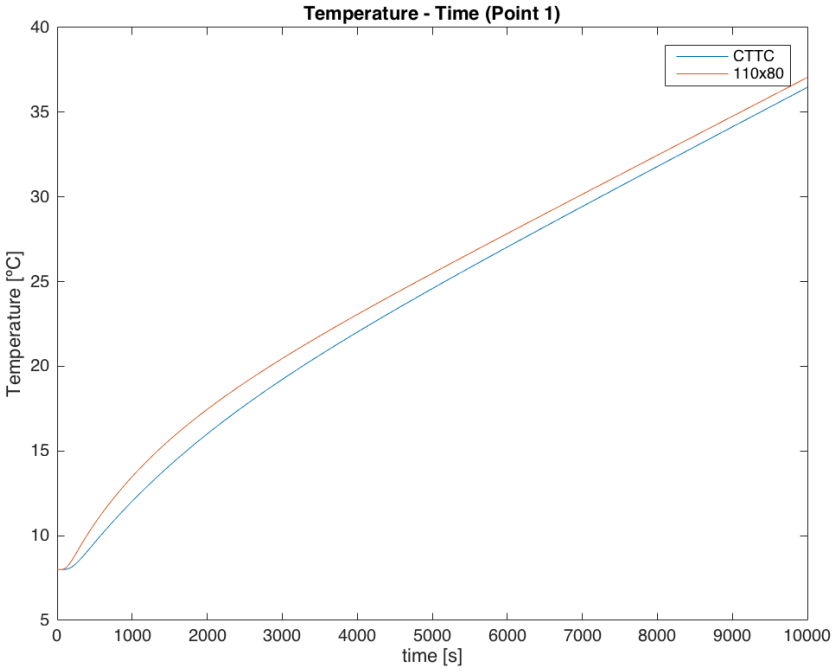


Figure 3.9: Comparison between the obtained results at Point 1 with mesh size 110x80

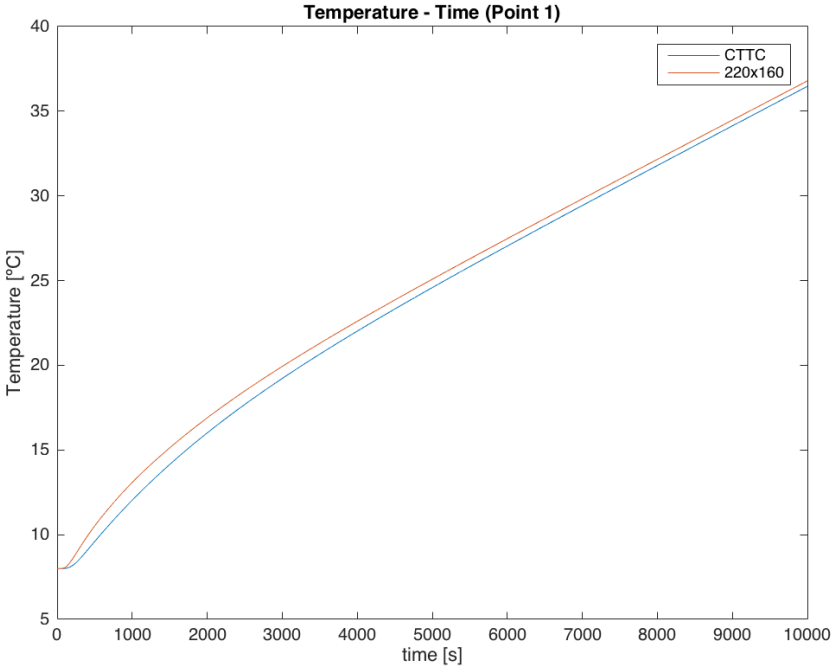


Figure 3.10: Comparison between the obtained results at Point 1 with mesh size 220x160

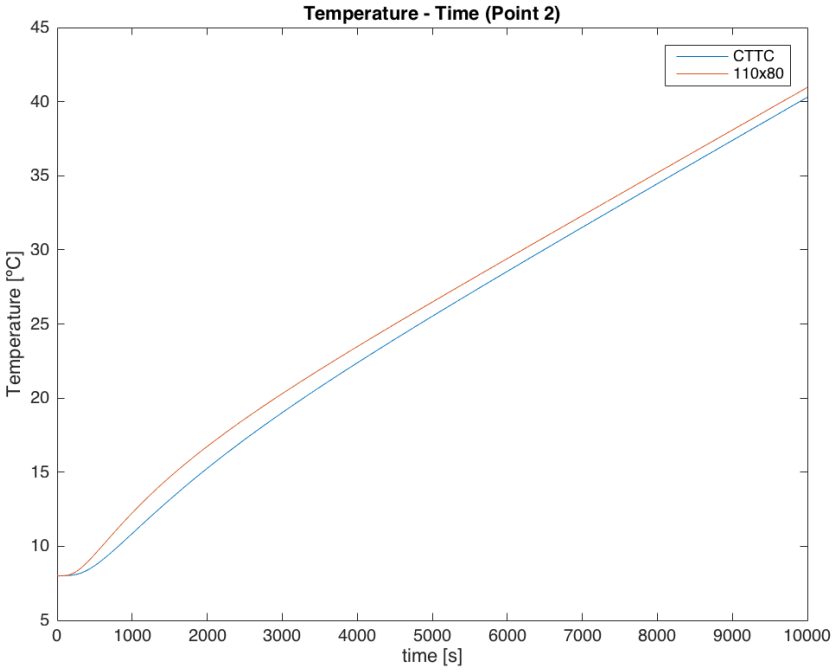


Figure 3.11: Comparison between the obtained results at Point 2 with mesh size 110x80

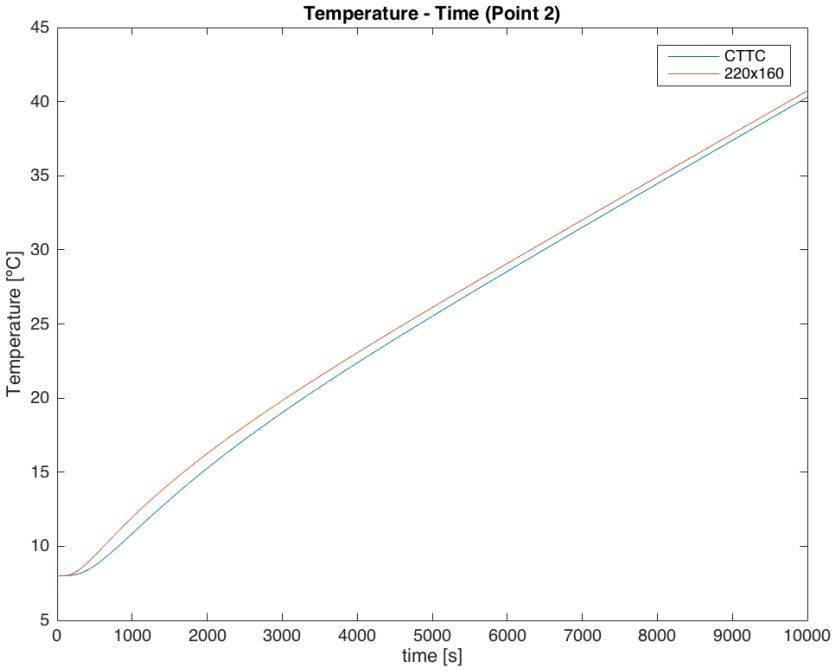


Figure 3.12: Comparison between the obtained results at Point 2 with mesh size 220x160

3.2.7 Verification

The numerical data used to solve the problem are the following:

- Mesh size: 110x80 and 220x160

- Time step: 1 s
- Convergence criteria: $5 \cdot 10^{-5}$

The isotherms at the domain points (0.65, 0.56) and (0.74, 0.72) are shown in Figure 3.7. These results match with the *CTTC* results shown in Figure 3.8. Even though, a study of the optimal mesh size has been undertaken.

Study of mesh size

As we can see in Figures 3.9 to 3.12, the most accurate results are obtained using N = 220 and M = 160, although the drawback is that the simulation lasts longer. Then, a compromise between number of nodes and relative fast time response would be appropiate. As we can see in Figure 3.13, when we double the mesh size in each direction (from 110x80 to 220x160) we only reduce the mean error value from 5 % to 3 %. Therefore, it can be proved that a 110x80 mesh is good enough to solve the case.

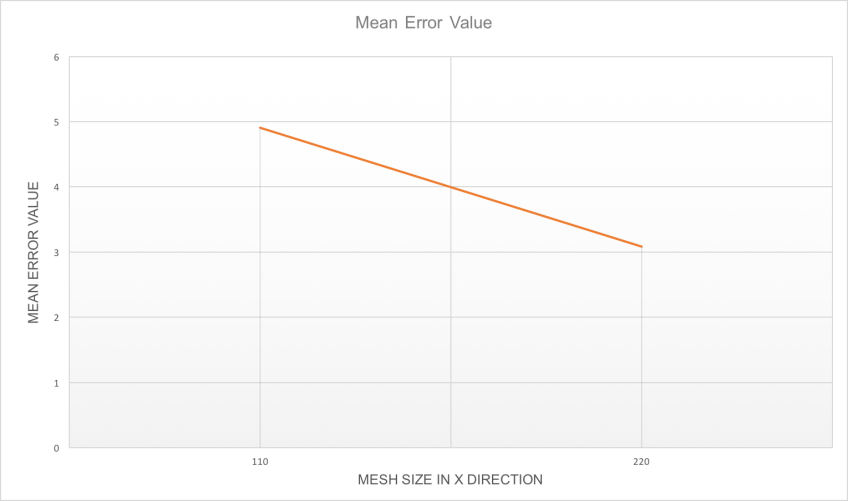


Figure 3.13: Comparison between the obtained results at Point 2 with mesh size 220x160

3.2.8 Conclusions

With Figures 3.7 and 3.8, we can conclude that the number of nodes is enough to obtain an accurate solution because the solution has a physical meaning due to the fact that the right side is at approximately 33ºC and the bottom side at 23ºC.

It can also be appreciated how temperature behaves as expected:

- Temperature gradient is stronger on the right wall due to the Dirichlet boundary condition and temperature on the right wall increases proportionally to time.
- At the top side there are isotherm regions with a perpendicular profile due to the Neumann boundary condition.
- At the lower wall, temperature remains constant in time.

- Materials 2 and 4 show the same thermal behaviour and this is justified because they both have the same thermal conductivity. The different temperature gradients can also be justified because of the different heat capacity and different density.
- Finally, as the temperature on the right wall always increases over time, the domain will never reach the steady state.

The temperature evolution of the specific points is shown in Figure 3.14. As it can be seen, Point 1 (closer to center) increases its temperature faster than the other point (closer to right wall) mainly because of the bottom wall being at higher temperature. Since the right wall increases its temperature continuously with time, there is a moment in which this tendency is inverted and Point 2 is the one increasing faster.

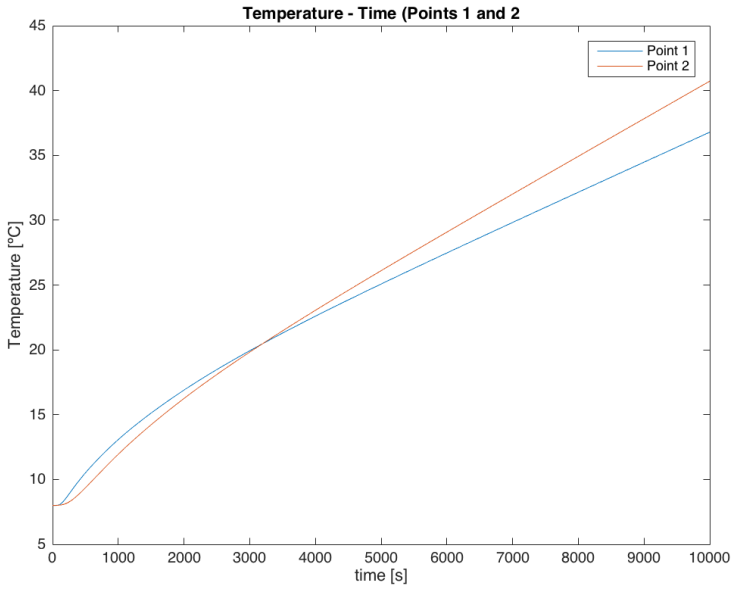


Figure 3.14: Comparison between the obtained results at Point 1 and 2.

3.2.9 Further applications

The proposed code can also solve the following case:

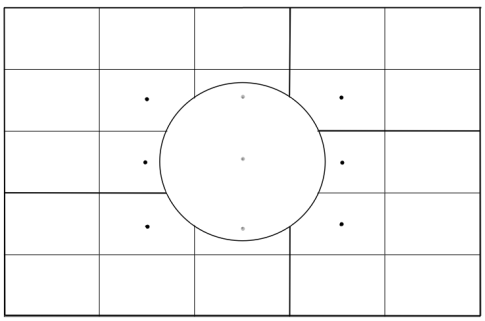


Figure 3.15: Further application: Long rod with a circular hole inside.

It is the same rod with a circular hole inside. Thanks to the implementation of an index that locates what material do we have at each position of the domain and thanks to the implementation of the calculation of the centroid of each CV, we can calculate the heat transfer through materials even if they have a hole in the middle, see Figure 3.15, or even if they have irregular shapes, see Figure 3.16.

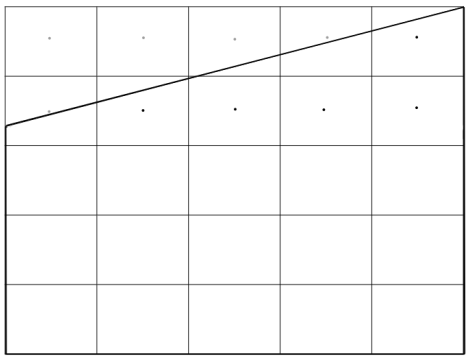


Figure 3.16: Further application: Long rod with irregular shape.

The code detects whether the centroid is within the domain or not and decides if it is necessary to perform the calculations on the involved CV.

3.2.10 Potential improvements

The proposed code could be improved with the incorporation of:

- Further validation: Both Crank-Nicolson and Explicit scheme could be employed to compare the results with the ones obtained with Implicit scheme. The algorithm would remain the same and the only change would be the value of β in the energy balance of internal, left-side and top-side nodes.
- Further validation with a study on convergence criteria and time step sensitivity.
- Include a new solver with TDMA Line-by-Line method.

Chapter 4

The convection - diffusion equation

This chapter provides an overview of the conservation principles governing fluid flow, heat and mass transfer and other related transport phenomena of interest in this thesis. The physical laws controlling the conservation principles are translated into mathematical relations. First, the continuity, momentum, and energy equations (collectively known as the Navier–Stokes equations) expressing the principles of conservation are presented. Finally, all these equations are summarized in the Convection - Diffusion equation [8].

The main aim of this chapter is to introduce and compare the numerical schemes applied to the convective terms in transport equations. The 2D convection–diffusion equation is a compact and non-physical model of transport of heat, mass, momentum, energy and other scalar magnitudes. The FVM applied to this equation allows us to do an approximation of the convection term and to compare it with different numerical schemes.

4.1 The Navier-Stokes Equations

We are already familiar with numerous conservation laws such as the Navier – Stokes laws of conservation, which describe the motion of fluids. The Navier – Stokes equations consist on:

- Mass conservation
- Momentum conservation
- Energy conservation

4.1.1 The mass conservation equation

The conservation of mass principle is simply a statement that mass cannot be created or destroyed during a process and all the mass must be accounted for during an analysis. In steady flow, the amount of mass within the CV remains constant, and thus the conservation of mass can be expressed in its differential form as:

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \vec{v}) = 0 \tag{4.1}$$

Integrating over the time and volume:

$$\int_{t^n}^{t^{n+1}} \int_{V_P} \frac{\partial \rho}{\partial t} dV dt + \int_{t^n}^{t^{n+1}} \int_{V_P} \nabla \cdot (\rho \vec{v}) dV dt = 0 \tag{4.2}$$

where ρ is the density, t is the time and $\vec{v}$ the is velocity. The meaning of the different terms of the equation are:

- **First term:** variation of mass in the CV in a differential of time.
- **Second term:** mass flow through the faces of the CV.

4.1.2 The momentum conservation equation

The principle of conservation of linear momentum [10], in absence of external forces acting on a body, indicates that the body retains the total momentum. The conservation of linear momentum can be expressed as:

$$\frac{\partial}{\partial t} (\rho \vec{v}) + \nabla \cdot (\rho \vec{v} \vec{v}) = -\nabla p + \nabla \cdot \vec{\tau} + \rho \vec{g} \tag{4.3}$$

where ρ is the density, t is the time, $\vec{v}$ the is velocity, p is the pressure, $\vec{\tau}$ is the total stress tensor and $\vec{g}$ is the gravitational acceleration. The meaning of the different terms of the equation are:

- **First term:** variation of the linear momentum in the CV.
- **Second term:** momentum flux through the faces of its CV.
- **Third term:** pressure gradient that acts on the faces of the CV.
- **Fourth term:** total stress tensor, a force acts axially and tangentially on the faces of the CV.
- **Fifth term:** volumetric force that may be a gravitational, electrical, magnetic... force.

See Reference [8] to know more about the body forces and the stress tensor.

4.1.3 The energy conservation equation

The conservation of energy is governed by the first law of thermodynamics, which states that energy can be neither created nor destroyed during a process. Consequently, the sum of all forms of energy in an isolated system remains constant [8].

$$\frac{\partial}{\partial t} (\rho (u + e_C)) + \nabla \cdot ((u + e_C) \rho \vec{v}) = -\nabla (p \vec{v}) + \nabla \cdot (\vec{v} \vec{\tau}) - \nabla \vec{q} + \rho \vec{g} \cdot \vec{v} + S_\Phi \tag{4.4}$$

where ρ is the density, t is the time, $\vec{v}$ the is velocity, p is the pressure, $\vec{\tau}$ is the total stress tensor, $\vec{g}$ is the gravitational acceleration, u and e_c are the internal and kinetic energy, respectively, $\vec{q}$ is the heat flow and S_Φ is the internal source. The meaning of the different terms of the equation are:

- **First term:** variation of the internal and kinetic energy in the CV.

- **Second term:** energy flow of these variables through the faces of its volume.
- **Third term:** work done by the superficial forces.
- **Fourth term:** work done by the superficial forces.
- **Fifth term:** incoming heat flow through the faces of the CV.
- **Sixth term:** work done by the volumetric forces.
- **Seventh term:** work done by the internal sources.

4.2 The Convection-Diffusion Equation

All the governing equations of the heat transfer by convection (equations 4.1.1, 4.1.2 and 4.1.3) can be summarized in the convection-diffusion equation:

$$\frac{\partial \rho\phi}{\partial t} + \nabla \cdot (\rho\vec{v}\phi) = \nabla \cdot (\Gamma \nabla\phi) + S_\phi \tag{4.5}$$

where:

- Γ is the diffusion coefficient and its quantity is specific to a particular meaning of ϕ.
- S_ϕ is the source term and its quantity is specific to a particular meaning of ϕ.

The four terms in the convection-diffusion equation are the unsteady term, the convective term, the diffusion term and the source term, respectively [14].

Table 4.1 shows how mass, momentum, energy and species conservation equations can be written using the convection-diffusion equation.

Table 4.1: Parameters to replace in convection-diffusion equation in order to reproduce the governing equations.

Equation	ϕ	Γ	S
Continuity	1	0	0
Momentum in the x direction	u	μ	$-\frac{\partial p_d}{\partial x}$
Momentum in the y direction	v	μ	$-\frac{\partial p_d}{\partial y} + \rho g \beta (T - T_\infty)$
Energy (with constant c_p)	T	$\frac{\lambda}{c_p}$	$\frac{\Phi}{c_p}$

4.2.1 Discretization equation for two dimensions

We proceed now to write the discretization equation to the general Equation 4.5. At first, let us consider the CV shown in Figure 4.1.

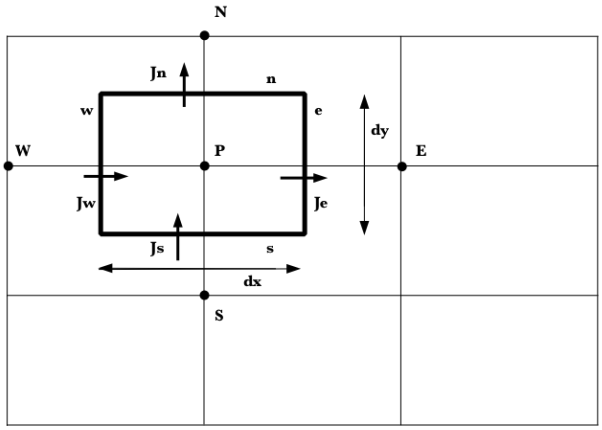


Figure 4.1: Used nomenclature to discretize the CV.

The 2D form of Equation 4.5 can be written as:

$$\frac{\partial}{\partial t}(\rho\phi) + \frac{\partial J_x}{\partial x} + \frac{\partial J_y}{\partial y} = S \tag{4.6}$$

where J_x and J_y are the total (convection and diffusion) fluxes defined by

$$J_x = \rho u\phi - \Gamma\frac{\partial \phi}{\partial x} \tag{4.7}$$

$$J_y = \rho v\phi - \Gamma\frac{\partial \phi}{\partial y} \tag{4.8}$$

The integration of Equation 4.6 over the CV shown in Figure 4.1 would give:

$$\frac{\left(\rho_P\phi_P - \rho_P^0\phi_P^0\right)\Delta x\Delta y}{\Delta t} + J_n - J_s + J_e - J_w = \left(S_C + S_P\phi_P\right)\Delta x\Delta y \tag{4.9}$$

where the source term has been linearized in the usual manner and, for the unsteady term, ρ_P and ϕ_P are assumed to prevail for the whole CV. We use an easier notation in which the different *old* and *new* values are denoted by $\phi^{n+1} = \phi$ and $\phi^n = \phi^0$.

The quantities J_n, J_s, J_e and J_w are the *integrated* total fluxes over the interface n, s, e and w.

In the same manner, we can integrate the continuity equation and obtain:

$$\frac{\left(\rho_P\phi_P - \rho_P^0\phi_P^0\right)\Delta x\Delta y}{\Delta t} + F_n - F_s + F_e - F_w = 0 \tag{4.10}$$

where the quantities F_n, F_s, F_e and F_w are the mass flow rates through the faces of the CV:

$$F_n = (\rho v)_n\Delta x \quad ; \quad F_s = (\rho v)_s\Delta x \quad ; \quad F_e = (\rho u)_e\Delta y \quad ; \quad F_w = (\rho u)_w\Delta y$$

If we know multiply Equations 4.10 by ϕ_P and substract it from Equation 4.10, we obtain:

$$\left(\phi_P - \phi_P^0\right)\frac{\rho_P^0 \Delta x \Delta y}{\Delta t} + (J_n - F_n\phi_P) - (J_s - F_s\phi_P) + (J_e - F_e\phi_P) - (J_w - F_w\phi_P) = (S_C + S_P\phi_P)\,\Delta x \Delta y \tag{4.11}$$

The assumption of uniformity over the CV faces enables us to employ the one-dimensional procedure:

$$J_e - F_e\phi_P = a_E\left(\phi_P - \phi_E\right) \tag{4.12}$$

$$J_w - F_w\phi_P = a_W\left(\phi_W - \phi_P\right) \tag{4.13}$$

Developing the equation we obtain the **final discretization equation:**

$$a_P\phi_P = a_N\phi_N + a_S\phi_S + a_E\phi_E + a_W\phi_W + b_P \tag{4.14}$$

where

$$a_N = D_n \cdot A\left(|P_n|\right) + max\left(-F_n, 0\right)$$
$$a_S = D_s \cdot A\left(|P_s|\right) + max\left(F_s, 0\right)$$
$$a_E = D_e \cdot A\left(|P_e|\right) + max\left(-F_e, 0\right)$$
$$a_W = D_w \cdot A\left(|P_w|\right) + max\left(F_w, 0\right)$$

$$a_P = a_N + a_S + a_E + a_W + \frac{\rho_P^0 \Delta x \Delta y}{\Delta t}$$

$$b_P = \frac{\rho_P^0 \Delta x \Delta y}{\Delta t} \cdot \phi_P^n + S_P \Delta x \Delta y$$

where:

$$D_n = \frac{\Gamma_n \Delta x}{d_{PN}}; \quad D_s = \frac{\Gamma_s \Delta x}{d_{PS}}; \quad D_e = \frac{\Gamma_e \Delta x}{d_{PE}}; \quad D_w = \frac{\Gamma_w \Delta x}{d_{PW}}$$

$$F_n = (\rho u)_n \Delta x; \quad F_s = (\rho u)_s \Delta x; \quad F_e = (\rho u)_e \Delta y; \quad F_w = (\rho u)_w \Delta y$$

and the Péclet number evaluated at the face of the CV (i) is:

$$P_i = \frac{F_i}{D_i}$$

The function $\cdot A\left(|P|\right)$ can be selected from Table 4.2 for the desired scheme.

Table 4.2: Value of $A\left(|P|\right)$ for different low numerical schemes.

Numerical Scheme	$A\left(\|P\|\right)$
UDS	1
CDS	$1 - 0.5\left(\|P\|\right)$
HDS	$max(0, 1 - 0.5\left(\|P\|\right))$
EDS	$\frac{\|P\|}{e^{\|P\|}-1}$
PLDS	$max(0, 1 - 0.1\left(\|P\|\right)^5)$

Now that we have the final discretization equation, we can start applying it to practical cases. The study of four cases related to the convection-diffusion equation is undertaken in the next sections. The variables ϕ, Γ and S are here general variables that satisfy mass, momentum and energy conservation equations.

4.2.2 Numerical Schemes

As we can see in the discretizated convection-diffusion equation, both convective and diffusive terms must be evaluated at the cell faces. This evaluation has to be expressed in terms of the nodal values by interpolation because the value of the dependent variable ϕ is known at the cell center. There are numerous available possibilities to do it, but some of the most known and used ones are the following ones [1].

Central Difference Scheme (CDS)

It is a second order scheme, variable at the cell face is calculated as an arithmetic mean. That is: The value of ϕ is calculated using a linear interpolation between the two neighbouring values.

$$\phi_e = \frac{1}{2}(\phi_E + \phi_P)$$

Upwind Difference Scheme (UDS)

It is a first order scheme and the value of ϕ at the cell face depends of the direction of the mass flow. It is a well-known low order numerical scheme because it has no convergence problems.

$$\phi_e = \phi_E \quad if \quad F_e > 0$$

$$\phi_e = \phi_P \quad if \quad F_e < 0$$

Exponential Difference Scheme (EDS)

It is a second order scheme and the evaluation of the dependent variable at the cell face comes from the exact solution of the convection-diffusion equation in one-dimensional, null source term and steady problem.

QUICK Scheme (QUICK)

This scheme consist in a quadratic interpolation in function of the direction of mass flow. Also is a combination of UDS and a high order interpolation, because depends of the direction of the mass flow.

To see the formulation of higher-order schemes please refer to Ferziger and Peric or Patankar [14].

[1] Explanations extracted from Reference [14].

4.3 Unidimensional flow with unidimensional variation of the variable solved in the same direction of the flow

The aim of this case is to develop a code for the resolution of a flow with a variation of the variable in the same direction of the flow. See Reference [6] for the whole explanation of the problem.

4.3.1 Problem definition

It is a steady state convection-diffusion problem in which the variable solved has a 1D variation in the same direction of the flow. Its analytical solution is known when the velocity field is given.

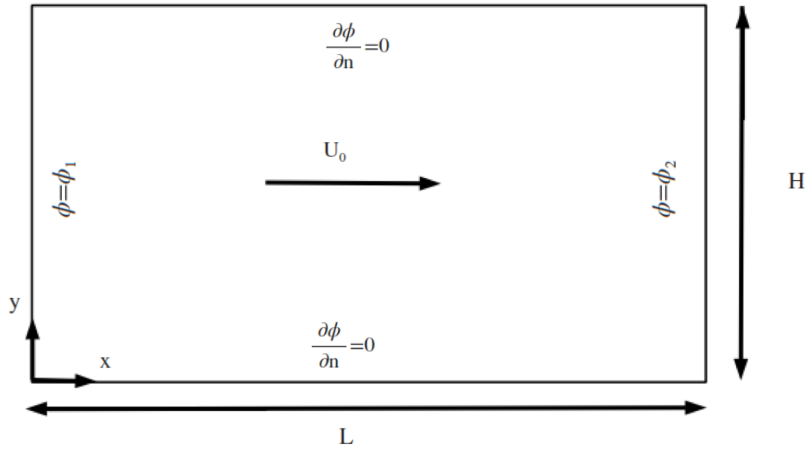


Figure 4.2: General schema of the proposed problem.

The velocity field is

$$u(x,y) = U_0$$
$$v(x,y) = 0$$

4.3.2 Resolution hypothesis

In order to be able to solve the exercise, many hypothesis have been formulated:

1. One - dimensional study of the case.
2. Steady study of the equations.
3. Constant thermophysical properties.
4. Implicit resolution scheme ($\beta = 1$).
5. Mesh scheme: Centered nodes.

4.3.3 Spatial discretization

We place faces along the wall and divide the total volume into N finite control volumes in the x-direction and M in y-direction, parallel to the walls and equidistant between nodes. We can see the result of the volume's discretization in Figure 4.3.

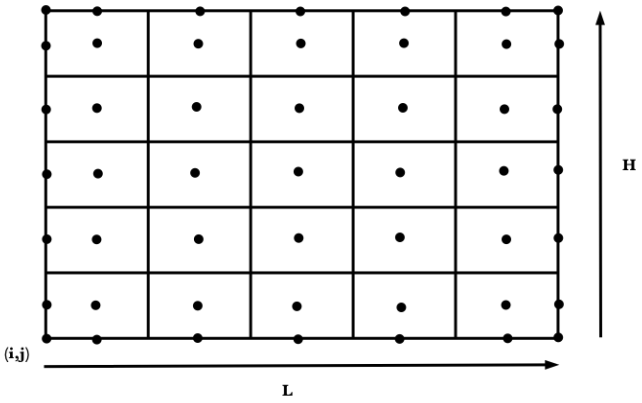


Figure 4.3: Schema of the domain discretization.

We place nodes at the top, bottom, left and right sides so that we finally have $N+2$ and $M+2$ nodes in each direction and it will be easily to impose our boundary conditions.

4.3.4 Calculation of coefficients

The values of the different coefficients can be obtained following Equation 4.14. However, there is a certain number of nodes with defined boundary conditions, so that these coefficients acquire special values that must be calculated separately, as follows.

1. Nodes of the left side: We can apply the *Dirichlet* boundary condition. All nodes are at a constant $\phi = \phi_1$.

2. Nodes of the right side: We can apply the *Dirichlet* boundary condition. All nodes are at a constant $\phi = \phi_2$

3. Nodes of the top and bottom side: We can apply the *Neumann* boundary condition

$$\frac{\partial \phi}{\partial n} = 0$$

of zero normal convective flow and impose

$$\phi_{new}(i,j) = \phi_{new}(i,j-1)$$

at the top side and

$$\phi_{new}(i,j) = \phi_{new}(i,j+1)$$

at the bottom one.

4. Interior nodes: They are the general nodes of the study domain which can be obtained following the previously developed equations.

4.3.5 Resolution algorithm

The resolution algorithm to solve the equations and on which the code is based is the following:

1. Introduction of the input data
   - Physical data: Data such as problem coordinates, physical properties, boundary conditions, initial conditions, etc.
   - Numerical data: Mesh scheme, time step, convergence criteria, maximum number of iterations, etc.
2. Previous geometry calculations: The code includes many functions in order to do some previous calculations such as to generate the mesh, to calculate Δx, Δy, V_P, S_i, etc.
3. Initial map: We assign initial values of ϕ to all the points of the domain.
4. Previous calculations of physical properties: The code includes a function that calculates the different convective conductances (F_n,F_s,F_e,F_w), diffusion conductances (D_n,D_s,D_e,D_w) and the local Péclet numbers (P_n,P_s,P_e,P_w).
5. Evaluation of the discretization coefficients: All the coefficients are calculated (a_N, a_S, a_E, a_W, a_P and b_P). This function differentiates the calculation of the coefficients according to the numerical scheme desired (CDS, UDS, EDS or PLDS).
6. Solve the set of equations: We solve the equation

$$a_P\phi_P = a_N\phi_N + a_S\phi_S + a_E\phi_E + a_W\phi_W + b_P$$

   with the Gauss - Seidel method.
7. Convergence: At this point, the difference between the new ϕ values and the ones obtained in the previous iteration are checked at each point (i , j) of the domain. We should check the following boolean expression: $max|\phi(i,j) - \phi^*(i,j)| < \delta$. If this criteria is met, the code will stop calculating and will move on to the next step of the algorithm. If not, the code will return to step number 5.
8. Final calculations, such as the analytical ϕ.
9. Print results
10. End

4.3.6 Results and Verification

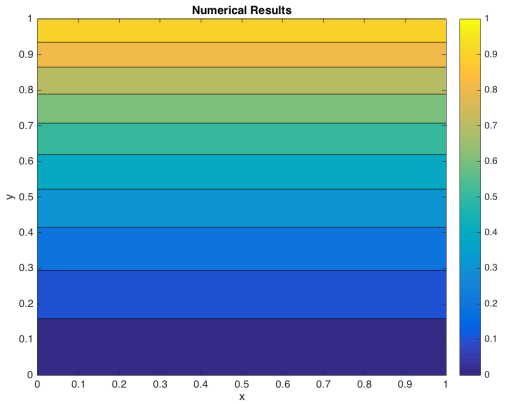
The solution for this first case is an exponential function for any value of U_0. If the numerical scheme used is the Exponential Difference Scheme (EDS) or the Powerlaw Difference Scheme (PLDS) as in this case, the numerical result must match perfectly with the theoretical result, which follows the equation:

$$\frac{\phi - \phi_0}{\phi_L - \phi_0} = \frac{e^{\frac{Px}{L}} - 1}{e^P - 1} \tag{4.15}$$

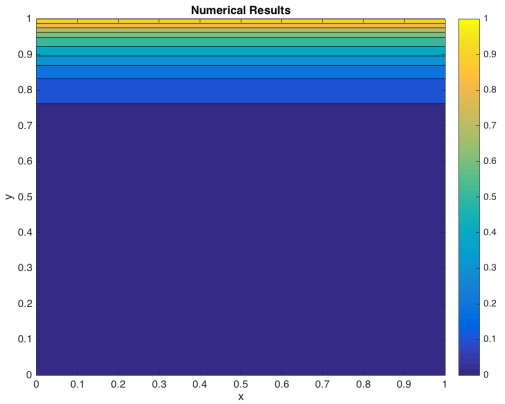
Where P is the dimensionless number of Péclet. The Péclet number [14] is defined as the *"ratio of convection transport rate of a physical quantity to its diffusive transport rate."* When dealing with heat transfer, the Péclet number is given by

$$P = \frac{\rho u L}{\Gamma}$$

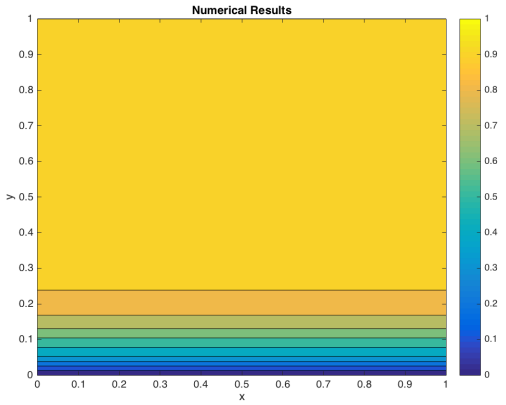
As it can be seen in Equation 4.15, results depend strongly on the Péclet number, which relates the importance of the convection of a fluid versus its diffusion (either thermal or mass diffusion). In this way, different results are obtained and analyzed in the following figures.



(a) Numerical results of ϕ field with P=1.

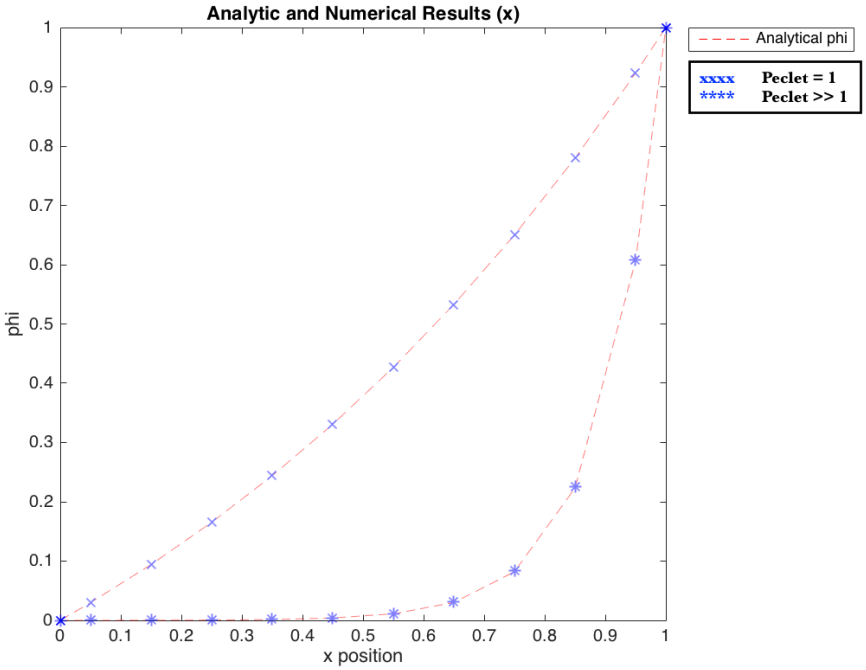


(b) Numerical results of ϕ field with P $\gg$ 1.

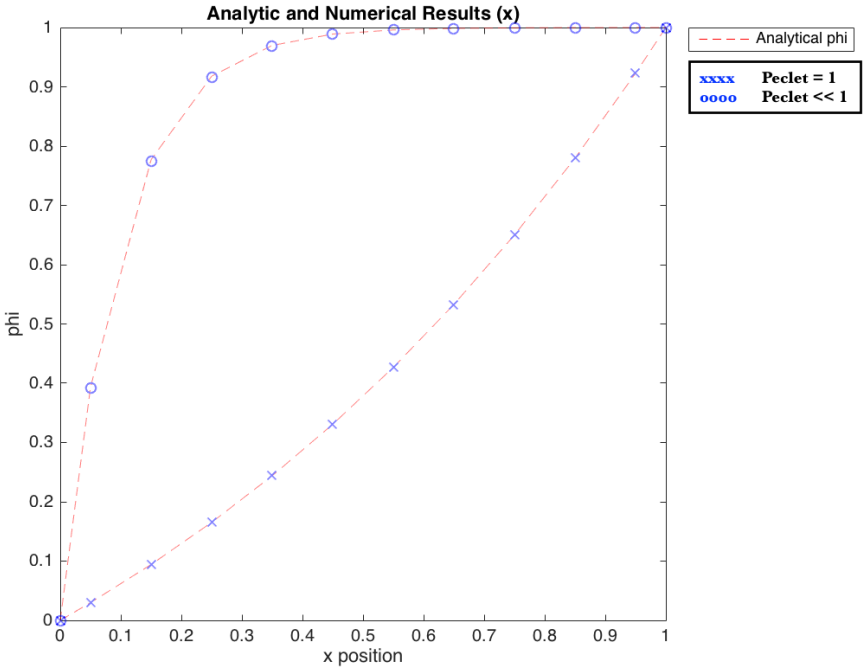


(c) Numerical results of ϕ field with P $\ll$ 1.

Figure 4.4: Comparison between different numerical results with a 10x10 mesh and different Péclet numbers.

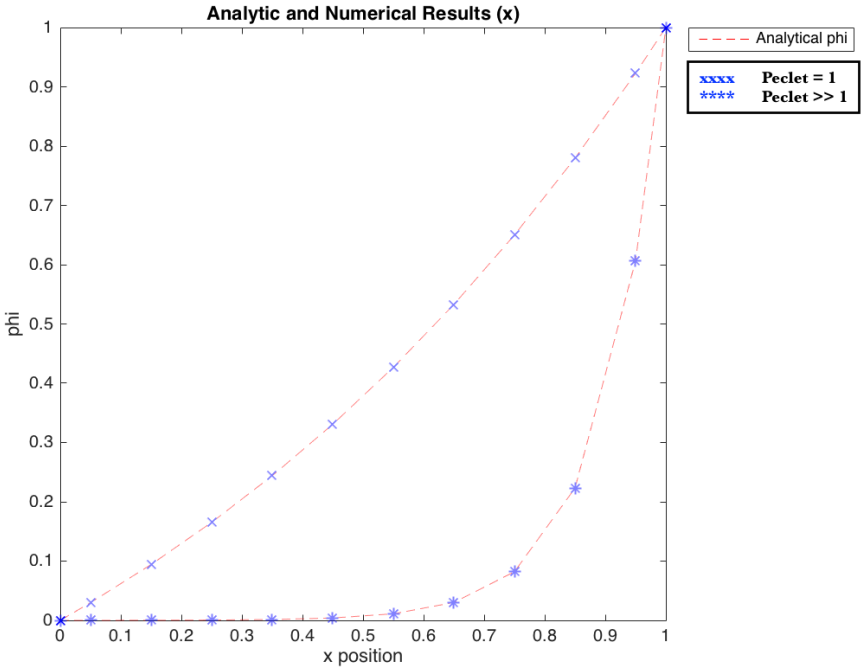


(a) Results with $P = 1$ and $P \gg 1$.

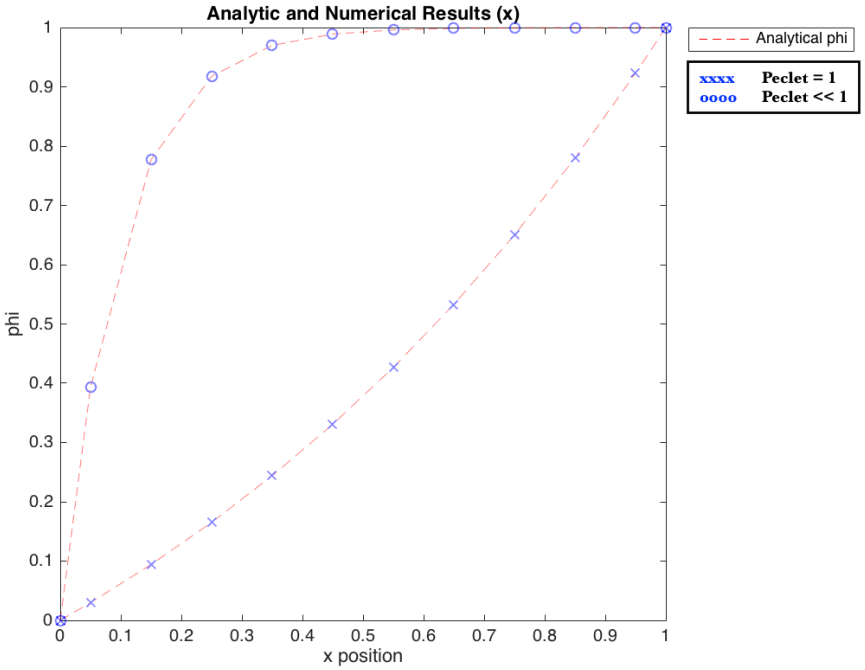


(b) Results with $P = 1$ and $P \ll 1$.

Figure 4.5: Comparison between results obtained with a PLDS scheme.

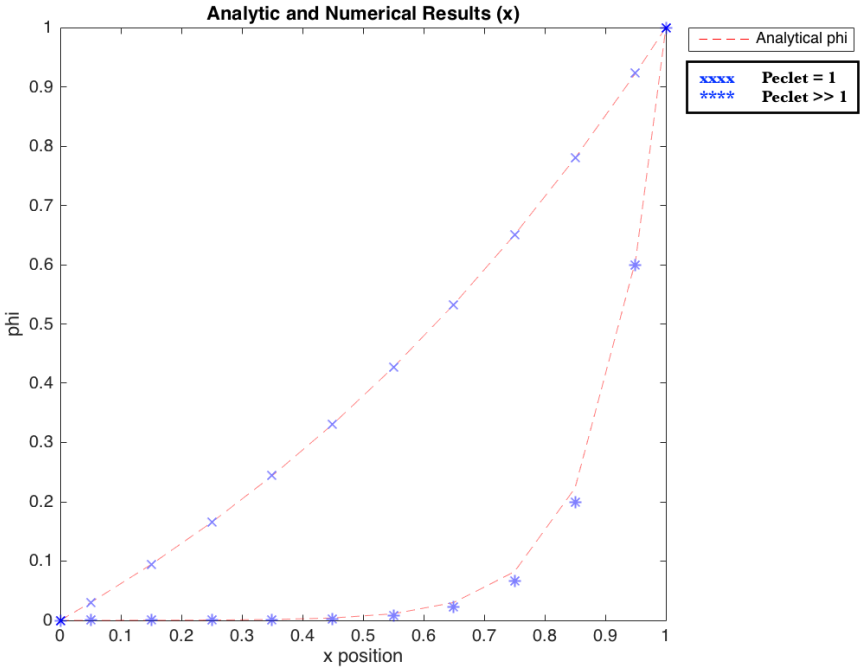


(a) Results with $P = 1$ and $P \gg 1$.

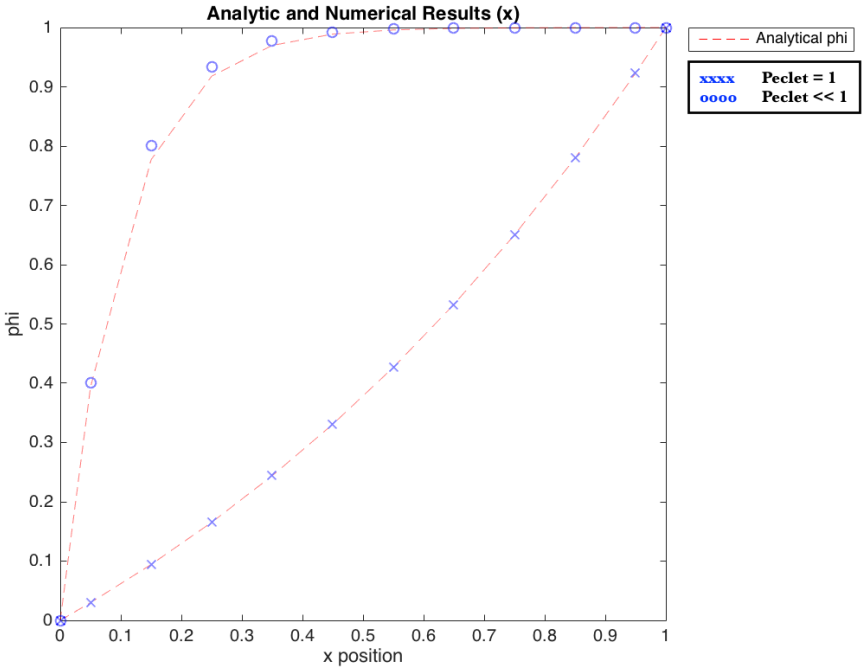


(b) Results with $P = 1$ and $P \ll 1$.

Figure 4.6: Comparison between results obtained with an EDS scheme.



(a) Results with $P = 1$ and $P \gg 1$.



(b) Results with $P = 1$ and $P \ll 1$.

Figure 4.7: Comparison between results obtained with an EDS scheme.

4.3.7 Conclusions

As we can see in the figures presented in the previous sections, the obtained results with the proposed numerical solution and the analytical one are very similar and the differences between them are negligible if we use an EDS or PLDS scheme. As we can see in Figure 4.7, with a CDS scheme, both results differ more.

As expected...

- The numerical solution obtained from the code is unidimensional.
- The difference between values when using an EDS or PLDS scheme is negligible.
- The values of ϕ on the boundaries are 0 and 1.
- The convective term, F, is bigger than the diffusive term, D. Therefore, the low boundary value of ϕ in Figure 4.4 (a) covers more space than the upper one [2].

[2] In real case, upper boundary value is right boundary value. We refer to it as upper because Figures 4.4 are rotated 90 degrees.

4.4 Unidimensional flow with unidimensional variation of the variable solved in the perpendicular direction of the flow

The aim of this case is to develop a code for the resolution of a flow with a variation of the variable in the perpendicular direction of the flow. See Reference [6] for the whole explanation of the problem.

4.4.1 Problem definition

It is a steady state convection-diffusion problem in which the variable solved has a unidimensional variation in the perpendicular direction of the flow. Its analytical solution is known when the velocity field is given.

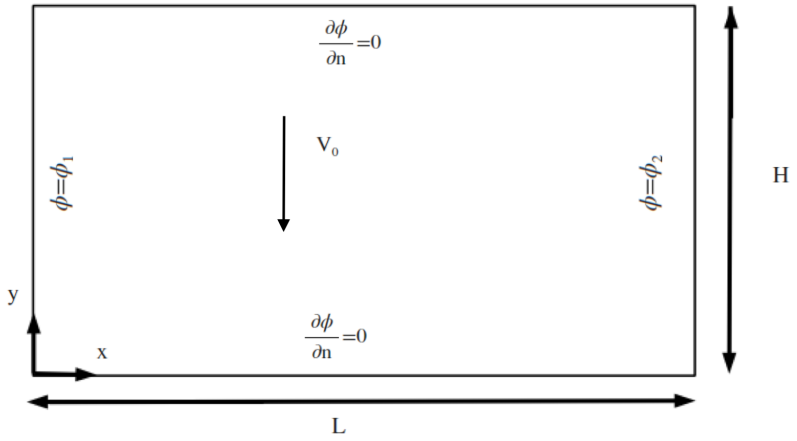


Figure 4.8: General schema of the proposed problem.

The velocity field is

$$u(x, y) = 0$$
$$v(x, y) = V_0$$

4.4.2 Resolution hypothesis

The resolution hypothesis are exactly the same as those exposed in section 4.3.2.

4.4.3 Spatial discretization

The domain discretization follows exactly the same procedure as the one exposed in section 4.3.3.

4.4.4 Calculation of coefficients

The values of the different coefficients can be obtained grouping the terms of equation 4.14. However, there is a certain number of nodes with defined boundary conditions, so that these coefficients acquire special values that must be calculated separately, as follows.

1. Nodes of the left side: We can apply the *Dirichlet* boundary condition. All nodes are at a constant $\phi = \phi_1$.

2. Nodes of the right side: We can apply the *Dirichlet* boundary condition. All nodes are at a constant $\phi = \phi_2$

3. Nodes of the top and bottom side: We can apply the *Neumann* boundary condition

$$\frac{\partial \phi}{\partial n} = 0$$

of zero normal convective flow and impose

$$\phi_{new}(i,j) = \phi_{new}(i,j-1)$$

at the top side and

$$\phi_{new}(i,j) = \phi_{new}(i,j+1)$$

at the bottom one.

4. Interior nodes: They are the general nodes of the study domain which can be obtained following the developed equations below 4.14.

4.4.5 Resolution algorithm

The scheme chosen to solve the problem is an implicit scheme because it returns a physically satisfactory behaviour and because of its simplicity.

The resolution algorithm to solve the equations and on which the code is based is the following:

1. Introduction of the input data
   - Physical data: Data such as problem coordinates, physical properties (ρ, Γ, S), boundary conditions, initial conditions, etc.
   - Numerical data: Mesh scheme, time step, convergence criteria, etc.
2. Previous calculations: The code includes many functions in order to do some previous calculations such as to generate the mesh, to calculate the convective and diffusion conductances, etc.
3. Setting of the property initial field. This step includes a function created to assign initial values to all the points of the domain: $\phi^*(i,j) = \phi^0(i,j)$
4. Evaluation of the discretization coefficients: All the coefficients are calculated (a_N, a_S, a_E, a_W, a_P and b_P). This function differentiates the calculation of the coefficients according to the numerical scheme desired (CDS, UDS, EDS or PLDS).
5. Calculation of the property map, $\phi(i,j)$. We solve the equation

$$a_P \phi_P = a_N \phi_N + a_S \phi_S + a_E \phi_E + a_W \phi_W + b_P$$

using the Gauss-Seidel solver.

6. Apply the convergence criteria: We ask the code if $max|\phi^*(i,j) - \phi(i,j)| < \delta$. If the variable meets this criteria, the code will stop calculating and will move on to the next step of the algorithm. If not, the code will refresh the guessed property map to the last calculated map and go to step 4.

7. Final calculations such as analytical ϕ.

8. Print results: We plot the evolution of both numerical and analytical ϕ through the domain in order to compare them.

9. End

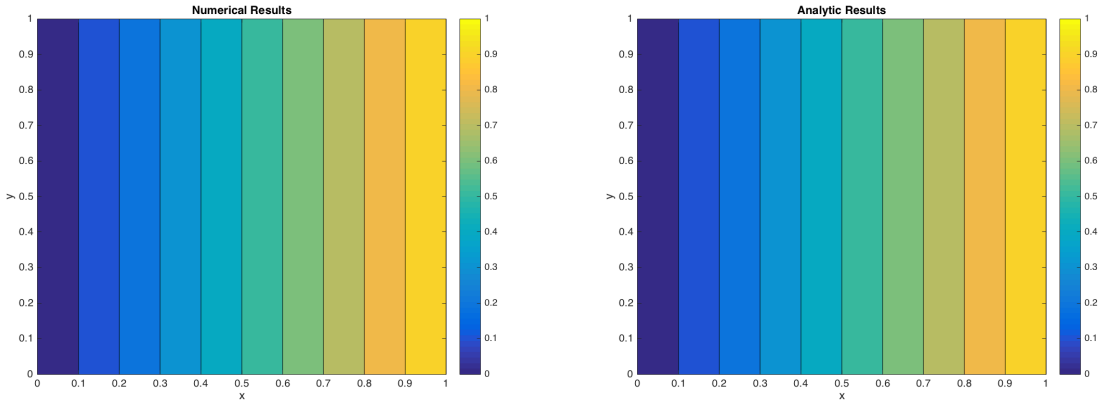
4.4.6 Results and Verification

The analytical solution follows the next expression:

$$\phi = \phi_1 + \frac{\phi_2 - \phi_1}{L} x \tag{4.16}$$

The code is validated with all ϕ values along the x and y coordinate points to ensure the correct calculation.

This problem is very similar to the first one, with the exception that, in this case, the solution follows a linear function (Equation 4.16) and must match perfectly with the results obtained by simulation, independently of the numerical scheme used for the convective terms, the mesh density and the Péclet number. In Figures 4.9 and 4.10 it can be seen how the numerical results fit exactly with the analytical solution.



(a) Numerical results of ϕ field.

(b) Analytical results of ϕ field.

Figure 4.9: Comparison between numerical and analytical results with $P = 1$.

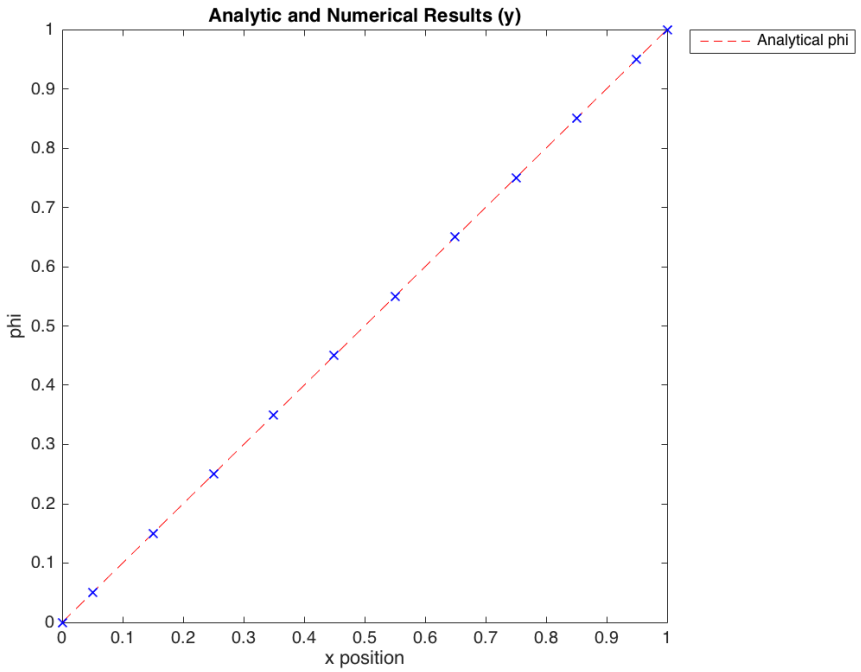


Figure 4.10: Comparison between the exact and calculated solutions of ϕ field with $P = 1$. Blue marker represents the numerical solution for ϕ.

We can also prove that the results are correct with Figures 4.11 and 4.12, where we see that the linear function does not change even if we modify the mesh size or Péclet number.

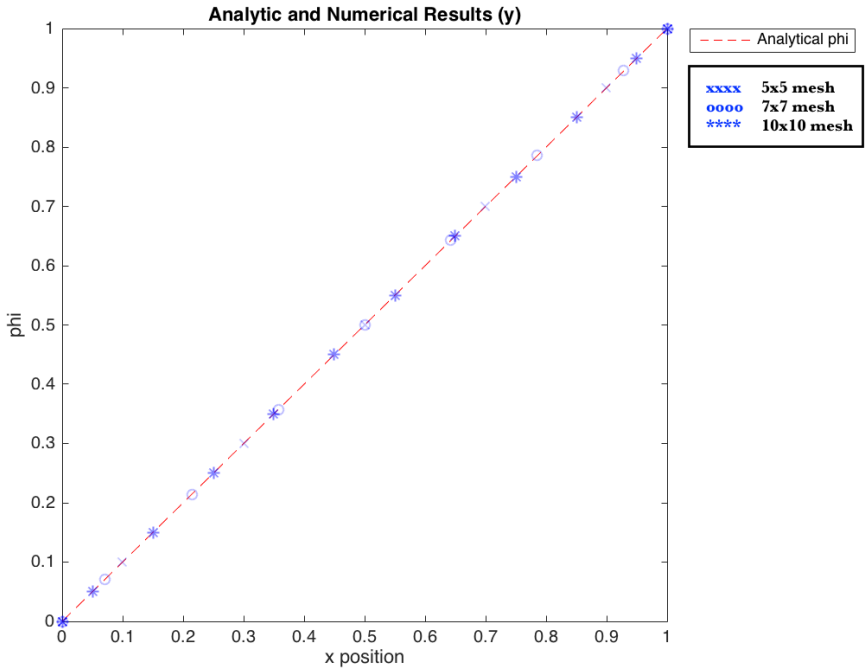


Figure 4.11: Comparison of results with different mesh sizes.

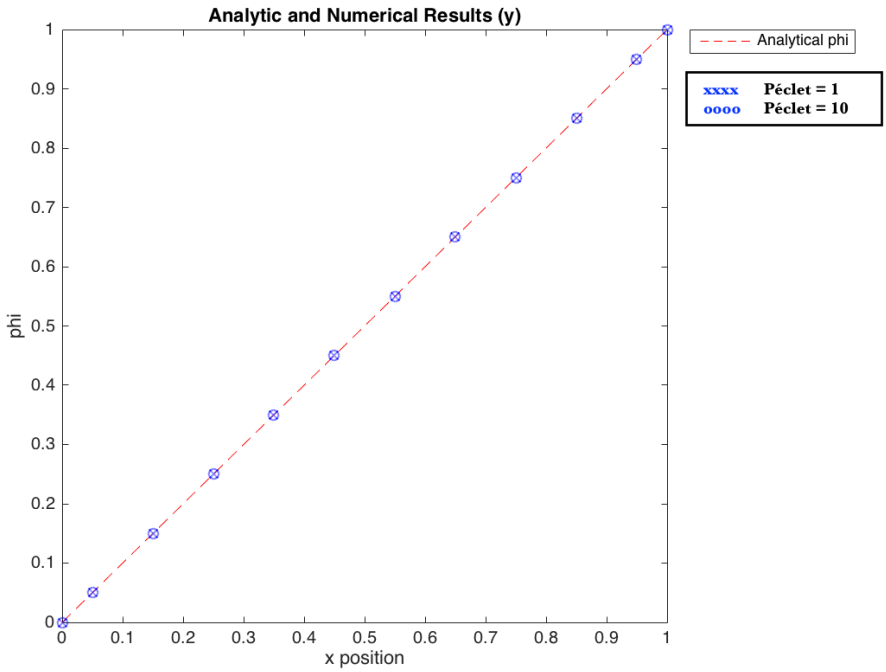


Figure 4.12: Comparison of results with different Péclet numbers.

Note that both solution's curves are practically the same, so it can be concluded that the program is correctly written.

4.4.7 Conclusions

As we can see in the figures presented in the previous sections, the results obtained with the proposed numerical solution and the analytical one are exactly the same.

As expected...

- The numerical solution obtained from the code is unidimensional.
- The difference between values when using an EDS or PLDS scheme is negligible.
- The values of ϕ on the boundaries are 0 and 1.
- The results do not depend on the Péclet number.
- The results do not depend on the mesh size.
- The results do not depend on the numerical scheme.
- The convective term, F, in the x direction is zero, so the diffusive term is the only non-zero term and, thus, the solution is equivalent to a simple conductive problem.

4.5 Diagonal Flow

The aim of this case is to develop a code for the resolution of a flow with a variation of the variable in an inclined direction ($\alpha = 45^o$). See Reference [6] for the whole explanation of the problem.

4.5.1 Problem definition

It is a steady convection-diffusion problem the solution of which is known for an infinite total Péclet (P) number when the flow is in the main diagonal and the boundary conditions of the dependent variables are the ones indicated in Figure 4.13.

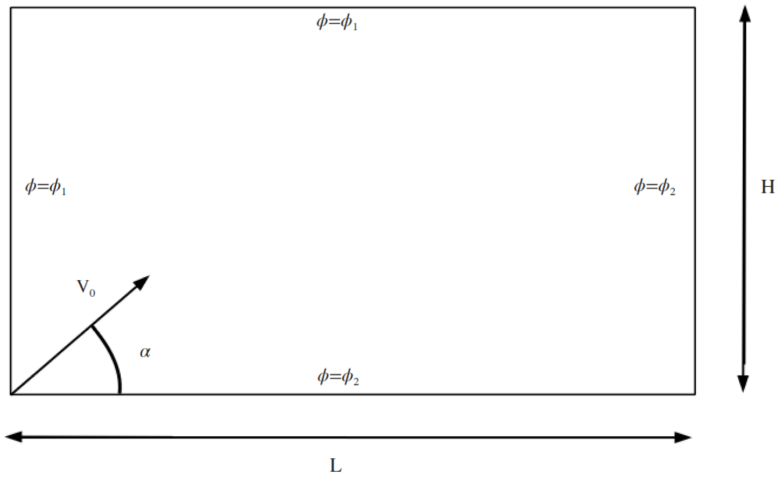


Figure 4.13: General schema of the proposed problem.

The velocity field is

$$u(x, y) = V_0 \cdot sin(\alpha)$$
$$v(x, y) = V_0 \cdot cos(\alpha)$$

The main input data and boundary conditions of the problem are:

Table 4.3: Input data

L	H	V_0	α	ϕ_1	ϕ_2
1	1	1	45	1	1

Table 4.4: Boundary conditions

Cavity wall	**Boundary condition**
Bottom	$\phi = \phi_2$
Top	$\phi = \phi_1$
Left	$\phi = \phi_1$
Right	$\phi = \phi_2$

4.5.2 Resolution hypothesis

The resolution hypothesis are exactly the same as those exposed in section 4.3.2.

4.5.3 Spatial discretization

The domain discretization follows exactly the same procedure as the one exposed in section 4.3.3.

4.5.4 Calculation of coefficients

The values of the different coefficients can be obtained grouping the terms of equation 4.14. However, there is a certain number of nodes with defined boundary conditions, so that these coefficients acquire special values that must be calculated separately, as follows.

1. Nodes of the left and top side: We can apply the *Dirichlet* boundary condition. All nodes are at a constant $\phi = \phi_1$.
2. Nodes of the right and bottom side: We can apply the *Dirichlet* boundary condition. All nodes are at a constant $\phi = \phi_2$
3. Interior nodes: They are the general nodes of the study domain which can be obtained following the developed equations below 4.14.

4.5.5 Resolution algorithm

The chosen scheme in order to solve the problem is an implicit scheme because it returns a physically satisfactory behaviour and because of its simplicity.

The resolution algorithm to solve the equations and on which the code is based is the following:

1. Introduction of the input data
   - Physical data: Data such as problem coordinates, physical properties (ρ, Γ, S), boundary conditions, initial conditions, etc.
   - Numerical data: Mesh scheme, time step, convergence criteria, etc.
2. Previous calculations: The code includes many functions in order to do some previous calculations such as to generate the mesh, to calculate the convective and diffusion conductances, etc.
3. Guess the property initial field. This step includes a function created to assign initial values to all the points of the domain: $\phi^*(i,j) = \phi^0(i,j)$
4. Evaluation of the discretization coefficients: a_N, a_S, a_E, a_W, a_P and b_P of all nodes according to the numerical scheme employed.
5. Calculation of the property map, $\phi(i,j)$. We solve the equation

$$a_P\phi_P = a_N\phi_N + a_S\phi_S + a_E\phi_E + a_W\phi_W + b_P$$

   using the Gauss-Seidel solver.

6. Apply the convergence criteria: We ask the code if $max|\phi^*(i,j) - \phi(i,j)| < \delta$. If the variable meets this criteria, the code will stop calculating and will go to the next step of the algorithm. If not, the code will refresh the guessed property map to the last calculated map and go to step 4.
7. Final calculations.
8. Print results: We plot the evolution of the numerical ϕ through the domain.
9. End

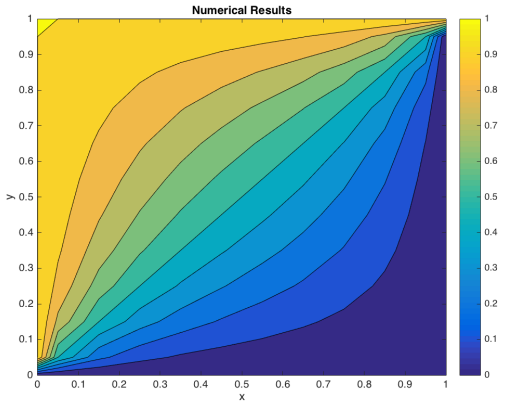
4.5.6 Results and Verification

The solution for $P_{total} = \infty$ is:

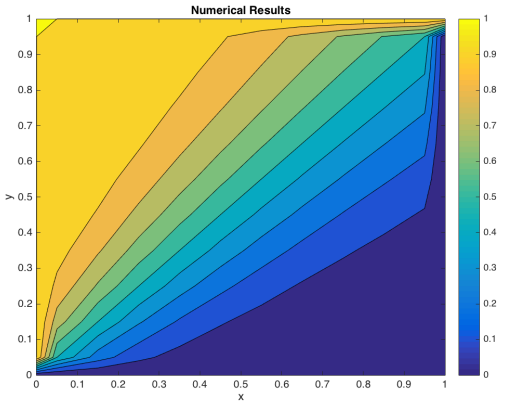
- $\phi = \phi_1$ above the diagonal
- $\phi = \phi_2$ below the diagonal

The numerical solution is very sensitive to the mesh size. For coarse meshes the truncation errors are important (false diffusion).

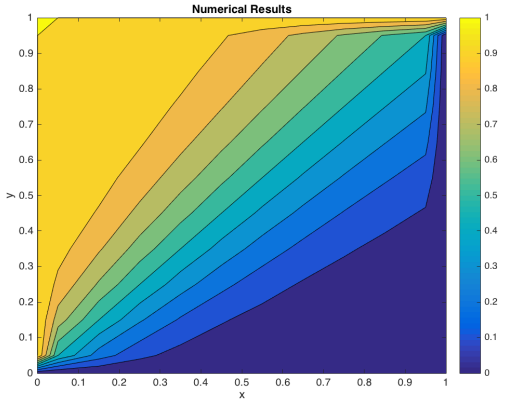
The different calculated solutions can be seen below:



(a) $P = 1$.

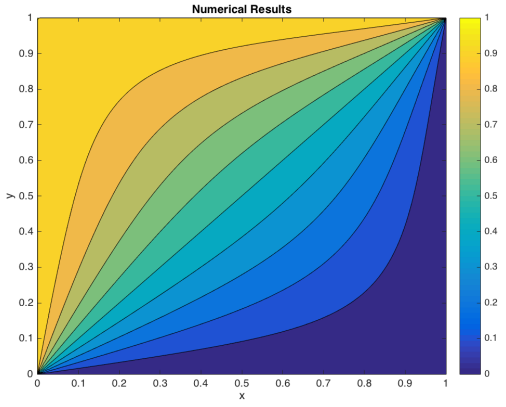


(b) $P = 100$.

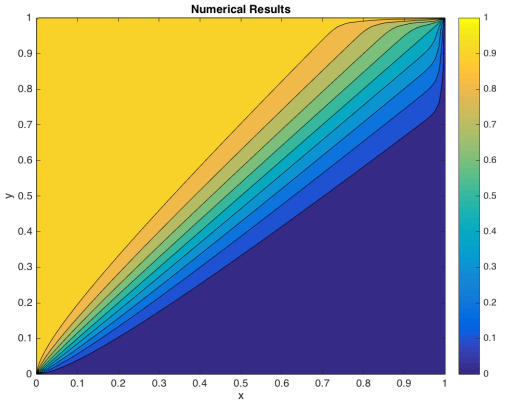


(c) $P = 1000$.

Figure 4.14: Numerical results of ϕ field with a 10x10 mesh.

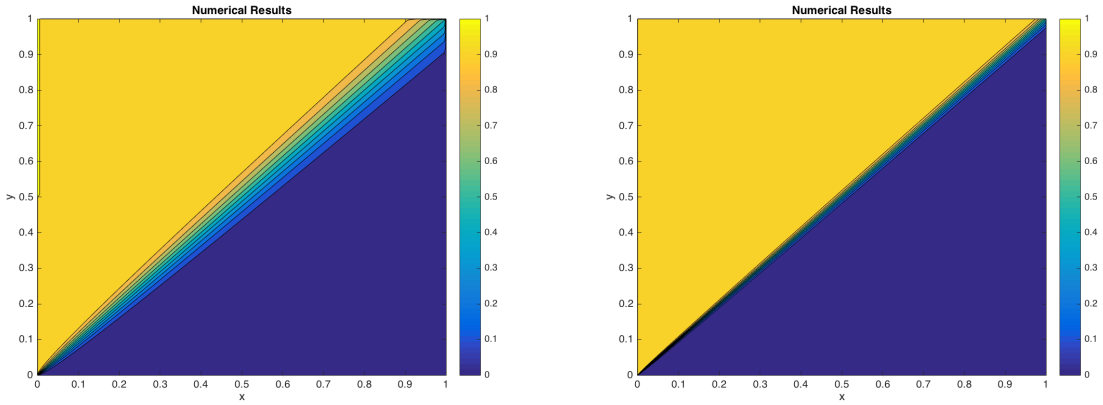


(a) $P = 1$.



(b) $P = 100$.

Figure 4.15: Numerical results of ϕ field with a 100x100 mesh.



(a) Numerical results with a 500x500 mesh and $P = 1000$.

(b) Numerical results with a 5000x5000 mesh and $P = 10^7$.

Figure 4.16: Numerical results of ϕ field.

4.5.7 Conclusions

The results presented in the previous section show a case of false diffusion. False diffusion happens when the flow is oblique to the grid lines and when there is a nonzero gradient of the dependent variable in the normal direction of the flow.

In this case, the convective term, F, is so large that its effect predominates on the diffusion effect, D. Therefore, the ideally ϕ field corresponds only to two streams with different values, equals to the values of the boundaries. The discrepancies with respect to the ideal case are on the boundary between the two streams, where there should not be a mixed layer.

We can prove that the numerical results obtained are correct if we take into account the following:

- The value of ϕ above the diagonal is equal to ϕ_1 when $P = \infty$.
- The value of ϕ below the diagonal is equal to ϕ_2 when $P = \infty$.

In addition, we can conclude that the numerical results are very sensitive to the mesh size. With a 10x10 or 100x100 mesh, the results differ a lot from the expected. When we analyze the flow with a bigger mesh (Figures 4.16) and with $P \gg 1$ we obtain the expected numerical results, independently of the numerical scheme used.

4.6 The Smith-Hutton Problem

The aim of this case is to develop a code for the resolution of the Smith - Hutton problem. See Reference [3] for the whole explanation of the problem.

4.6.1 Problem definition

We are now interested in the steady state solution of the Smith-Hutton problem, described in [13]. To do so, the 2D convection-diffusion equation 4.5

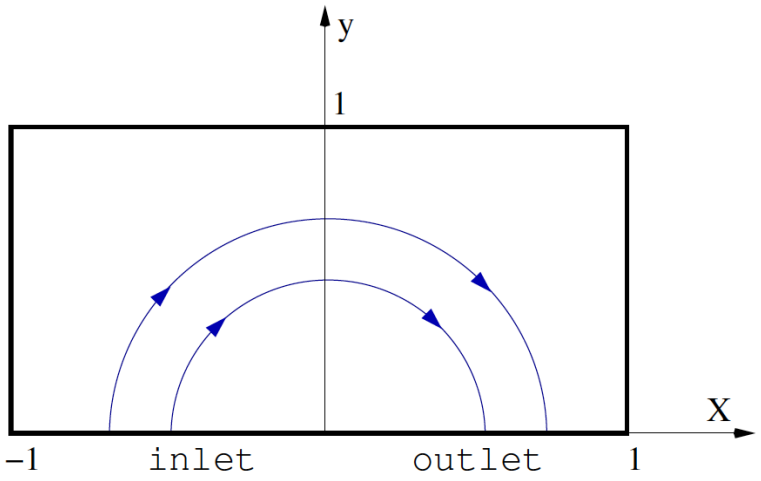


Figure 4.17: General schema of the proposed problem.

must be solved numerically in a rectangular domain (see Figure 4.17) with the prescribed velocity field given by

$$u(x,y) = 2y(1-x^2)$$
$$v(x,y) = -2x(1-y^2)$$

and the following boundary conditions for the variable ϕ:

$$\begin{cases} \phi = 1 + tanh\left[\alpha\left(2x+1\right)\right] & if \quad y = 0 \quad and \quad x \quad \epsilon \quad (-1,0) \\ \frac{\partial \phi}{\partial y} = 0 & if \quad y = 0 \quad and \quad x \quad \epsilon \quad (0,1) \\ \phi = 1 - tanh(\alpha) & elsewhere \end{cases}$$

where $\alpha = 10$.

4.6.2 Resolution hypothesis

The resolution hypothesis are exactly the same as those exposed in section 4.3.2.

4.6.3 Spatial discretization

The domain discretization follows the same procedure as the one exposed in section 4.3.3 with one difference: The first horizontal node, $i = 1$, is located at -1 m and the last one, $i = N + 2$ at 1 m. The nodes in the y direction are located in the same way as in the previous cases.

4.6.4 Calculation of coefficients

The values of the different coefficients can be obtained grouping the terms of equation 4.14. However, there is a certain number of nodes with defined boundary conditions, so that these coefficients acquire special values that must be calculated separately, as follows.

1. Nodes of the inlet side: We can apply the *Dirichlet* boundary condition. All nodes are at a constant $\phi = 1 + tanh\left[\alpha\left(2x + 1\right)\right]$.

2. Nodes of the outlet side: We can apply the *Neumann* boundary condition

$$\frac{\partial \phi}{\partial n} = 0$$

   of zero normal convective flow and impose

$$\phi_{new}(i, j) = \phi_{new}(i, j + 1)$$

3. Nodes of the left, top and right side: We can apply the *Dirichlet* boundary condition. All nodes are at a constant $\phi = 1 - tanh(\alpha)$.

4. Interior nodes: They are the general nodes of the study domain which can be obtained following the developed equations below 4.14.

4.6.5 Resolution algorithm

The chosen scheme to solve the problem is an implicit scheme because it returns a physically satisfactory behaviour and because of its simplicity.

The resolution algorithm to solve the equations and on which the code is based is the following:

1. Introduction of the input data
   - Physical data: Data such as problem coordinates, physical properties (ρ, Γ, S), boundary conditions, initial conditions, etc.
   - Numerical data: Mesh scheme, time step, convergence criteria, etc.
2. Previous calculations: The code includes many functions in order to do some previous calculations such as to generate the mesh, to calculate the convective and diffusion conductances, etc.
3. Guess the property initial field. This step includes a function created to assign initial values to all the points of the domain: $\phi^*(i, j) = \phi^0(i, j)$
4. Evaluation of the discretization coefficients: a_N, a_S, a_E, a_W, a_P and b_P of all nodes according to the numerical scheme employed.

5. Calculation of the property map, $\phi(i,j)$. We solve the equation

$$a_P\phi_P = a_N\phi_N + a_S\phi_S + a_E\phi_E + a_W\phi_W + b_P$$

using the Gauss-Seidel solver.

6. Apply the convergence criteria: We ask the code if $max|\phi^*(i,j) - \phi(i,j)| < \delta$. If the variable meets this criteria, the code will stop calculating and will go to the next step of the algorithm. If not, the code will refresh the guessed property map to the last calculated map and go to step 4.

7. Final calculations.

8. Print results: We plot the evolution of the numerical ϕ through the domain.

9. End

4.6.6 Results and Verification

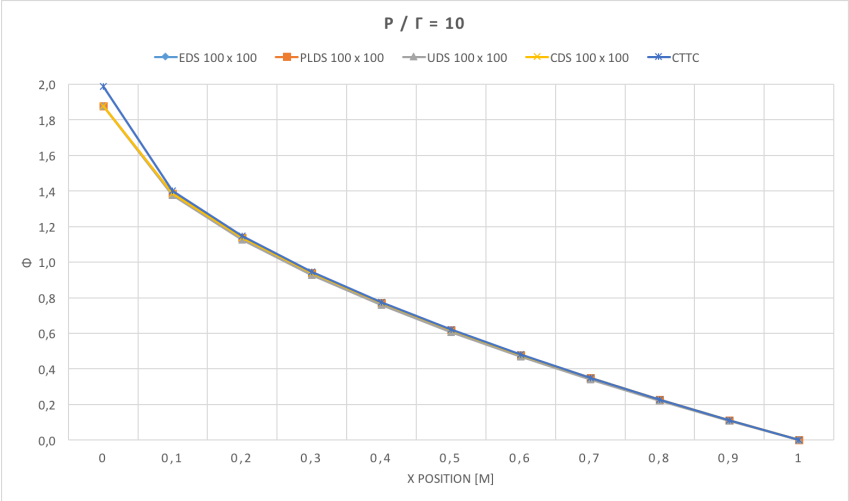
Table 4.5 shows the simulation parameters used for the simulation of the Smith-Hutton problem. We can verify the numerical resolution if we analyze the ϕ field for different values of the Péclet number:

$$\frac{\rho}{\Gamma} = \begin{pmatrix} 10 & ; & 10^3 & ; & 10^6 \end{pmatrix}$$

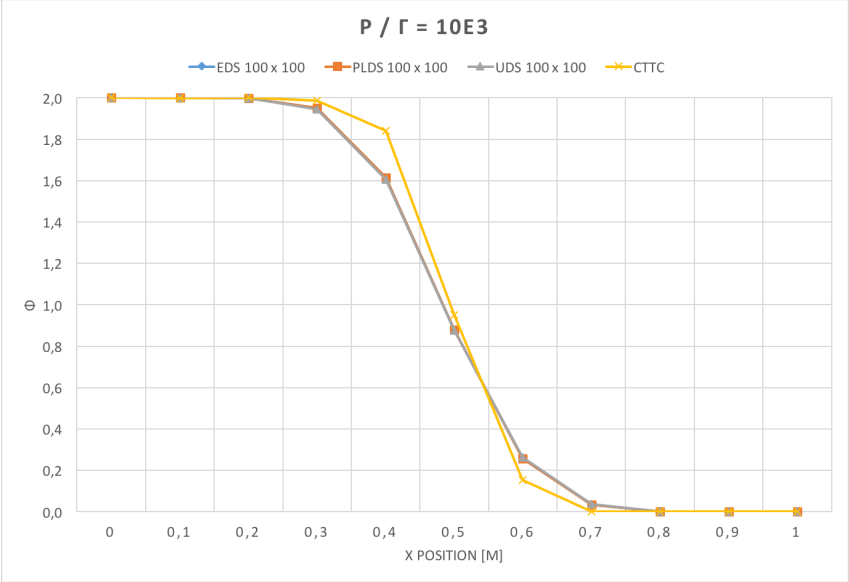
Table 4.5: Simulation parameters.

Convergence criteria	$1 \cdot 10^{-10}$
Mesh size	100 x 100
Relaxation factor	Variable
Solver	Gauss - Seidel

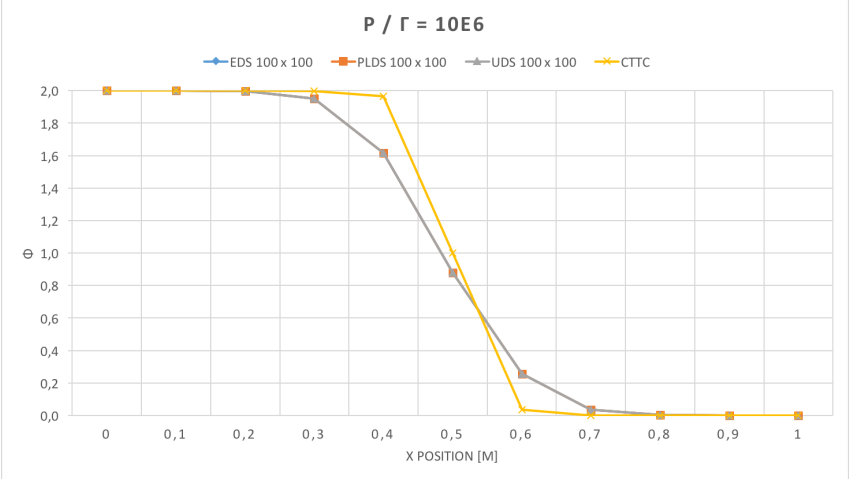
Next Figure 4.18 shows a comparison between the simulation results and the benchmark solution data provided by the *CTTC* in Reference [3].



(a) $\rho/\Gamma = 10$.



(b) $\rho/\Gamma = 10^3$.



(c) $\rho/\Gamma = 10^6$.

Figure 4.18: Comparison between obtained and expected results of ϕ field with a 100x100 mesh.

Table 4.6 represents the average relative error of the different simulations for each scheme and each Péclet number. In Tables 4.7, 4.8 and 4.9 we can see the same results presented in a tabular form.

Table 4.6: Average relative error between obtained and reference solutions.

ρ/Γ	**EDS (%)**	**PLDS (%)**	**UDS (%)**
10	0,92	0,92	2,29
10^3	0,67	1,56	2,14
10^6	1,66	1,66	1,65

Table 4.7: Results comparison with $\rho/\Gamma = 10$ and 100x100 mesh size.

		EDS		**PLDS**		**UDS**	
x [m]	**CTTC**	**Obtained**	**Error (%)**	**Obtained**	**Error (%)**	**Obtained**	**Error (%)**
0,0	1,989	1,879	5,52	1,879	5,52	1,877	5,62
0,1	1,402	1,386	1,12	1,386	1,12	1,377	1,76
0,2	1,146	1,137	0,77	1,137	0,77	1,126	1,76
0,3	0,946	0,941	0,57	0,941	0,57	0,928	1,86
0,4	0,775	0,771	0,53	0,771	0,53	0,759	2,10
0,5	0,621	0,618	0,44	0,618	0,44	0,607	2,26
0,6	0,480	0,478	0,38	0,478	0,39	0,468	2,44
0,7	0,349	0,348	0,25	0,348	0,26	0,340	2,50
0,8	0,227	0,226	0,24	0,226	0,25	0,221	2,64
0,9	0,111	0,111	0,31	0,111	0,29	0,109	2,21
1,0	0,000	0,000	0,00	0,000	0,00	0,000	0,00

Table 4.8: Results comparison with $\rho/\Gamma = 10^3$ and 100x100 mesh size.

		EDS		**PLDS**		**UDS**	
x [m]	**CTTC**	**Obtained**	**Error (%)**	**Obtained**	**Error (%)**	**Obtained**	**Error (%)**
0,0	2,0000	2,0000	0,00	2,0000	0,00	2,0000	0,00
0,1	1,9990	2,0000	0,05	2,0000	0,05	2,0000	0,05
0,2	1,9997	1,9999	0,01	1,9999	0,01	1,9998	0,01
0,3	1,9850	1,9900	0,25	1,9900	0,25	1,9879	0,14
0,4	1,8410	1,7790	3,37	1,7790	3,37	1,7656	4,10
0,5	0,9510	0,9193	3,34	0,9193	3,34	0,9184	3,43
0,6	0,1540	0,1695	0,20	0,1695	10,08	0,1783	15,78
0,7	0,0010	0,0097	0,10	0,0097	0,10	0,0112	0,00
0,8	0,0000	0,0002	0,00	0,0002	0,00	0,0003	0,00
0,9	0,0000	0,0000	0,00	0,0000	0,00	0,0000	0,00
1,0	0,0000	0,0000	0,00	0,0000	0,00	0,0000	0,00

Table 4.9: Results comparison with $\rho/\Gamma = 10^6$ and 100x100 mesh size.

		EDS		PLDS		UDS	
x [m]	**CTTC**	**Obtained**	**Error (%)**	**Obtained**	**Error (%)**	**Obtained**	**Error (%)**
0,0	2,000	2,000	0,00	2,000	0,00	2,000	0,00
0,1	2,000	2,000	0,00	2,000	0,00	2,000	0,00
0,2	2,000	2,000	0,01	2,000	0,01	2,000	0,01
0,3	1,999	1,990	0,45	1,990	0,45	1,990	0,45
0,4	1,964	1,780	9,39	1,780	9,39	1,780	9,39
0,5	1,000	0,919	8,08	0,919	8,08	0,919	8,08
0,6	0,036	0,169	0,20	0,169	0,20	0,169	0,20
0,7	0,001	0,010	0,10	0,010	0,10	0,010	0,00
0,8	0,000	0,000	0,00	0,000	0,00	0,000	0,00
0,9	0,000	0,000	0,00	0,000	0,00	0,000	0,00
1,0	0,000	0,000	0,00	0,000	0,00	0,000	0,00

In order to analyze the results and establish a reasonable comparison between the different numerical schemes it is necessary to find an optimal convergence criteria and mesh size. The purpose of the next subsections is to get an order of magnitude of the iteration's number needed by the different schemes to reach the convergence.

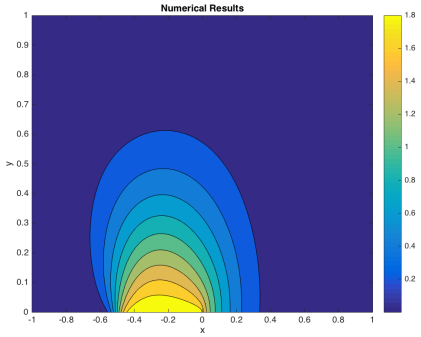
Convergence criteria

As the solution is given with four decimal positions, a convergence criteria of, at least, $\delta = 10^{-8}$ is expected to be truthful.

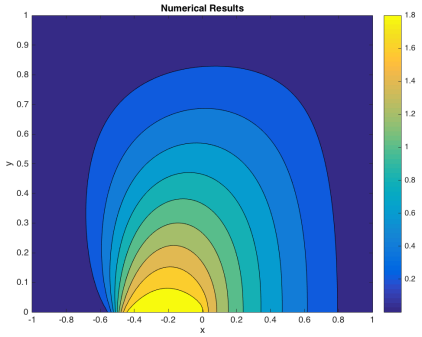
Table 4.10: Comparison between results calculated with different convergence criteria. *EA* stands for *Enough Accurate*.

Convergence criteria	ρ/Γ	**Results**	**Computational Time [s]**
10^{-4}	10	Not EA	3,86
	10^3	Not EA	1,13
	10^6	Not EA	0,78
10^{-5}	10	EA	7,49
	10^3	EA	1,24
	10^6	EA	1,24
10^{-6}	10	EA	11,37
	10^3	EA	1,4
	10^6	EA	1,3
10^{-8}	10	EA	25,01
	10^3	EA	1,7
	10^6	EA	1,6
10^{-9}	10	EA	31,49
	10^3	EA	1,65
	10^6	EA	1,32

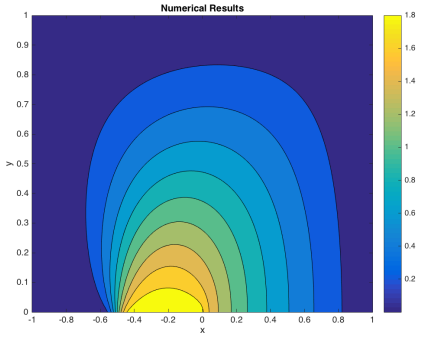
The following figures show the difference between the ϕ fields calculated with different convergence criteria.



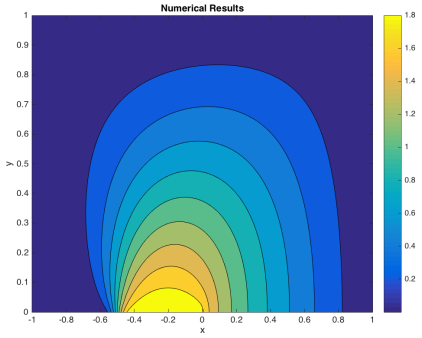
(a) $\delta = 10^{-3}$.



(b) $\delta = 10^{-4}$.



(c) $\delta = 10^{-5}$.



(d) $\delta = 10^{-6}$ // $\delta = 10^{-8}$ // $\delta = 10^{-9}$.

Figure 4.19: Comparison between ϕ fields with different δ and a 100x100 mesh.

As we can see in Figure 4.19, from a certain convergence criteria, the results of the ϕ field do not change. Seeing the computation times necessary for each case in Table 4.10, we can assure that a convergence criteria $\delta = 10^{-4}$ is good enough for the resolution of this problem, since good results are obtained. Even though, using a $\delta = 10^{-6}$ criteria, the computation time only increases in 7.51 seconds. Thus, we can conclude that the convergence criteria $\delta = 10^{-6}$, although it takes a little longer to calculate, is more precise.

Mesh size

As the grid is regular and the nodes position are multiples of 0.1, the number of nodes shall be multiple of 10 to have results at the exact positions. As we can see when computing with different mesh sizes, a mesh with a minimum of 100x100 nodes is suitable, as it gives enough accurate results.

4.6.7 Conclusions

The results presented in the previous sections lead us to the following conclusions:

- EDS is stable and the code converges correctly. As ρ/Γ increases, the computing time is reduced but the accuracy at the mid points turn worse.

- UDS is stable and the code converges correctly. As ρ/Γ increases, the computing time is reduced but the accuracy at the mid points become worse again. The source of error is attributed to false diffusion.
- CDS converges for $\rho/\Gamma = 10$ but can not for $\rho/\Gamma = 10^3$ or $\rho/\Gamma = 10^6$. This scheme leads to oscillations in the solution when the local Peclet number is larger than 2, so this can justify its instability [1].
- As the ρ/Γ ratio increases, the convective term takes a more predominant effect and, therefore, the diffusive effect decreases. This behaviour can be observed in all figures where $P \ll 1$.
- As Peclet number increases, the diffusive term decreases, and only a solenoidal field is seen because the convective term predominates versus the diffusive one.

Chapter 5

The Fractional Step Method

In the previous chapter, the procedure for discretizing and solving the general transport equation for the general variable (ϕ) in the presence of a known velocity field was formulated. In general, the velocity field is not known and has to be computed by solving the NS equations. For incompressible flows this task is complicated because of the coupling that exists between pressure and velocity and by the fact that pressure does not appear as a primary variable in either the momentum or continuity equations [8]. The main aim of this chapter is to present a method that addresses these two issues and computes the flow field for incompressible fluid flows.

In this chapter, we focus our attention on the resolution of the moment conservation equation as the basic way to obtain the velocity field. In this task, the flow field of the different problems will be solved out through the *Fractional Step Method* (FSM).

There are several objectives we need to fulfill in this new chapter [5]:

- Solve the NS equations by means of the FSM and understand its key features.
- Study the checkerboard problem.
- Implement a CFD code for structured and staggered and/or collocated meshes.
- Verify the developed code using different benchmark case data.

5.1 Introduction to Fractional Step Method

The FSM is a common technique for solving the incompressible NS equations that was first formulated independently by Chorin [2] and Temam [15]. As we know, solving general fluid flows requires an algorithm that can deal with the pressure - velocity coupling. This method provides an approach to approximate the convective and diffusive terms using a *predictor* velocity calculated with no pressure gradient. In this way, the constraint of incompressible flow can not be met and, to correct so, a Poisson equation is introduced into the system to calculate the *updated* velocity field.

Before going into detail in the explanation of the FSM, we present the general steps of it:

1. Calculation of an intermediate velocity field from the momentum equation without taking into account the pressure gradient.
2. Calculation of the pressure field from this intermediate velocity field making use of the continuity equation.

3. Calculation of the velocity field that satisfies the equation of continuity.

5.1.1 Theoretical background: The Helmholtz-Hodge Theorem

The Helmholtz-Hodge (HH) Theorem states that: *"A given vector field, ω, defined in a bounded domain, Ω, with smooth boundary, $\delta\Omega$, is uniquely decomposed in a pure gradient field and a divergence-free vector parallel to $\delta\Omega$"* [5]

$$\omega = \alpha + \nabla\varphi$$

where

$$\nabla \cdot \alpha = 0 \qquad \alpha\epsilon\Omega$$

5.1.2 Application of the HH theorem to the NS equations

As we already know, the NS equations for incompressible and with constant viscosity flows are:

$$\nabla \cdot \vec{v} = \quad 0 \tag{5.1}$$

$$\rho\frac{\partial v}{\partial t} + (\rho\vec{v} \cdot \nabla)\,\vec{v} = -\nabla p + \mu\Delta\vec{v} \tag{5.2}$$

or

$$\rho\frac{\partial v}{\partial t} = R(\vec{v}) - \nabla p \tag{5.3}$$

where

$$\vec{v} = u\vec{i} + v\vec{j} + w\vec{k}$$

and

$$R(\vec{v}) = -\,(\rho\vec{v} \cdot \nabla)\,\vec{v} + \quad \mu\Delta\vec{v}$$

where $R(\vec{v})$ stands for the convective and diffusive terms, which depends on velocity.

The final form of the FSM depends on the time integration method chosen. The integration of the momentum equations can be done at time instant t = n+1 or at time instant t = n + 1/2 (which involves the pressure gradient in n and n+1 instants). In this case, the integration of momentum equations is done at time instant (n + 1/2) while continuity equations are implicitly integrated. Time integration of the mentioned NS equations gives:

$$\nabla \cdot \vec{v}^{n+1} = \quad 0$$

$$\rho\frac{v^{n+1} - v^n}{\Delta t} = \frac{3}{2}R(v^{n-1}) - \nabla p^{n+1}$$

Defining an intermediate $\vec{v}$ following the Helmholtz-Hodge theorem (considering $\nabla\varphi = \nabla p$)

$$\vec{v}^P = \vec{v}^{n+1} + \frac{\Delta t}{\rho}\nabla p^{n+1}$$

we can transform the original momentum equation into the following velocity projection equation:

$$\rho\frac{v^P - v^n}{\Delta t} = \frac{3}{2}R(v^n) - \frac{1}{2}R(v^{n-1})$$

Moving on to the velocity field, we need the pressure field in order to obtain the real physical velocity. An equation for the pressure can be derived from the velocity decomposition equation if we apply the divergence operator:

$$\nabla \cdot \vec{v}^{n+1} = \nabla \vec{v}^{P} - \nabla \cdot \left(\frac{\Delta t}{\rho} \nabla p^{n+1} \right)$$

Since $\nabla \vec{v}^{n+1} = 0$ (continuity equation constrain), a final Poisson equation for the pressure is found:

$$\Delta p^{n+1} = \frac{\rho}{\Delta t} \nabla \cdot \vec{v}^{P} \tag{5.4}$$

We need to know the pressure on the neighboring nodes in order to obtain the pressure on our current node, so we need to do iterations (with a Gauss-Seidel method with over-relaxation, for example). When iterations are done, we obtain the pressure field from Equation 5.4.

We can summarize the entire procedure explained (FSM) in four different steps mentioned below:

1. Evaluation of $R(\vec{v}^{n})$
2. Evaluation of v^{P}
3. Evaluation of the pressure field through the Poisson equation
4. Obtention of the new velocity

5.1.3 The checkerboard problem

If we focus our attention in the one-dimensional spatial discretization of the third step mentioned above and we apply finite differences at node **P**:

$$\vec{v}^{n+1} = \vec{v}^{P} - \frac{\Delta t}{\rho} \nabla p^{n+1} \tag{5.5}$$

For the x-component of the velocity at node **P** (see Figure 5.1), we obtain:

$$u_{P}^{n+1} = u_{P}^{P} - \frac{\Delta t}{\rho} \cdot \frac{p_{E}^{n+1} - p_{W}^{n+1}}{2\Delta x} \tag{5.6}$$

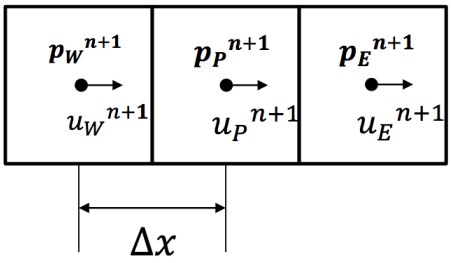


Figure 5.1: Discrete approximation of ∇p^{n+1} at node P. Figure taken from [5].

As we can see in Figure 5.1, the pressure gradient on P is $p_E - p_W$ and does not depend on p_P. This means that the discrete approximation of ∇p^{n+1} at node P is independent of p_P^{n+1} and, therefore, we can obtain converged velocity fields for nonphysical pressure distributions.

In other words, the pressure gradient term in the u-momentum equation involves pressures that are $2 \cdot \Delta x$ apart on the mesh, and does not involve the pressure at the point P. The same is true for the v-momentum equation. This means that if a checkerboarded pressure field is imposed on the mesh during iteration, the momentum equations are not able to distinguish it from a completely uniform pressure field. If the continuity equation was consistent with this pressure field as well, it would persist at convergence. This is known as the **checkerboard problem**. For example:

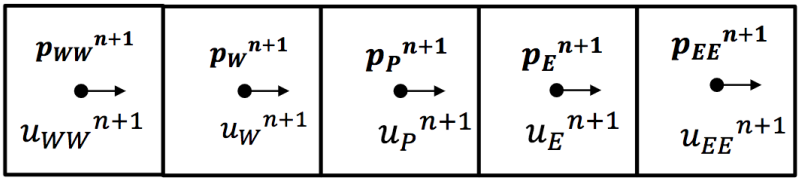


Figure 5.2: The checkerboard problem. Example taken from [5].

where:

$$p_{WW}^{n+1} = 100 \quad ; \quad p_W^{n+1} = 0 \quad ; \quad p_P^{n+1} = 100 \quad ; \quad p_E^{n+1} = 0 \quad ; \quad p_{EE}^{n+1} = 100$$

Since ∇p^{n+1} at node P is independent of p_P^{n+1}, the final velocity field will verify $\nabla p^{n+1} = 0$ and, therefore, we need a smarter strategy to couple ∇p^{n+1} with the velocity field v^{n+1}.

There are two possible solutions to solve the checkerboard problem:

- Staggered meshes: A typical staggered mesh arrangement is shown in Figure 5.3. We distinguish between the main cell or CV and the staggered cell or CV. The pressure is stored at centroids of the main cells and the velocity components are stored on the faces of the main cells, as shown, and are associated with the staggered cells. In this case, no further interpolation of velocity is necessary since discrete velocities are available directly where required. Thus, the possibility of velocity checkerboarding is eliminated.

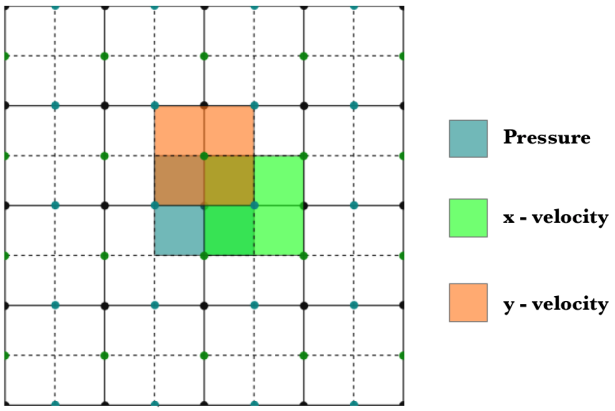


Figure 5.3: Staggered mesh.

- Collocated meshes: Although staggered grids have been very successful, in some cases it is desirable to use collocated meshes. In a collocated grid, all the flow field variables are stored at the same set of nodal points.

Collocated meshes offer significant advantages: *a)* all variables share the same location and hence, there is only one set of control volumes; *b)* the convection contribution to the coefficients in the discretized equations is the same for all variables; *c)* they offer simpler CFD code implementation than the staggered ones. Even though, it has an important disadvantage: the collocated mesh does not ensure the pressure-velocity coupling which may lead to the appearance of nonphysical checkerboard pressure field.

To make a first approach to the FSM we will focus on the application of it through staggered meshes. The basic algorithm for the FSM for staggered meshes at each time step is the following:

1. Evaluation of $R(u^n)$ and $R(v^n)$
2. Evaluation of v^P with
$$\vec{v}^P = \vec{v}^n + \frac{\Delta t}{\rho}\left[\frac{3}{2}R\left(\vec{v}^n\right) - \frac{1}{2}R\left(\vec{v}^{n-1}\right)\right]$$
3. Evaluation of the pressure field through the Poisson equation
$$\Delta p^{n+1} = \frac{\rho}{\Delta t}\nabla \cdot \vec{v}^P$$
4. Obtention of the new velocity through the evaluation of $\vec{v}^{n+1}$ with
$$\vec{v}^{n+1} = \vec{v}^P - \frac{\Delta t}{\rho}\nabla p^{n+1}$$
5. Choose a new $\Delta t = min(\Delta t_c, \Delta t_d)$

5.2 The lid-driven cavity flow problem

The study of a well-known case related to the dimensionless equations for incompressible flows of Newtonian fluids for different Reynolds (Re) values using the FSM is going to be undertaken. The unknown variables will be the velocity at interface nodes and the pressure at each node. The chosen case to be solved and to prove these equations is the lid-driven cavity flow problem.

5.2.1 Problem definition

The lid-driven cavity flow problem is commonly used to check new numerical schemes and their accuracy. This problem can be described as:

An incompressible and laminar flow is enclosed in a square cavity. The top side of the cavity moves with a constant horizontal velocity while all the other walls are considered to be static.

We can see the general schema of the proposed problem in next Figure.

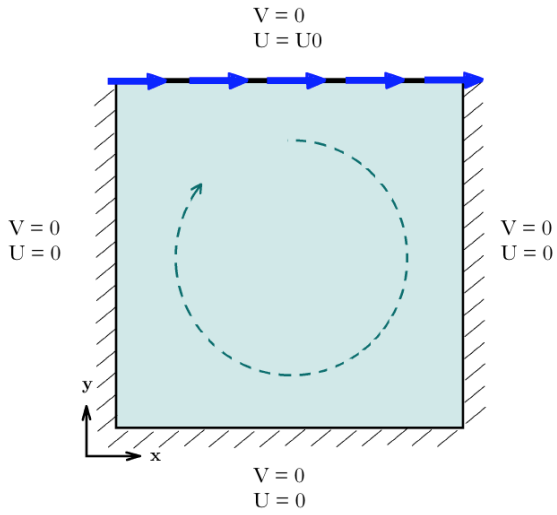


Figure 5.4: General schema of the proposed problem.

The main aim of this problem is to find the fluid velocities at different points within the cavity as a function of the Reynolds number.

5.2.2 Resolution hypothesis

To solve this exercise many hypothesis have been formulated:

1. Two - dimensional study of the case, so that the equations do not consider the contribution of depth.
2. Transient study of the equations.
3. Constant thermophysical properties.
4. Laminar flow.
5. Incompressible flow.
6. Newtonian flow.

5.2.3 Spatial discretization

The first step to solve the problem is to decide the appropriate mesh and configuration for the cavity. It is chosen to work on a staggered rectangular grid. In a staggered grid, the pressure is decoupled with the velocity of the CV and this prevents the apparition of unfeasible solutions arrangements such as the checkerboard pressure field.

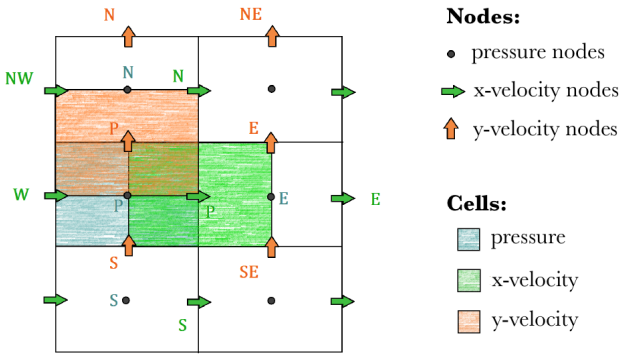


Figure 5.5: Spatial discretization by means of staggered meshes.

The domain is divided into N control volumes in the horizontal direction and M control volumes in the vertical direction. Each node had a length of $dx = \frac{B}{N}$ and $dx = \frac{H}{M}$.

As it can be appreciated in Figure 5.5, the main velocities (u and v) are placed in the faces of the pressure control volumes. In this way, the dimensions of the pressure mesh are (N+2,M+2) when considering all the control volumes, the dimensions of the x-staggered mesh are (N+1,M+2), and the dimensions of the y-staggered mesh are (N+2,M+1).

This problem will be solved with two different spatial discretizations. First spatial discretization will be a staggered uniform mesh and, when this last one gives enough accurate results, we will implement a non-uniform mesh in order to see if we obtain better results.

5.2.3.1 Non - Uniform Staggered Mesh

We take the pressure mesh as example to explain how non-uniform meshes work. The pressure mesh is a non-uniform structured mesh condensed in the walls of the domain. Consider a line segment joining two points, x_1 and x_2. A non-uniform mesh between both points can be definided following next equation:

$$\vec{x_i} = \vec{x_1} + s_i \cdot (\vec{x_2} - \vec{x_1}) \tag{5.7}$$

where s_i is:

$$s_i = 1 + \frac{tanh\left[k\left(\frac{i}{N} - 1\right)\right]}{tanh(k)} \tag{5.8}$$

where N is the number of nodes, and k is the stretching factor of the mesh. When $k \to 0$ we have an uniform distribution and, when k increases, the concentration towards $\vec{x_1}$ increases. We can confirm this behavior with Figures 5.6 and 5.7, see also Tables 5.1 and 5.2, where the position of the nodes of the mesh is represented as a function of the stretching factor, k.

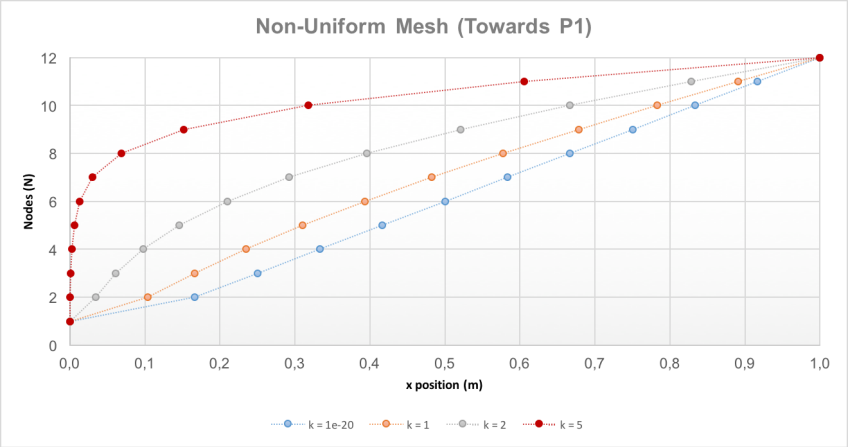


Figure 5.6: Non-Uniform mesh towards $\vec{x_1}$.

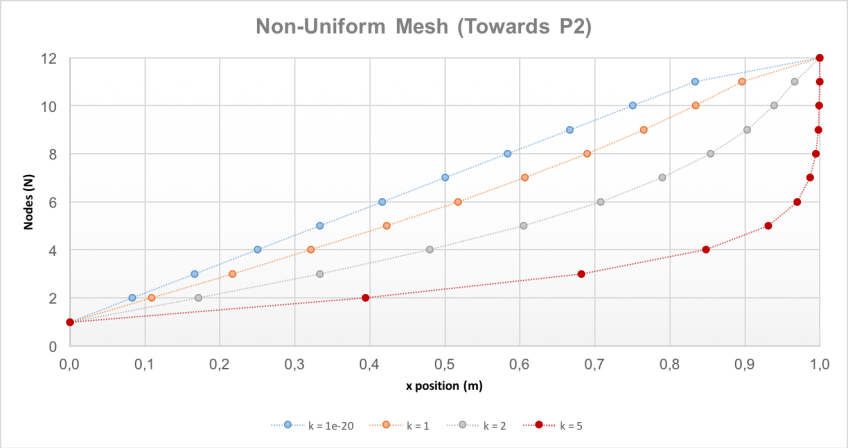


Figure 5.7: Non-Uniform mesh towards $\vec{x_2}$.

As we can see, as the k factor increases, the nodes get closer to $\vec{x_1}$ (if s_i is positive) or to $\vec{x_2}$ (if s_i is negative).

Table 5.1: Non-Uniform mesh towards $\vec{x_1}$.

	$\mathbf{k = 1 \cdot 10^{-20}}$	**k = 0,001**	**k = 1**	**k = 2**	**k = 3**	**k = 5**
N	**x (m)**	**x (m)**	**x (m)**	**x (m)**	**x (m)**	**x (m)**
1	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000
2	0,1667	0,1667	0,1042	0,0341	0,0085	0,0004
3	0,2500	0,2500	0,1660	0,0611	0,0171	0,0010
4	0,3333	0,3333	0,2348	0,0975	0,0312	0,0025
5	0,4167	0,4167	0,3105	0,1461	0,0539	0,0057
6	0,5000	0,5000	0,3932	0,2100	0,0904	0,0133
7	0,5833	0,5833	0,4825	0,2923	0,1475	0,0304
8	0,6667	0,6667	0,5778	0,3955	0,2346	0,0688
9	0,7500	0,7500	0,6784	0,5206	0,3617	0,1516
10	0,8333	0,8333	0,7832	0,6665	0,5356	0,3177
11	0,9167	0,9167	0,8908	0,8287	0,7539	0,6058
12	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000

Table 5.2: Non-Uniform mesh towards $\vec{x_2}$.

	$\mathbf{k = 1 \cdot 10^{-20}}$	**k = 0,001**	**k = 1**	**k = 2**	**k = 3**	**k = 5**
N	**x (m)**	**x (m)**	**x (m)**	**x (m)**	**x (m)**	**x (m)**
1	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000
2	0,0833	0,0833	0,1092	0,1713	0,2461	0,3942
3	0,1667	0,1667	0,2168	0,3335	0,4644	0,6823
4	0,2500	0,2500	0,3216	0,4794	0,6383	0,8484
5	0,3333	0,3333	0,4222	0,6045	0,7654	0,9312
6	0,4167	0,4167	0,5175	0,7077	0,8525	0,9696
7	0,5000	0,5000	0,6068	0,7900	0,9096	0,9867
8	0,5833	0,5833	0,6895	0,8539	0,9461	0,9943
9	0,6667	0,6667	0,7652	0,9025	0,9688	0,9975
10	0,7500	0,7500	0,8340	0,9389	0,9829	0,9990
11	0,8333	0,8333	0,8958	0,9659	0,9915	0,9996
12	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000

5.2.4 Boundary conditions

At this stage, it would be convenient to highlight some initial boundary conditions: impenetrability and non-slip conditions.

- Nodes at the bottom, left and right side: We can apply the *Dirichlet* boundary condition. This condition dictates that the velocity of the fluid will be zero relative to the boundary. Therefore, all the nodes are at a constant velocity.

$$\begin{cases} u = 0 \\ v = 0 \end{cases}$$

- Nodes at the top side: We can apply the *Dirichlet* boundary condition again.

$$\begin{cases} u = 1 \\ v = 0 \end{cases}$$

- Nodes at all sides: We can apply the *Neumann* boundary condition:

$$\frac{\partial p}{\partial n} = 0$$

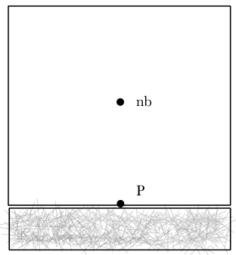


Figure 5.8: Neumann boundary condition.

Therefore:

$$\begin{cases} a_P = 1 \\ a_{nb} = 1 \\ a_{i \neq nb} = 0 \end{cases}$$

5.2.5 Evaluation of R(u) in the x-direction

We need to discretize Equation 5.9 in the x direction.

$$R(\vec{v}) = -\left(\rho \vec{v} \cdot \nabla\right) \vec{v} + \quad \mu \Delta \vec{v} \tag{5.9}$$

The function R(u) includes the convective and diffusive terms of the governing equation. In order to evaluate it, it is necessary to set up a new CV and perform integration. This CV is built around the main velocities of the problem, as seen in Subsection 5.2.3. When the function R(u) is integrated over this CV, the result is shown in next equations.

$$R(\vec{u}) = -\int_{\partial\Omega_x} (\rho u) u \cdot \vec{n} dS + \int_{\partial\Omega_x} \mu \nabla \vec{u} \cdot \vec{n} dS$$

$$\int_{V_P} R(\vec{u}) dV = \frac{1}{Re}\left[\frac{u_N - u_P}{d_{PN}} S_n - \frac{u_P - u_S}{d_{PS}} S_s + \frac{u_E - u_P}{d_{PE}} S_e - \frac{u_P - u_W}{d_{PW}} S_w\right] - \\ - \left[(v)_n v_n S_n - (v)_s v_s S_s + (u)_e u_e S_e - (u)_w u_w S_w\right]$$

As said before, velocities are evaluated at the CV faces. Doing so, the R(u) term is evaluated at the x-staggered mesh and the R(v) term at the y-staggered one. Once we have the discretization equations defined, we need to focus on how the mass flow and velocities at the faces are evaluated.

Evaluation of u_n, u_s, u_e and u_w in mass flow calculations:

In order to calculate the mass flow evaluated at each face of the CV it is convenient to pay attention to Figure 5.9. The mass flows, $\dot{m}_e$ and $\dot{m}_w$ can be calculated by means of the average velocity between:

$$(\rho u)_i = \rho \cdot 0.5 \cdot A_i \cdot (u_P + u_I)$$

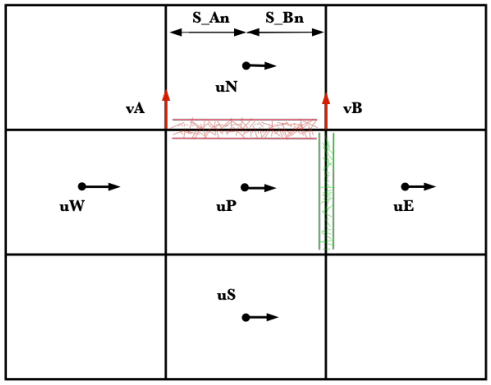


Figure 5.9: Staggered-x calculations.

The mass flows, $\dot{m}_n$ and $\dot{m}_s$ can be calculated as:

$$\dot{m}_n = (\rho v)_A \cdot A_{An} + (\rho v)_B \cdot A_{Bn}$$

Evaluation of u_e, u_w, v_n and v_s in x-staggered mesh:

We also need to evaluate the velocity at the faces of the x-staggered mesh. These velocities make reference to the velocity in the respective face of the CV (u_e, u_w, etc.). In order to calculate them, a special scheme has to be used. For simplicity and for convergence reasons, an UDS scheme is used. The UDS is a first order scheme and the value of ϕ at the cell face depends of the direction of the mass flow and has no convergence problems.

$$u_e = UDS(u_P, u_E)$$
$$u_e = CDS(u_P, u_E)$$
$$u_e = QUICK(u_P, u_E, u_W, u_{EE}, u_{WW})$$

5.2.6 Evaluation of the predictor velocity

All the procedure undertaken to calculate R(u) can be extrapolated to calculate R(v). Once we have evaluated $R(\vec{v}$, the next step in the algorithm is to find the predictor velocity: u^P and v^P. The expression to find it can be found below:

$$u^P = u^n + \frac{\Delta t}{2}\left[3R_x(u^n) - R(u^{n-1})\right] \tag{5.10}$$

$$v^P = v^n + \frac{\Delta t}{2}\left[3R_y(v^n) - R(v^{n-1})\right] \tag{5.11}$$

5.2.7 Discretization of Poisson equation

In order to obtain the discretization of Poisson equation, we need to integrate the next equation:

$$\Delta p^{n+1} = \frac{\rho}{\Delta t} \nabla \cdot \vec{v}^P \tag{5.12}$$

We obtain:

$$\int_\Omega \Delta p^{n+1} d\Omega = \frac{\rho}{\Delta t} \int \nabla u^P d\Omega \tag{5.13}$$

$$\int_\Omega \Delta p^{n+1} n dS = \frac{\rho}{\Delta t} \int u^P n dS \tag{5.14}$$

$$\frac{p_N - p_P}{d_{PN}} + \frac{p_P - p_S}{d_{PS}} + \frac{p_E - p_P}{d_{PE}} - \frac{p_P - p_W}{d_{PW}} =$$
$$\frac{1}{\Delta t} \left[(\rho v^P)_n S_n - (\rho v^P)_s S_s + (\rho u^P)_e S_e - (\rho u^P)_w S_w \right]$$

So we have a general equation with the next form:

$$a_P P_P = a_N p_N + a_S p_s + a_E p_E + a_W p_W + b_P \tag{5.15}$$

Where:

$$a_N = \frac{S_n}{dPN}$$
$$a_S = \frac{S_s}{dPs}$$
$$a_E = \frac{S_e}{dPE}$$
$$a_W = \frac{S_w}{dPW}$$
$$b_P = \frac{1}{\Delta t} \left[(\rho v^P)_n S_n - (\rho v^P)_s S_s + (\rho u^P)_e S_e - (\rho u^P)_w S_w \right]$$

5.2.8 Selection of time step

The explicit temporal scheme introduces severe restrictions on the time step due to stability reasons. The value of the time step is established by the Courant Friedrichs Lewy (CFL) condition.

$$\Delta t_C = min \left(0.35 \frac{\Delta x}{|v|} \right)$$

$$\Delta t_D = min \left(0.20 \frac{\rho \Delta x^2}{\mu} \right)$$

$$\Delta t = min(\Delta t_C, \Delta t_D)$$

The CFL condition is a necessary condition for convergence while solving certain partial differential equations numerically by means of finite differences method.

5.2.9 Resolution algorithm

The resolution algorithm to solve the equations and on which the code is based is the following:

1. Introduction of the input data
   - Physical data: Data such as problem coordinates, physical properties (ρ, Γ, S, Reynolds number), boundary conditions, initial conditions, etc.
   - Numerical data: Mesh scheme, time step, convergence criteria, etc.
2. Previous calculations: The code includes many functions in order to do some previous calculations such as to generate the mesh, to calculate the spatial dimensions, etc.
3. Setting of the initial fields for all properties. This step includes a function created to assign initial values to all the points of the domain: $u^*(i,j) = u^0(i,j)$, $v^*(i,j) = v^0(i,j)$, $p^*(i,j) = p^0(i,j)$, etc.
4. Evaluation of boundary velocity: u and v.
5. Evaluation of convective and diffusive terms: $R(u)_x$ and $R(v)_y$.
6. Evaluation of predictor velocity: $\vec{v}^P$.
7. Evaluation of constant discretization coefficients of Possion equation: a_N, a_S, a_E, a_W and a_P of all nodes.
8. Evaluation of non-constant discretization coefficients of Possion equation: b_P of all nodes.
9. Calculation of the property map, $p(i,j)$. We solve the equation

   $$a_P p_P = a_N p_N + a_S p_S + a_E p_E + a_W p_W + b_P$$

   using the Gauss-Seidel solver.
10. Evaluation of the velocity field, u and v.
11. Setting the calculated variables to the next guessed variable: $R(u)_{new} = R(u)$, $R(v)_{new} = R(v)$, $u_{new} = u$, $v_{new} = v$ and $p_{new} = p$.
12. Evaluation of new time step with equations written in Section 5.2.8. Start of time loop.
13. Evaluation of boundary velocity: u and v.
14. Evaluation of convective and diffusive terms: $R(u)_x$ and $R(v)_y$.
15. Evaluation of predictor velocity: $\vec{v}^P$.
16. Evaluation of non-constant discretization coefficients of Possion equation: b_P of all nodes.
17. Calculation of the property map, $p(i,j)$. We solve the equation

    $$a_P p_P = a_N p_N + a_S p_S + a_E p_E + a_W p_W + b_P$$

    using the Gauss-Seidel solver.
18. Evaluation of the velocity field, u and v.

19. Applying the first convergence criteria: We ask the code if $max|u^*(i,j)-u(i,j)|/timestep < \delta$. If the variable meets this criteria, the code will stop calculating and will go to the next step of the algorithm. If not, the code will refresh the guessed property map to the last calculated map and go to step 12.

20. Final calculations.

21. Print results

22. End

5.2.10 Results and Verification

The numerical parameters of the simulation are shown below:

Table 5.3: Simulation parameters.

Type of discretization	Staggered
Mesh size	100x100
Precision (values)	10^{-3}
Precision (steady state)	10^{-3} or 10^{-6}
Time step	CFL
Solver	Gauss - Seidel
Reynolds	100, 400, 1000, 3200, 5000, 7500, 10000

After the code is created, it is validated with results presented in Ghia, see Reference [16]. These results are related to velocity field. The authors of the article present different solution data:

1. The horizontal components of velocity in the vertical line located in the middle of the cavity.

2. The vertical component of speed in the horizontal line located halfway up the cavity.

The cases analyzed by the authors in the benchmark solution include different Reynolds numbers: Re = 100, Re = 400, Re = 1000, Re = 3200, Re = 5000, Re = 7500 and Re = 10000, whereas in the present analysis the upper limit of the Reynolds number will be 5000. They are omitted because it does not make sense to include them in the comparison of results, but why?

Note that speed values provided in these tables are single numbers which correspond to the value that the variable takes when the case is stabilized, that is, when the system enters a steady regime and the speeds acquire stable values. It turns out that, from cases over Re = 5000, it is not true that the steady state is reached. What happens is that, around this Reynolds value, we begin to have convergence difficulties and this implies that we are in unstable flow zones. The flow becomes turbulent and therefore, there is no temporary stabilization and it does not make sense to provide a single speed value.

Table 5.4: Comparative between values for u velocity component.

	Re = 100		**Re = 400**		**Re = 1000**		**Re = 3200**		**Re = 5000**	
y	**Ghia (u)**	**Code (u)**	**Ghia (u)**	**Code (u)**	**Ghia (u)**	**Code (u)**	**Ghia (u)**	**Code (u)**	**Ghia (u)**	**Code (u)**
1,0000	1,00000	1,00000	1,00000	1,00000	1,00000	1,00000	1,00000	1,00000	1,00000	1,00000
0,9766	0,84123	0,83978	0,75837	0,74402	0,65928	0,62753	0,53236	0,51090	0,48223	0,43469
0,9688	0,78871	0,78692	0,68439	0,66643	0,57492	0,53418	0,48296	0,45824	0,46120	0,41671
0,9609	0,73722	0,73455	0,61756	0,59618	0,51117	0,46233	0,46547	0,43439	0,45992	0,36825
0,9531	0,68717	0,68463	0,55892	0,53651	0,46604	0,43116	0,46101	0,45202	0,46036	0,40003
0,8516	0,23151	0,22818	0,29093	0,27456	0,33304	0,27082	0,34682	0,21868	0,33556	0,30196
0,7344	0,00332	0,00358	0,16256	0,14528	0,18719	0,14925	0,19791	0,12516	0,20087	0,18390
0,6172	-0,13641	-0,13924	0,02135	0,01311	0,05702	0,04744	0,07156	0,06708	0,08183	0,07498
0,5000	-0,20581	-0,20133	-0,11477	-0,11661	-0,06080	-0,04945	-0,04272	-0,03883	-0,03039	-0,02796
0,4531	-0,21090	-0,20468	-0,17119	-0,16880	-0,10648	-0,08546	-0,86636	-0,07743	-0,07404	-0,05210
0,2813	-0,15662	-0,15236	-0,32726	-0,27276	-0,27805	-0,24062	-0,24427	-0,22964	-0,22855	-0,20345
0,1719	-0,10150	-0,10016	-0,24299	-0,20853	-0,38289	-0,29337	-0,34323	-0,31714	-0,33050	-0,30379
0,1016	-0,06434	-0,06414	-0,14612	-0,13740	-0,29730	-0,24256	-0,41933	-0,39348	-0,40435	-0,36020
0,0703	-0,04775	-0,0466	-0,10338	-0,10140	-0,22220	-0,19785	-0,37827	-0,35844	-0,43643	-0,28468
0,0625	-0,04192	-0,04199	-0,09266	-0,09176	-0,20196	-0,18367	-0,35344	-0,29989	-0,42901	-0,28339
0,0547	-0,03717	-0,03727	-0,08186	-0,08180	-0,18109	-0,16802	-0,32407	-0,28867	-0,41165	-0,28185
0,0000	0,00000	0,00000	0,00000	0,00000	0,00000	0,00000	0,00000	0,00000	0,00000	0,00000

Table 5.5: Comparative between values for v velocity component.

	Re = 100		**Re = 400**		**Re = 1000**		**Re = 3200**		**Re = 5000**	
x	**Ghia (v)**	**Code (v)**	**Ghia (v)**	**Code (v)**	**Ghia (v)**	**Code (v)**	**Ghia (v)**	**Code (v)**	**Ghia (v)**	**Code (v)**
1,0000	0,00000	0,00000	0,00000	0,00000	0,00000	0,00000	0,00000	0,00000	0,00000	0,00000
0,9688	-0,05906	-0,06290	-0,12146	-0,12268	-0,21388	-0,24572	-0,39017	-0,33466	-0,49774	-0,47817
0,9609	-0,07391	-0,07872	-0,15663	-0,15692	-0,27669	-0,29785	-0,47425	-0,43476	-0,55069	-0,48406
0,9531	-0,08864	-0,09494	-0,19254	-0,19044	-0,33714	-0,34432	-0,52357	-0,50174	-0,55408	-0,49990
0,9453	-0,10313	-0,10904	-0,22847	-0,22300	-0,39188	-0,38367	-0,54053	-0,50334	-0,52876	-0,45644
0,9063	-0,16914	-0,17569	-0,23827	-0,35178	-0,51550	-0,44058	-0,44307	-0,39332	-0,41442	-0,37937
0,8594	-0,22445	-0,22806	-0,44993	-0,40086	-0,42665	-0,34959	-0,37401	-0,33036	-0,36214	-0,33691
0,8047	-0,24533	-0,24288	-0,38598	-0,34884	-0,31966	-0,25960	-0,31184	-0,22194	-0,30018	-0,29011
0,5000	0,05454	0,05319	0,05186	0,06550	0,02526	0,02936	0,00999	0,0088	0,00945	0,01218
0,2344	0,17527	0,17434	0,30174	0,25568	0,32235	0,27389	0,28188	0,26507	0,27280	0,25034
0,2266	0,17507	0,17432	0,30203	0,25579	0,33075	0,27983	0,29030	0,27136	0,28066	0,23798
0,1563	0,16077	0,16146	0,28124	0,24151	0,37095	0,30228	0,37119	0,32464	0,35368	0,22555
0,0938	0,12317	0,12446	0,22965	0,20075	0,32627	0,26213	0,42768	0,37262	0,42951	0,25629
0,0781	0,10890	0,11013	0,20920	0,18311	0,30353	0,24344	0,41906	0,37390	0,43648	0,35012
0,0703	0,10091	0,10219	0,19713	0,17274	0,29012	0,23253	0,40917	0,23817	0,43329	0,34580
0,0625	0,09233	0,09348	0,18360	0,16070	0,27485	0,22037	0,39560	0,23084	0,42447	0,23625
0,0000	0,00000	0,00000	0,00000	0,00000	0,00000	0,00000	0,00000	0,00000	0,00000	0,00000

We have also calculated the relative error between values obtained by the own code and values given by Ghia in Tables 5.6 and 5.7.

Table 5.6: Relative error between values for u velocity component.

	Re = 100	**Re = 400**	**Re = 1000**	**Re = 3200**	**Re = 5000**
y	**Error (%)**	**Error (%)**	**Error (%)**	**Error (%)**	**Error (%)**
1,0000	0,00	0,00	0,00	0,00	0,00
0,9766	0,17	1,89	4,82	4,03	9,86
0,9688	0,23	2,62	7,09	5,12	9,65
0,9609	0,36	3,46	9,55	6,68	19,93
0,9531	0,37	4,01	11,78	1,95	13,11
0,8516	1,44	15,94	24,69	36,95	10,01
0,7344	7,83	16,78	20,27	36,76	8,45
0,6172	2,07	38,59	22,06	6,26	8,37
0,5000	2,18	1,60	18,67	9,11	7,98
0,4531	2,95	1,40	19,74	91,06	29,63
0,2813	2,72	16,65	13,46	5,99	10,98
0,1719	1,32	14,18	23,38	7,60	8,08
0,1016	0,31	5,97	18,41	6,16	10,92
0,0703	2,41	1,92	10,96	5,24	34,77
0,0625	0,17	0,97	9,06	15,15	33,94
0,0547	0,27	0,07	7,22	10,92	31,53
0,0000	0,00	0,00	0,00	0,00	0,00
Average Error	1,46	7,42	13,01	14,65	14,54

Table 5.7: Relative error between values for v velocity component.

	Re = 100	**Re = 400**	**Re = 1000**	**Re = 3200**	**Re = 5000**
x	**Error (%)**	**Error (%)**	**Error (%)**	**Error (%)**	**Error (%)**
1,0000	0,00	0,00	0,00	0,00	0,00
0,9688	6,50	1,00	14,89	14,23	3,93
0,9609	6,51	0,19	7,65	8,33	12,10
0,9531	7,11	1,09	2,13	4,17	9,78
0,9453	5,73	2,39	2,09	6,88	13,68
0,9063	3,87	47,64	14,53	11,23	8,46
0,8594	1,61	10,91	18,06	11,67	6,97
0,8047	1,00	9,62	18,79	28,83	3,35
0,5000	2,48	26,30	16,25	11,71	28,84
0,2344	0,53	15,26	15,03	5,96	8,23
0,2266	0,43	15,31	15,39	6,52	15,21
0,1563	0,43	14,13	18,51	12,54	36,23
0,0938	1,05	12,58	19,66	12,87	40,33
0,0781	1,13	12,47	19,80	10,78	19,79
0,0703	1,27	12,37	19,85	41,79	20,19
0,0625	1,25	12,47	19,82	41,65	44,34
0,0000	0,00	0,00	0,00	0,00	0,00
Average Error	2,40	11,40	13,09	13,48	15,97

Comparison of velocity field through tables

The direct comparison by values is summarized in tables 5.4 and 5.5, where the benchmark velocity profiles are compared to those calculated with the created code. In order to do a more comfortable comparison, see Table 5.8:

Table 5.8: Values comparison for Re = 100.

	U - Velocity (Re = 100)				**V - Velocity (Re = 100)**		
y	**Ghia**	**Code**	**Error (%)**	**x**	**Ghia**	**Code**	**Error (%)**
1,0000	1,00000	1,00000	0,00	1,0000	0,00000	0,00000	0,00
0,9766	0,84123	0,83978	14,23	0,9688	-0,05906	-0,06290	-0,54
0,9688	0,78871	0,78692	8,33	0,9609	-0,07391	-0,07872	-0,55
0,9609	0,73722	0,73455	4,17	0,9531	-0,08864	-0,09494	-0,52
0,9531	0,68717	0,68463	6,88	0,9453	-0,10313	-0,10904	-0,49
0,8516	0,23151	0,22818	11,23	0,9063	-0,16914	-0,17569	-0,41
0,7344	0,00332	0,00358	11,67	0,8594	-0,22445	-0,22806	-0,36
0,6172	-0,13641	-0,13924	28,83	0,8047	-0,24533	-0,24288	-0,30
0,5000	-0,20581	-0,20133	11,71	0,5000	0,05454	0,05319	0,01
0,4531	-0,21090	-0,20468	5,96	0,2344	0,17527	0,17434	0,27
0,2813	-0,15662	-0,15236	6,52	0,2266	0,17507	0,17432	0,28
0,1719	-0,10150	-0,10016	12,54	0,1563	0,16077	0,16146	0,35
0,1016	-0,06434	-0,06414	12,87	0,0938	0,12317	0,12446	0,42
0,0703	-0,04775	-0,0466	10,78	0,0781	0,10890	0,11013	0,44
0,0625	-0,04192	-0,04199	41,79	0,0703	0,10091	0,10219	0,44
0,0547	-0,03717	-0,03727	41,65	0,0625	0,09233	0,09348	0,44
0,0000	0,00000	0,00000	0,00	0,0000	0,00000	0,00000	0,00

As seen in this table, the velocity values between the benchmark case and the code ones are similar. However, if we pay attention to the relative error, some concepts need to be identified in order to evaluate the simulation results.

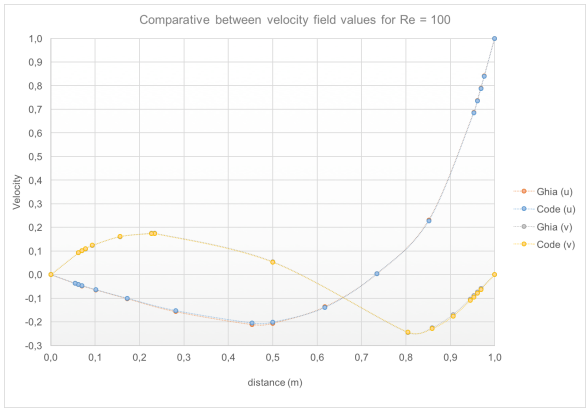
The error made at the point where the u-component of velocity turns negative is quite high when compared with others: 28 %. This is due to the fact of natural calculation of relative errors. When dealing with numbers which are on the order of 10^{-3} or more, the differences between numbers to be compared are very large from a relative point of view. In absolute terms, the error made at this point is similar to the other ones and thus, the high value of the relative error does not represent at all the simulation quality. We can check this if we take a look to Figure 5.10(a), where it is not possible to see such a large error at any point.

The relative error near the motionless walls is similar in nature to the previous error. The velocity values in these areas are relatively small in comparison to the rest of the domain and the relative error is slightly higher. It is also important to highlight that, as Reynolds number increases, the velocity gradients grow near the walls and, as the mesh is not concentrated towards these points, is not possible to correctly represent the movement of the fluid in the cavity.

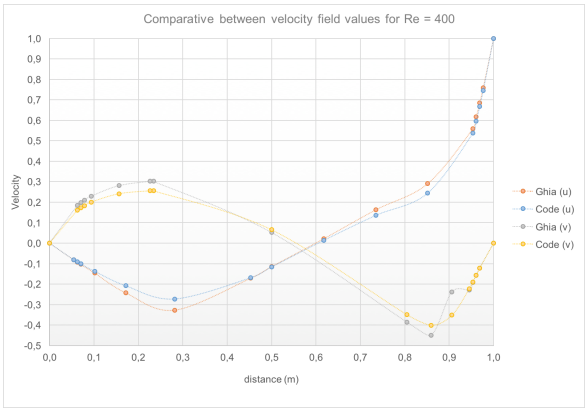
Comparison of velocity field through graphs

We can also compare the values graphically in Figures 5.10 to observe the relative error in an easier way. The coincidence between profiles is not total but it is quite good. Nevertheless, it is

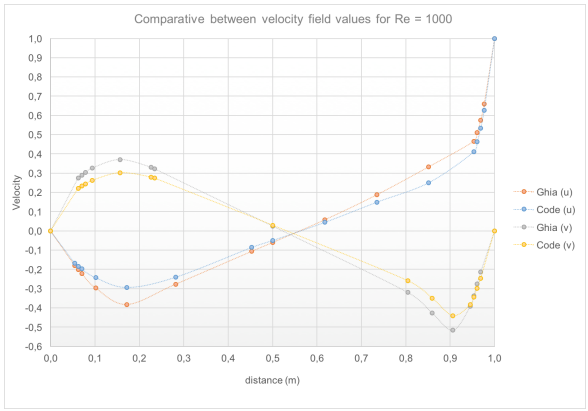
worth commenting some details:



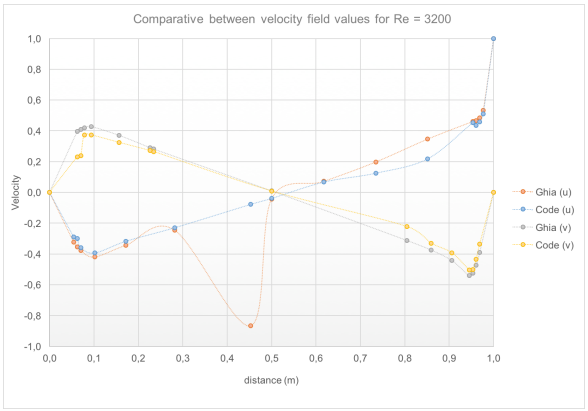
(a) Re = 100



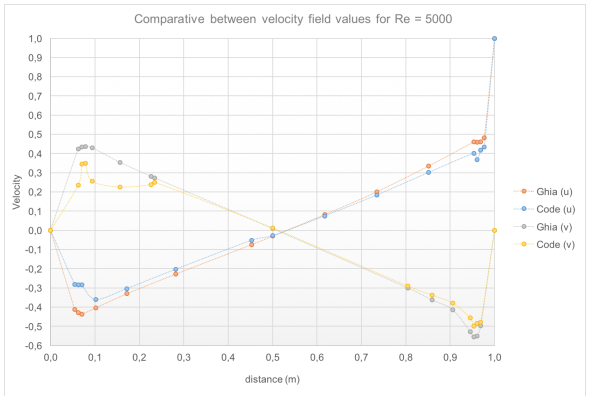
(b) Re = 400



(c) Re = 1000



(d) Re = 3200



(e) Re = 5000

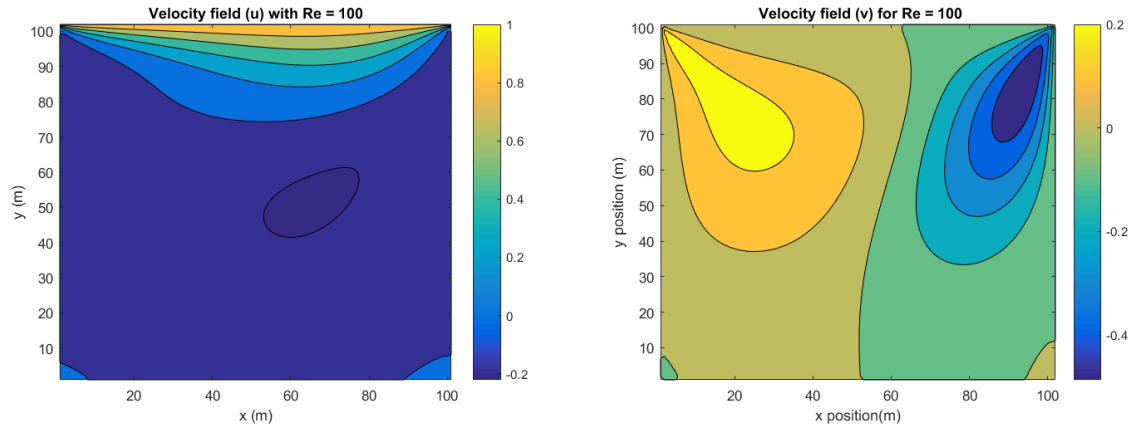
Figure 5.10: Graphical error between benchmark values and code values.

- **Case Re = 100:** This is the case with better precision in relation to the comparison with Ghia. Recall that the relative error of 28 % refers to this graph. However, in view of the graphical comparison, it is needless to say that this error is not representative.
- **Case Re = 5000:** The high relative errors in the calculation of the horizontal component of the velocity in this case appear due to the mesh density used (50x50 nodes). This mesh turns out to be insufficient, causing these error magnitudes. The use of this mesh,

however, results from the limits imposed by the code itself, either by the computational requirements of a case which is that close to turbulence.

Once the validity of the results obtained has been discussed, the objective is to analyze the physics of the problem and to understand the fluid behavior. To do this, the horizontal and vertical velocity are plotted below.

Case Re = 100



(a) Velocity (u) field.

(b) Velocity (v) field.

Figure 5.11: General results for Re = 100.

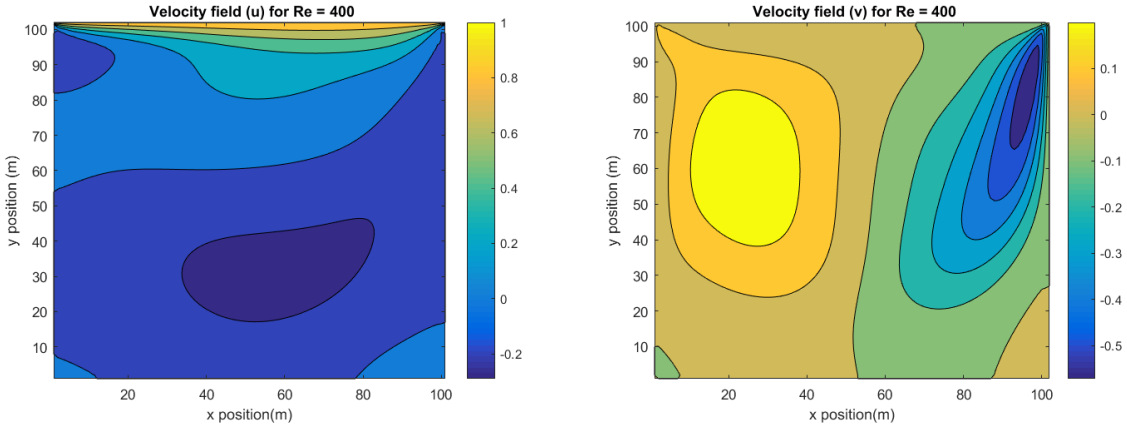
Figure 5.11(a) clearly distinguishes two zones: An upper zone where the fluid reaches its highest velocity (just in the wall, where the boundary condition is $u = 1$) and a lower area closer to the center of negative velocity, where the flow is moving from the right to the left area of the cavity.

Figure 5.11(b) shows how the fluid on the right area of the cavity has a high velocity in the negative direction, while on the left side it has a positive velocity. This clearly indicates that the flow decreases on the right side and rises to the left.

Both figures show the beginning of the generation of two re-circulation zones in the lower corners of the cavity. These areas will acquire importance as the Reynolds increases, since it is a phenomenon strongly related to turbulence. As the size of these areas increase, the stronger is their influence and this causes the cavity to enter a turbulent regime.

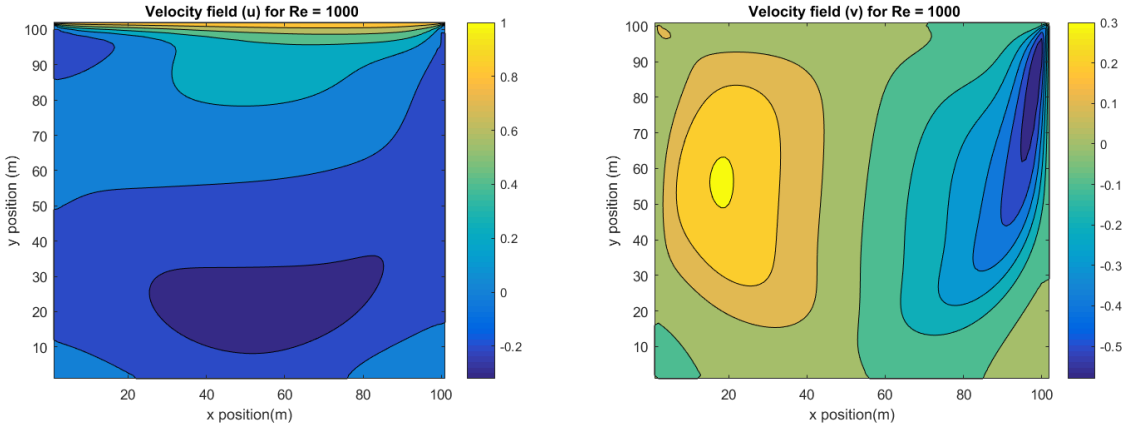
Cases Re = 400, Re = 1000 and Re = 3200

In both horizontal and vertical components of velocity the same phenomenon is observed: due to the drop of the convective term, the areas become more sharp and elongated, extending all along the streamlines.



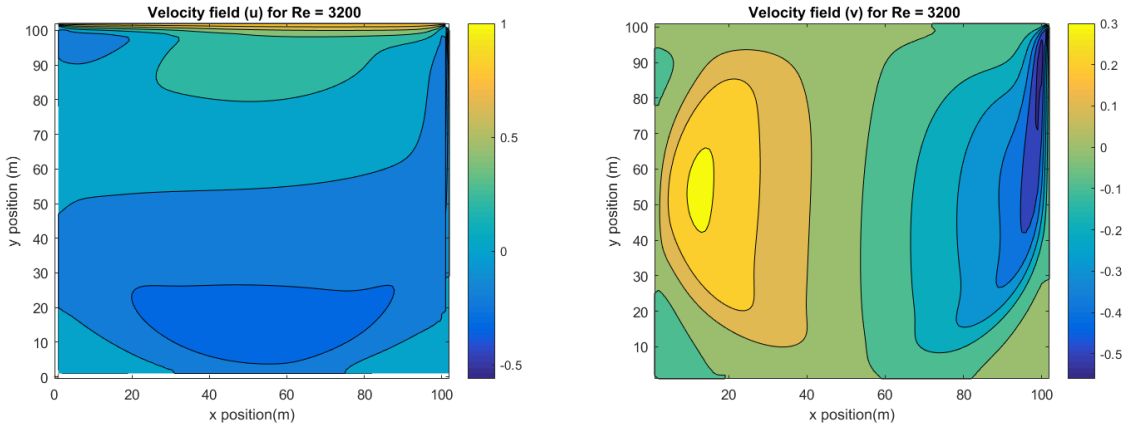
(a) Velocity (u) field. (b) Velocity (v) field.

Figure 5.12: General results for Re = 400.



(a) Velocity (u) field. (b) Velocity (v) field.

Figure 5.13: General results for Re = 1000.



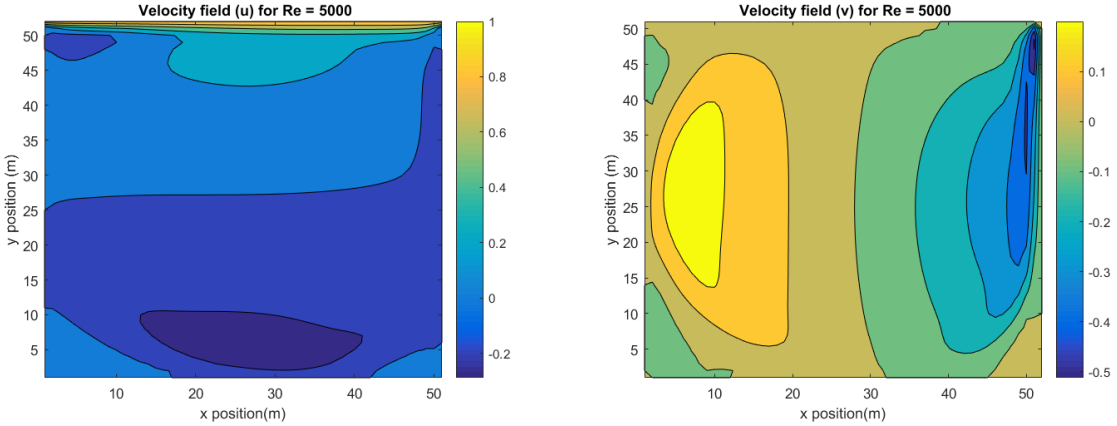
(a) Velocity (u) field. (b) Velocity (v) field.

Figure 5.14: General results for Re = 3200.

Figures 5.12(a), 5.13(a) and 5.14(a) also distinguish two zones. The difference now is that the upper zone where the fluid reaches its highest velocity gets smaller as Reynolds increases. For low Reynolds, the top velocity does not affect due to the fact that viscous forces are dominant but, as Re gets higher, the convective terms gain force and, therefore, this maximum velocity area decreases and the eddy is well centred inside the cavity.

These figures also show that the growth of the re-circulation zones in the lower corners of the cavity. This proves that these areas acquire importance as the Reynolds increases.

Case Re = 5000



(a) Velocity (u) field.

(b) Velocity (v) field.

Figure 5.15: General results for Re = 5000.

Figure 5.15(a) offers an intuitive view of what happens when the flow transition between both states takes place.

Comparing Figures 5.11(a) and 5.15(a), it can be seen a significant change in the velocity gradient at the upper wall. The drag zone of the fluid has been considerably reduced due to the Reynolds increase.

Because the fluid at Re = 5000 has more energy than the fluid at Re = 100 due to less viscous dissipation or an increase in velocity, the area in which the fluid has a negative velocity is displaced down in Figure 5.15(b).

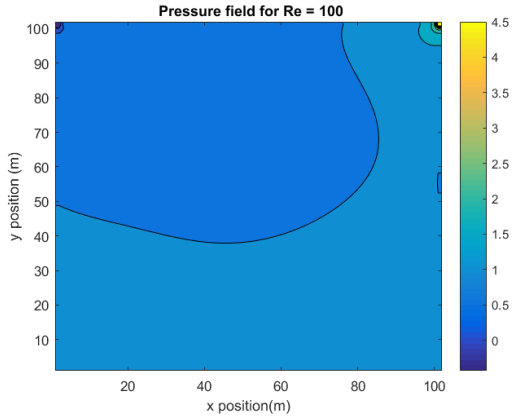
Finally, it is important to notice two facts that prove that the flow is approaching the turbulent regime:

1. The growth of the re-circulation zones in the lower corners of the cavity. This proves that as these areas grow in both number and size, easier is the transition to turbulent flow.

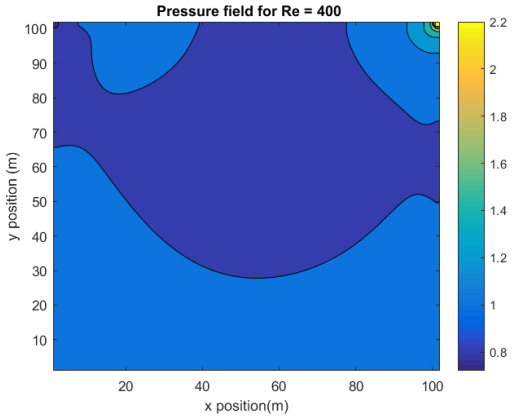
2. Velocity gradients near $x = 0$ and $x = 1$ become more sudden. This indicates that, as boundary layer phenomena become more important, the flow becomes more likely to enter turbulent regime.

Comparison of pressure field through graphs

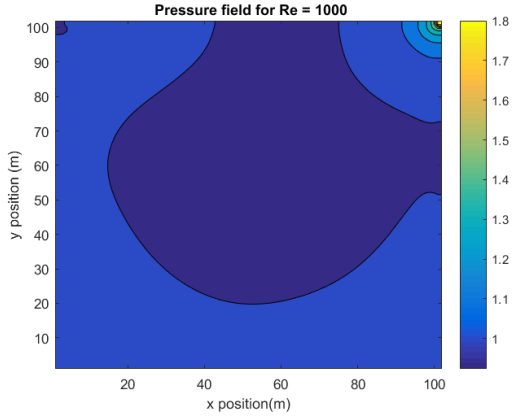
The pressure field is analyzed only through graphs due to the fact that values of pressure field are not relevant since there are not boundary pressure values. Based on this, it is not strange to see negative pressures: this can be solved by fixing a pressure value in one node. What is relevant in the driven cavity case is the pressure gradients within the mesh. Figure 5.16 shows the pressure map of the domain with different Reynolds numbers.



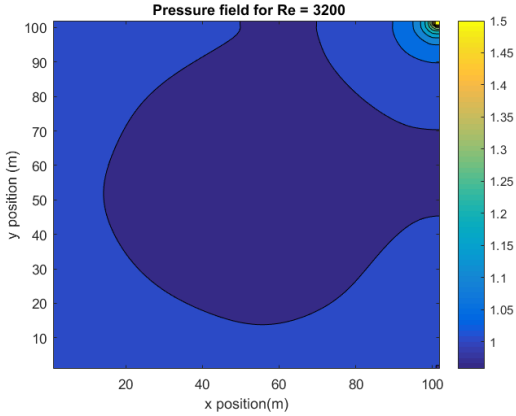
(a) Re = 100



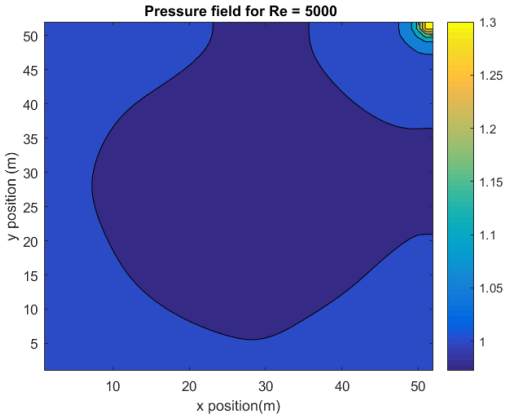
(b) Re = 400



(c) Re = 1000



(d) Re = 3200



(e) Re = 5000

Figure 5.16: Pressure field results.

As we can see, the conservation of energy is fulfilled in this problem: Points with maximum speeds have the minimum pressures and the same happens in the opposite way. The upper left corner suffers bigger depressions, while the upper right corner encounters high pressures. This is normal since, by the movement from left to right of the upper wall, the left part is emptied of fluid and it is sent to the right part.

Study of mesh size

The results obtained by the own code have been compared satisfactorily with those presented in [16] and a complete analysis of the physics has been presented. However, the influence of the mesh on the results has not been discussed. In this subsection, the mesh size will be analyzed in two different cases. The first case to be analyzed will be the Re = 100 case due to the fact that it is the simplest one. The second case could have been Re = 5000, but given that it is the case in which some stability problems begin to occur, it has been chosen to study the Re = 1000 case. The different mesh densities will be compared through the obtained results in relation with the u-velocity field.

Case Re = 100

The driven cavity case with Re = 100 has been solved numerous times with different uniform meshes: [10x10; 20 x 20; 50 x 50; 70 x 70; 100 x 100]

The convergence criteria used to carry out the analysis and to make sure the code reached a steady state is 10^{-5}. The different results are summarized in next figure:

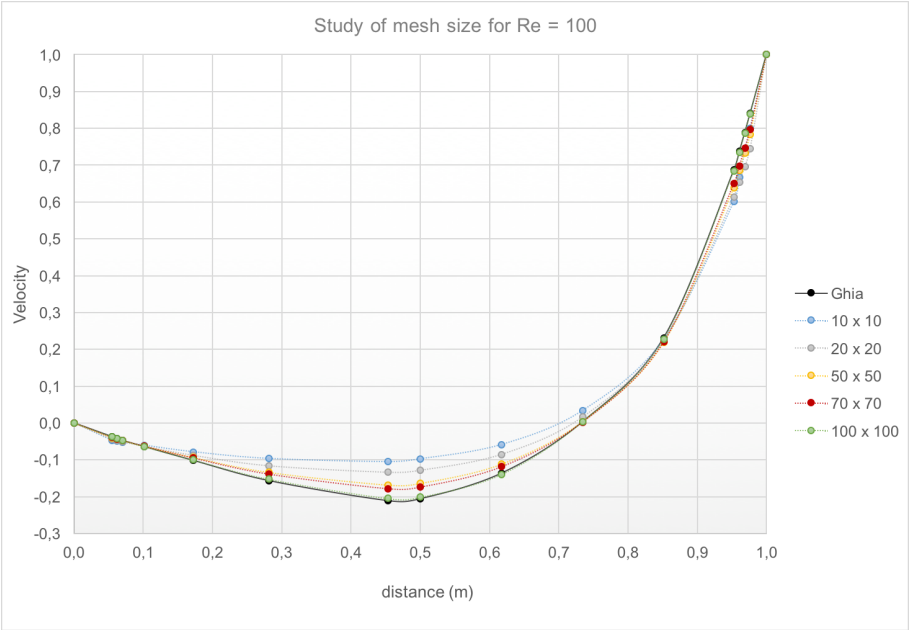


Figure 5.17: Study of mesh size for Re = 100.

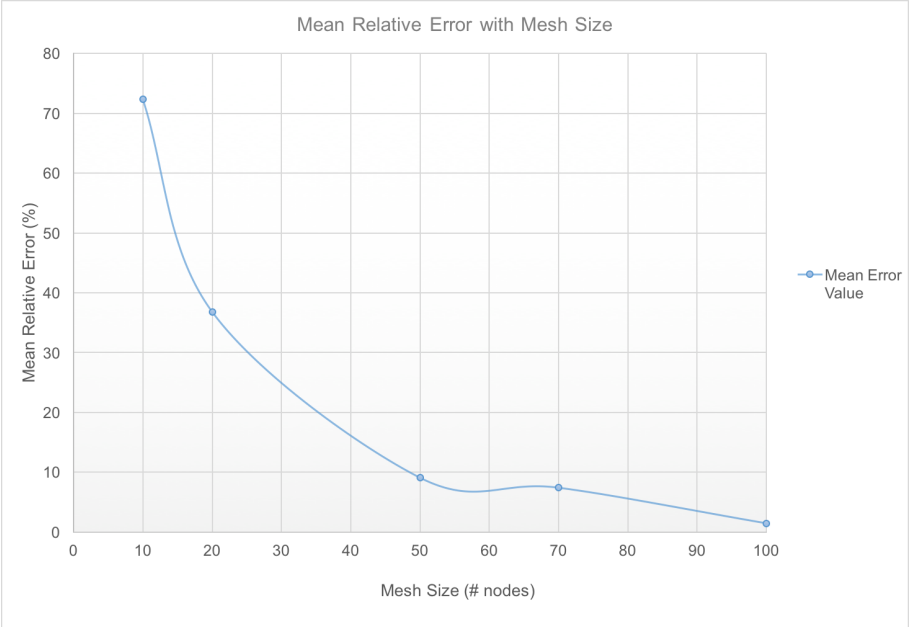


Figure 5.18: Study of variation of mean relative error with mesh size for Re = 100.

As we can see in Figure 5.17, with a mesh size of 100x100 the obtained results are very similar to benchmark values. However, with 50x50 or 70x70 meshes we get good enough results and computational time is reduced. Thus, this mesh can be considered good enough.

Case Re = 1000

The driven cavity case with Re = 1000 has been solved numerous times with different uniform meshes: [10x10; 20 x 20; 50 x 50; 70 x 70; 100 x 100]

The different results are summarized in next figure:

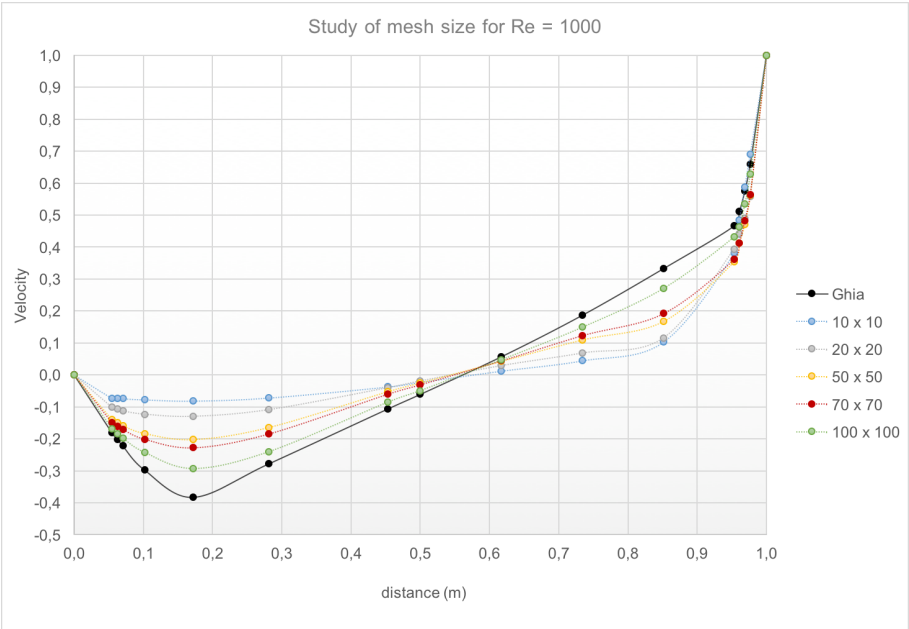


Figure 5.19: Study of mesh size for Re = 1000.

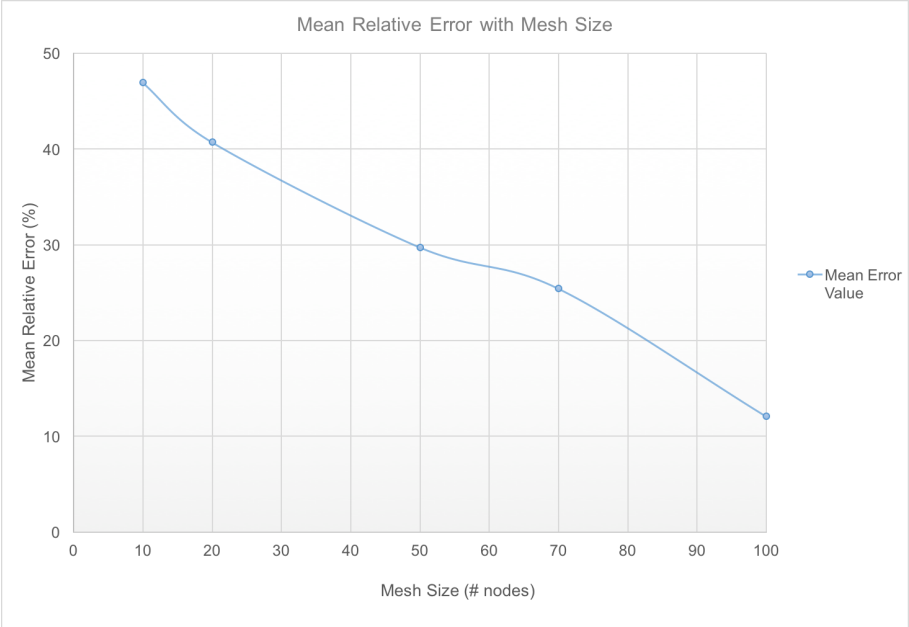


Figure 5.20: Study of variation of mean relative error with mesh size for Re = 1000.

As we can see in Figure 5.19, with a mesh size of 100x100 the obtained results are not good enough. For this reason, it would be advisable to carry out the study with a mesh density of 150x150.

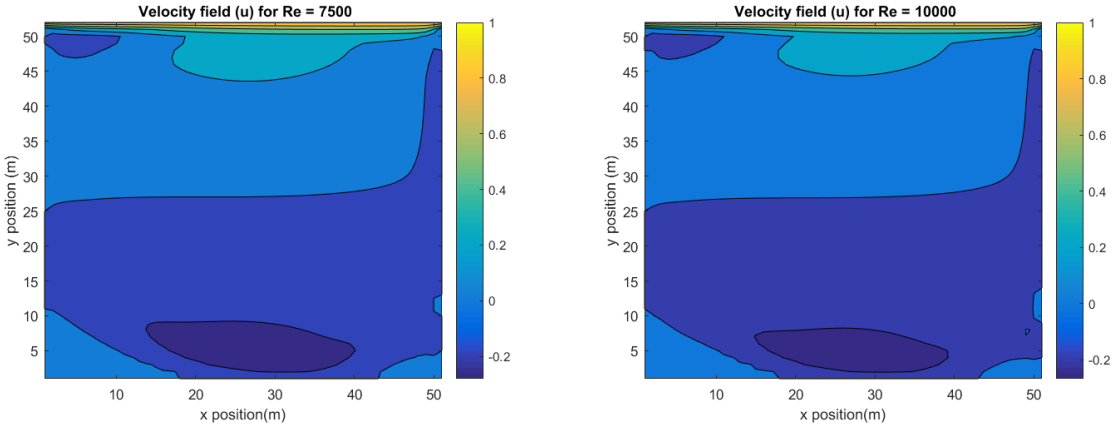
We can also see how does the mean relative error value decrease when the mesh size increases: see Figures 5.18 and 5.20. We can see how the average error decreases much more in low Reynolds cases.

Although good results are obtained with the mentioned meshes, results could be improved by implementing non-uniform meshes. With non-uniform meshes, simulation accuracy is gained: The nodes are placed closer to the walls, where more information is required to correctly represent the strong gradients.

Study of higher Reynolds cases

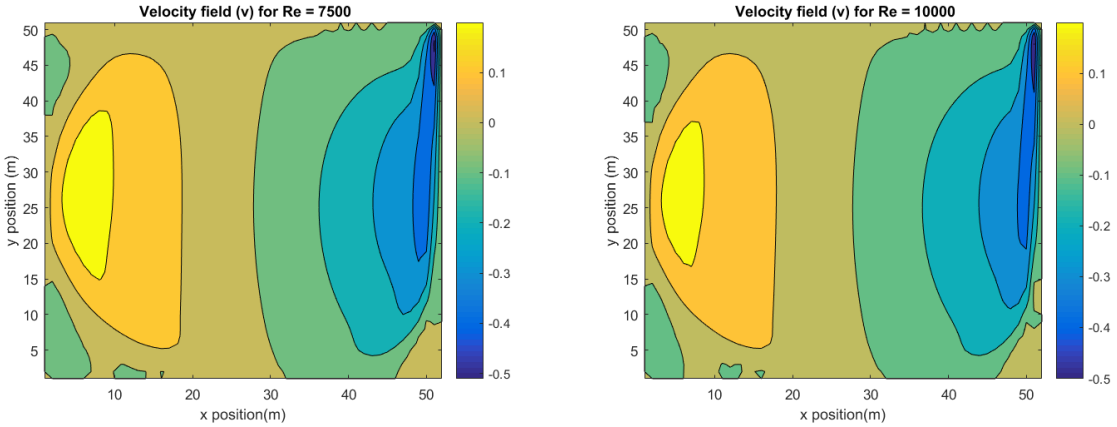
When Reynolds numbers are over 5000, some stability problems begin to occur and the flow could begin to enter turbulent regime. Results obtained for Reynolds cases over 5000 are presented below.

For high Reynolds, the convective term have much more importance than diffusive terms and, for this reason, the eddy gets centred inside the cavity. At the same time, vortex near the walls appear. In these cases, the kinematic viscosity tends to zero and this creates instability problems.



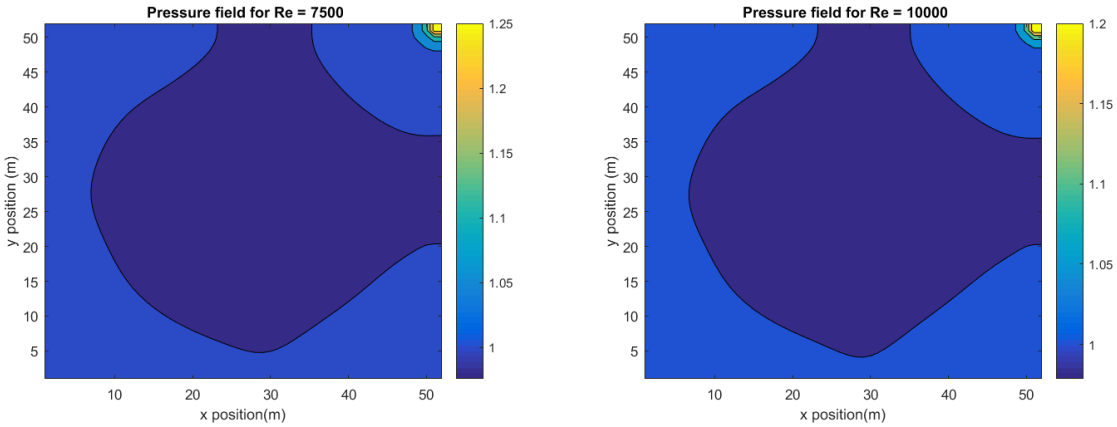
(a) Re = 7500 (b) Re = 10000

Figure 5.21: Velocity field (u) for Re = 7500 and Re = 10000.



(a) Re = 7500 (b) Re = 10000

Figure 5.22: Velocity field (v) for Re = 7500 and Re = 10000.



(a) Re = 7500 (b) Re = 10000

Figure 5.23: Pressure field for Re = 7500 and Re = 10000.

5.2.11 Conclusions

Conclusions of this case are mainly confined to the used parameters and numerical schemes. That is because conclusions related to the flow behaviour have been already presented in the previous section.

The results presented in the previous sections lead us to the following conclusions:

- From simulations carried out with meshes 10x10 to 100x100, it is concluded that 100x100 meshes are adequate for moderate values of Reynolds.
- The relative error is bigger near the walls. This can be solved increasing the mesh density or using a non-uniform mesh.
- As Reynolds increases, mesh refinement is important in order to obtain better results.
- The CFL condition does not always guarantee the convergence. Therefore, a timestep corrective factor of 0.5 was included, as recommended in [5].
- The relative error becomes bigger for high Reynolds numbers even if the mesh density is increased. The reason of it is the turbulence of the flow.
- The flow shows the logical movement: the top border causes a horizontal displacement of the fluid until collides with the right wall.

Chapter 6

Environmental impact

The direct environmental impact of this study is related to the study process and it is almost null. However, the electricity consumed by the computer generated CO_2 emissions which must be taken into account.

According to the *Budget Study*, the study has lasted 485 hours. During this period, a personal computer with a mean consumption of 15 kWh per month has been used. The full study has consumed 75 kWh. According to Reference [9], each kWh produces approximately 0,33 kg of CO_2. Thus, this study has produced a 24,75 kg of CO_2.

As seen, CO2 emissions are relatively low. However, on a possible future study, a computer with more execution capacity could be used in order to reduce these emissions.

Chapter 7

Conclusions and future actions

The main objective of the present study was to provide the author with experience in the field of heat transfer. Starting from the medium knowledge of fluid dynamics and a basic knowledge of Matlab programming, the aim was to create codes in order to solve the Navier-Stokes equations, which govern the phenomenology of fluid mechanics.

To achieve these objectives, some reference cases were studied. Throughout the study cases, it has been easier to understand the phenomenology of numerical methods, to acquire experience in heat transfer field and to achieve the final objective: to solve the Navier-Stokes equations.

At this point of study it could be said that the objectives set at the beginning of this project has been successfully accomplished. If something has to be pointed out regarding the last part of the study (the NS equations resolution) it is that the developed code for the differentially heated cavity has not been totally completed and thus, a longer study on FSM and, concretely, the energy equation, will be carried out throughout this case.

Finally, it makes sense to discuss on the future research. If the project could go forward in a hypothetical case, the actions to consider would be:

- First, to carry out a longer study on the FSM throughout the differentially heated cavity case. This further study would imply the introduction to non-uniform meshes, energy equation, etc.
- To improve skills in *Matlab* and, maybe, start programming with *C++* language. This way, codes could be more efficient and difficult cases could be solved with more accuracy.
- To improve personal skills to carry out validation and verification of codes.
- To get into a more deep study about the involved phenomenology in turbulent regime in order to extrapolate the acquired experience to turbulent flows.

Bibliography

[1] André Bakker. *Applied Computational Fluid Dynamics*. 2002. URL: https://bit.ly/2LpT4s2.

[2] A.J. Chorin. *Numerical solution of the Navier-Stokes equations*. Vol. 22. New York University: Mathematics of Computation, 1968, pp. 745–762.

[3] CTTC. *A Two-dimensional Steady Convection-Diffusion Equation: the Smith-Hutton problem*. 2018. URL: https://bit.ly/2sCHWRy.

[4] CTTC. *A Two-dimensional Transient Conduction Problem*. 2018. URL: https://bit.ly/2sBaxqu.

[5] CTTC. *Fractional Step Method: Staggered and Collocated Meshes (Course on Numerical Methods in Heat Transfer and Fluid Dynamics)*. 2018. URL: https://bit.ly/2M5dB6D.

[6] CTTC. *Verification strategies for the convection-diffusion equation*. 2018. URL: https://bit.ly/2Jfxg5S.

[7] M. Darwish F. Moukalled L. Mangani. *The Finite Volume Method in Computational Fluid Dynamics*. 1st ed. Switzerland: Springer, 2016, p. 817. ISBN: 978-3-319-16873-9.

[8] M. Darwish F. Moukalled L. Mangani. *The Finite Volume Method in Computational Fluid Dynamics*. 1st ed. Switzerland: Springer, 2016, pp. 40–45, 561. ISBN: 978-3-319-16873-9.

[9] Energía y Turismo y Ministerio de Fomento Ministerio de de Industria. *Factores de emisión de CO2*. 2018. URL: https://bit.ly/2s4v7wy.

[10] J.Y. Murthy and S.R. Mathur. *Numerical Methods in Heat, Mass, and Momentum Transfer*. 1st ed. Purdue: School of Mechanical Engineering Purdue University, 2002, pp. 21–36, 125–126.

[11] O. Darrigol. *Between Hydrodynamics and Elasticity Theory: The First Five Births of the NS Equation.* 1st ed. Archive Historic Exact Science, 2002.

[12] A. Pérez. *Resolución Numérica de las Ecuaciones de Navier-Stokes*. 2018. URL: https://bit.ly/2sB8bbn.

[13] R.M. Smith and A.G. Hutton. *The numerical treatment of advection: a performance comparison of current methods. Numerical Heat Transfer.* 5th ed. Mc Graw Hill Education, 1982, 439–461.

[14] Suhas V. Patankar. *Numerical Heat Transfer and Fluid Flow*. 1st ed. 1221 Avenue of the Americas (NY): Mc. Graw Hill Education, 1980, 56 to 58. ISBN: 0-07-048740-5.

[15] R. Temam. *Sur l'approximation de la solution des équations de Navier-Stokes par la méthode des pas fractionnaires*. Vol. 33. Berlin/Heidelberg: Springer Berlin/Heidelberg, 1969, pp. 377–385.

[16] K.N. Ghia U. Ghia and C.T. Shin. *High-Re Solutions for Incompressible Flow using the Navier-Stokes Equations and a Multigrid Method*. Journal of computational physics. University of Cincinnati, Cincinnati, Ohio: Mathematics of Computation, 1982, pp. 387–411. ISBN: 48,

[17] Dormund Uni. *Lecture 8: Time - stepping techniques*. 2018. URL: https://bit.ly/2Jz9ReZ.

[18] Unknown. *Numerical Methods for Differential Equations*. 2018. URL: https://bit.ly/2HtAiBH.

[19] Afshin J. Ghajar. Yunus A. Çengel. *Heat and mass transfer. Fundamentals Applications.* 5th ed. New York: Mc Graw Hill Education, 2015, p. 17. ISBN: 978-981-4595-27-8.