A NUMERICAL PERFORMANCE ANALYSIS OF HEAT RECOVERY VENTILATORS WITH STAGGERED-BAFFLE CHANNELS

A Thesis

by

FELIPE LEITE ASSUNCAO

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Chair of Committee,	Michael B, Pate
Committee Members,	Timothy J, Jacobs
	Pavel V, Tsvetkov
Head of Department,	Andreas Polycarpou

August 2016

Major Subject: Mechanical Engineering

ABSTRACT

This thesis presents a 2D numerical analysis of a cross flow compact energy recovery ventilator (HRV) with staggered baffle channels, using finite difference iterative method. The model was developed by discretization of the momentum and continuity equation into convective-diffusive terms and applying the power law scheme. Solving the system of equations required the combination of the Gauss-Seidel iterative method and the tridiagonal-matrix direct method applied to a staggered velocity and pressure grid. The Semi-Implicit method for the pressure-linked equations algorithm (SIMPLE) was chosen for coding of the program. The simulation was carried out to determine the effects of baffle height h/Dh, baffle spacing S/Dh and Reynolds number on thermal and flow performance, therefore it was necessary to vary one of the parameters within a range while all others were kept constant. The results showed that the baffle height has the greatest effect on the overall performance. At greater baffle heights the pressure drop, the Nusselt number, and heat transfer effectiveness was increased. The baffle pitch has the least effect on the overall performance; with increasing baffle pitch the pressure drop decreased while the Nusselt number had a slight increase prior to stabilization. The change of Reynolds increased the pressure drop and Nusselt number but reduced the residence time in the channel, diminishing the heat transfer effectiveness.

DEDICATION

I dedicate this first to my family, without them I would never be where I am today. I am very thankful that they were always supportive of my choices and have provided guidance and been there for me during the good and the bad times. Second I would like to thank my girlfriend for all her love and support and for helping me grow for all these years together. I would also like to acknowledge my professors from my undergraduate studies and Brazil, for giving me the necessary background to be where I am now, and the professors at Texas A&M University that have deepened my engineering skills, specially to my advisors with a closer feedback.

TABLE OF CONTENTS

Page

LIST OF FIGURES

Page

LIST OF TABLES

Page

CHAPTER I

INTRODUCTION

With increased energy consumption as a result of economic growth, competitiveness and the necessity to reduce overall emissions of greenhouse gases have become vital for success of modern organizations. The HVAC industry has a crucial role for global economy to reach target greenhouse gas emission levels, as a result of energy consumption by sector in industrialized countries, which accounts for 33% of total energy consumption Zhang (2008). They are more than ever, driven to provide more sustainable and energy efficient systems with comfort and air quality for a wide range of operating conditions.

One of the most effective methods for heat transfer enhancement is through the use of baffles placed on channel walls, due their high thermal loads for decreased dimensions. This occurs mainly because of the growth interruption of the hydrodynamic and thermal boundary layer, as at reattachment heat transfer rate increases. This method is commonly used in modern engineering applications such as in compact heat exchangers, solar collectors and in electronics.

To predict effectiveness of heat transfer design, numerical analysis has become a common engineering task. Current available models provide reliable results eliminating the necessity of prototypes and tedious testing. This has become a powerful resource for organizations as they are able to get much faster response of design model performance,

eliminating the bottleneck of the design process to define the optimal arrangement much quicker.

Objectives

The goal of this work is creating a numerical model to simulate the effects of the dimensions of baffle height, baffle pitch and Reynolds number on the overall heat transfer effectiveness of an HRV. The numerical model is implemented by applying a finite difference method to the general convection and diffusion differential equation using Matlab, and a simulation is carried out, changing one variable at a time such as baffle height, baffle pitch and Reynolds number, while the others are kept constant.

CHAPTER II

LITERATURE REVIEW

Compact Heat Exchangers

Compact heat exchangers are widely used, due their high offered area density, or in other words the ratio of heat transfer surface to the total volume of the heat exchanger. This results in a reduction volume and weight, and also in improved performances and lower costs other convectional heat exchanger designs.

Heat Recovery Ventilation Systems

HRVs are a process for recovering heat from an air to air heat exchanger. Schurcliff (1988) defines heat or energy recovery as any device that removes, recovers or salvages heat or mass from one air stream and transfers it to another airstream. In other words the air that would be discarded into the outside environment is used to preheat or precool incoming air of and HVAC system.

Generally Heat Recovery Ventilation systems are separated by the sensible heat recovery or sensible and latent heat recovery. While the first only exchanges sensible heat between incoming and exhausted air stream, the second also exchanges latent heat by means of mass exchange of moisture through a porous membrane.

Heat recovery systems typically recover about 60%-95% of the heat in exhaust air and have significantly improved the energy efficiencies of buildings Idayu and Riffat (2012). Other advantages than the energy savings, is the reduced loss of heat and

effective ventilation, which can be extremely important in environments where open windows are a security risk.

There are numerous types, sizes, configurations and flow arrangements of heat recovery units which all depend on the heat exchanger core. They can be classified based on geometry, flow arrangement and number of different working fluids. The main types of recovery systems that are currently used are the fixed plate, heat pipe, and rotary wheel.

Fixed-Plate Heat Recovery System

The Fixed plate heat recovery system is the most common type, due to the simplicity of arrangement combined with a high efficiency. They are usually constructed by thin plates stacked together. The plates can be smooth, have baffles or other disturbances for enhancing heat transfer. The schematic of a cross flow fixed-plate heat recovery system is shown in Figure 1. Typical effectiveness of sensible heat transfer is 50-80% and air flow arrangements can be counter-flow, cross-flow and parallel flow ASHRAE (2005).

A study performed by Han H et al (2007) investigated the effects of outdoor weather conditions and performance of the plate-type heat recovery ventilators. The experiments were carried out to measure efficiencies, while varying the outdoor conditions and maintaining fixed indoor heating/cooling conditions. The results showed that during the winter the efficiencies are higher than during the summer, mainly due to heat generation of the fan.

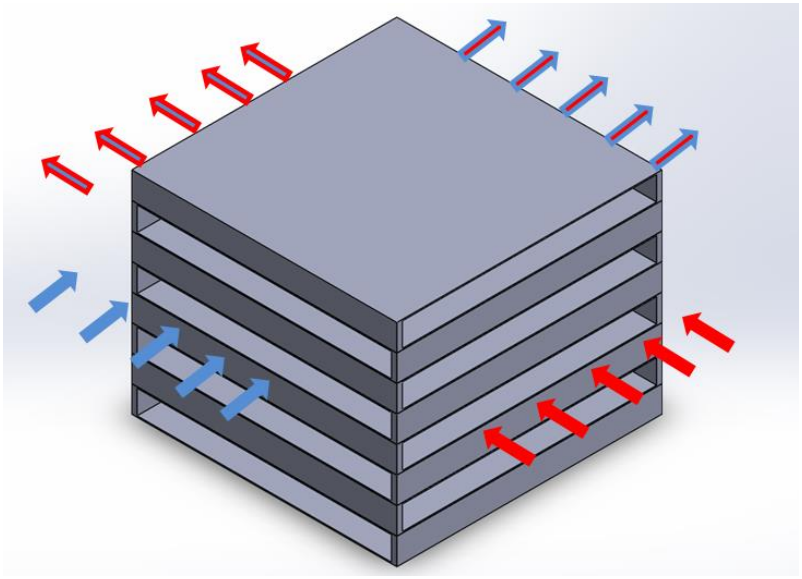

Figure 1. Fixed-Plate Heat Exchanger

The main disadvantages of this model of compact heat exchanger is the high pressure drop for the equivalent flow velocities, this is due to the narrow passages generating high velocities and resulting in inefficient transverse vortices Li et al. (2011).

The fixed-plate heat exchangers with disturbances of fins or baffles are denoted as plate-fin compact heat exchangers. The main function of the disturbance is to act as a secondary heat transfer surface increasing the secondary heat transfer surface area, consequently reducing the thermal resistance and also increasing the total heat transfer from the different air-flow channels of same temperature gradient.

Heat exchange enhancement occurs through the increase of area density, also due to mixing which generates complex secondary flow and boundary layer separation. Surface interruption can also enhance heat transfer, by preventing continuous growth of the thermal boundary layer with periodical interruption.

Thus, thicker boundary layers present in a continuous flow offer greater resistance to heat transfer, while interruption of a thin boundary layer offers lower resistance to heat transfer. This enhancement mechanism occurs even at low Reynolds numbers when flow is steady and laminar. Above critical numbers, the interrupted surfaces can offer an additional mechanism of heat transfer enhancement by introducing oscillations in the flow in form of vortex shedding. Vortices start near the leading edge of the interruption and subsequently travel downstream in the wake. They act as large scale mixers and continuously bring fresh fluid from the stream towards the surface in the upstream of the interruption and eject the fluid away from the surface on the downstream side of the interruption. It is important to note that, there is an associated increase in pressure drop and consequently, greater pumping power is required. The increased pressure drop is mainly caused due to an increase of the friction factor associated with the periodic restart of the hydrodynamic boundary layer. For an optimal design, heat transfer enhancement provided by design parameters must also account for their effect on increasing the required pumping power.

Developing Flow

Albakhit et al (2005) investigated numerically parallel flow and heat transfer in compact heat exchangers channels using the hybrid method. Developing laminar flow was studied in two parallel rectangular channels, and it was found that the developing region leads to a higher overall heat transfer coefficient.

Periodically Fully Developed Flow

Patankar et al. (1977) were the first to present the concept of periodically fully developed flow to investigate the flow characteristics and heat transfer in ducts. For a module of length Lx in the stream wise direction (x), with the cross direction being denoted by y, the velocity profile repeats periodically:

$$\begin{aligned} u(x,y) &= u(x+L_x,y) = u(x+2L_x,y) = \cdots \\ v(x,y) &= v(x+L_x,y) = v(x+2L_x,y) = \cdots \end{aligned} \tag{1}$$

For the heat transfer and pressure drop problem the variable does not repeat from module-to-module, as the case for the velocities presented in Equation (1). Although, the gradients repeat from module-to-module,

Thus,

$$\begin{aligned} \beta &= \frac{P(x,y) - P(x+L_x,y)}{L_x} = constant \\ \gamma &= \frac{T(x+L_x,y) - T(x,y)}{L_x} = constant \end{aligned} \tag{2}$$

The modules that satisfy Equation (1) and Equation (2) are shown in Figure igure 2.

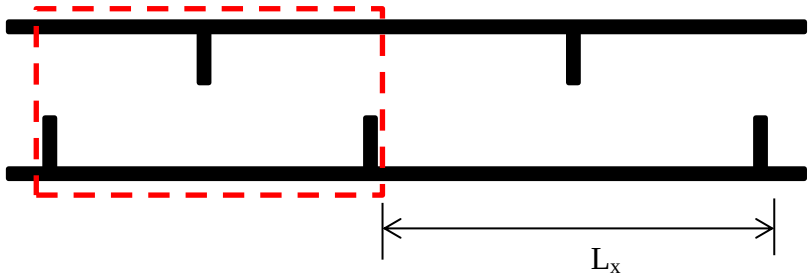

Figure 2. Flow and Energy Module

Baffle Ribs

Previous works have shown that that obstruction of flow can have negative or positive effect. For this type of heat enhancement technique, design engineers strive to obtain heat transfer enhancement without increasing significantly the pressure drop. A numerical study of parallel plates with staggered transverse baffles, with constant heat flux boundary condition performed by Webb and Ramadhyani (1985) has shown that by varying the Reynolds number from 340 to 1400, while maintaining the baffle spacing and baffle height to h/Dh=0.25 and S/Dh=1 respectively, the point of reattachment is no longer sensitive to the Reynolds number beyond Re=940, but the size of the upstream recirculation eddies keep increasing.

Decreasing the rib spacing S/Dh while maintaining a constant Reynolds number of Re=340, the authors found that the upstream recirculating eddies occupy more of the

available space until a full recirculation cell is formed, isolating the core flow from the walls.

Increasing the rib height h/Dh while maintaining a constant Reynolds number of Re=340, the streamlines are deflected and flow impinges with greater strength against the opposite walls, thus the size and the recirculation zone also increases and higher velocities are found at the core of the flow. The study concluded that the Nusselt number and friction factor increase with Reynolds number. Also high Prandtl number fluid- such as water- yields a significantly higher Nusselt number combined with comparatively low friction factor, when compared to a fluid with lower Prandtl such a gas.

Another numerical study by Rowley and Pantakar (1984) in circumferential in-line finned tubes concluded the contrary, showing an overall decrease in the Nusselt number for low Prandtl number fluids, due to complete detachment of fluid from the tube wall.

A finite volume numerical simulation done by Mousavi and Hooman (2005) investigated fluid flow and heat transfer in the entrance region of a two dimensional horizontal channel with staggered baffles and with isothermal walls. As previously mentioned, the flow detaches from the walls upstream of the baffles, creating a recirculating zone. Downstream of the recirculating zone the flow is reattached and the local Nusselt number will reach its peak value. The investigation showed that increasing the height of the baffles h/Dh in the ranges of (0.25 to 0.5) will lead to a greater change

in friction factor and also Nusselt number, in that sense one may conclude that the optimal heat transfer design, based on minimal pressure drop may be within that range.

Staggered Baffle Ribs and In-Line Baffle Ribs

Kelkar and Pantakar (1987) solved a parallel plate problem with staggered baffle ribs and based on their result the staggered baffle arrangement performs better than the in-line arrangement. According to their results the staggered arrangement causes the flow to deflect and impinge on the walls, while the in-line configuration only helps the flow detach from the walls of the channel, thus reducing the heat transfer performance.

A similar study by Cheng and Huang (1990) investigated the effects of geometric parameters and Reynolds number on heat transfer coefficients and friction factor for parallel-plate of transverse fins using a finite difference stream-function vortices transformation. The study concluded that flow attains a fully-developed periodic profile after a short entrance length and in that region the higher fins have greater influence on the friction factor ($h/Dh > 0.3$). As for the overall heat transfer coefficient the correct arrangement of the fins are of great importance in order to operate at higher Reynolds numbers which enhance heat transfer, without generating flow detachment and large recirculation zones that create an insulating effect.

The results indicated that the staggered arrangement behaves more efficiently than the in-line arrangement, due to higher flow detachment of the latter. Although it is important to note that for the staggered arrangement, as the friction factor increases by an order of 100 the heat transfer coefficient will only be increased by 10. To minimize

this effect the fin height should diminish to drive the solution to approach the fully developed flow in smooth channel solution.

Figure 3 shows the arrangement of the staggered baffle ribs in (a) and the in-line baffle ribs in (b) along with the standard geometry dimensions that will be used further sections of this investigation.

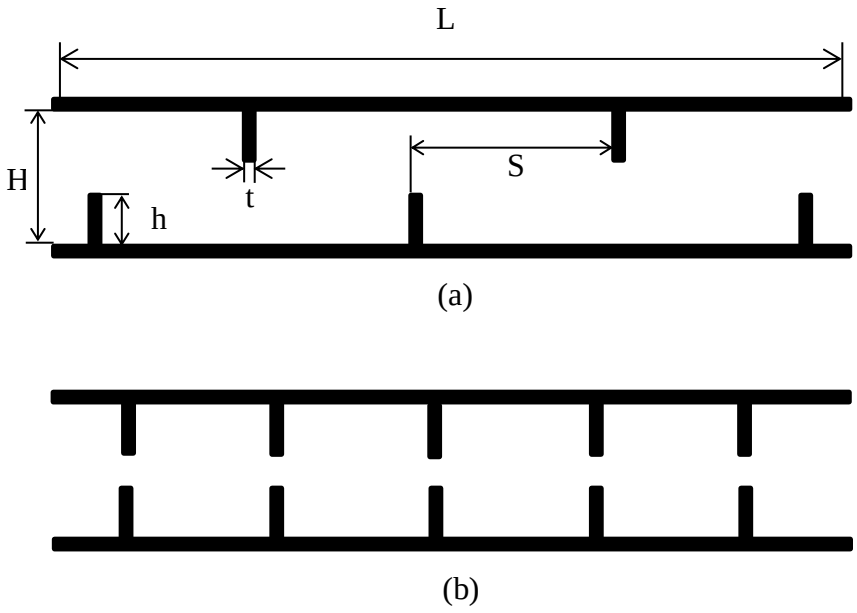

Figure 3. Staggered Baffle Ribs (a) and In-Line Baffle Ribs (b)

Yaici et al (2012) performed a detailed numerical analysis of energy recovery ventilators (HRV/ERV), using computational fluid dynamics, investigating Canadian summer and winter conditions in both con-current and counter-current flow arrangements. The results showed that effectiveness decreases with increasing air velocity, which is attributed to the residence time in the heat recovery system core.

When the velocity of the air flow is low, the air remains in the core of the heat recovery system for more time, permitting a higher amount of heat transfer per mass of airflow, and which in turn enhances the effectiveness. In addition, the effectiveness is higher for the counter-current arrangement when compared to the con-current arrangement.

CHAPTER III

METHODOLOGY

2D - Convection and Diffusion Equation General Form

For given flow field the convection and diffusion equation for a general variable ∅ is given by Equation (3)

$$\frac{d}{dt}(\rho\emptyset) + \frac{d}{dx}(\rho u\emptyset) + \frac{d}{dy}(\rho v\emptyset) = \frac{d}{dx}\left(\Gamma\frac{d\emptyset}{dx}\right) + \frac{d}{dy}\left(\Gamma\frac{d\emptyset}{dy}\right) + S \tag{3}$$

2D - Convection and Diffusion Equation Steady State

For steady state the first term on the left hand side of Equation (3) is eliminated. Additionally the source term S on the right hand side of Equation (3) can be broken down into different terms (body forces, pressure forces, internal generation) depending on the chosen general variable ∅. For now, the S term will not be considered, but will be inserted later for the pressure field in the momentum equation in the following sections.

Consequently

$$\frac{d}{dx}(\rho u\emptyset) + \frac{d}{dy}(\rho v\emptyset) = \frac{d}{dx}\left(\Gamma\frac{d\emptyset}{dx}\right) + \frac{d}{dy}\left(\Gamma\frac{d\emptyset}{dy}\right) \tag{4}$$

The right hand side of Equation (4) represents the convective terms — originated by the bulk motion of fluid— which is linked to the fluid density (ρ) and the velocity components (u) and (v) .

The left hand side represents the diffusivity terms, linked to the flux due to the gradient of the general variable Ø in the x and y direction, where Γ is the dynamic viscosity.

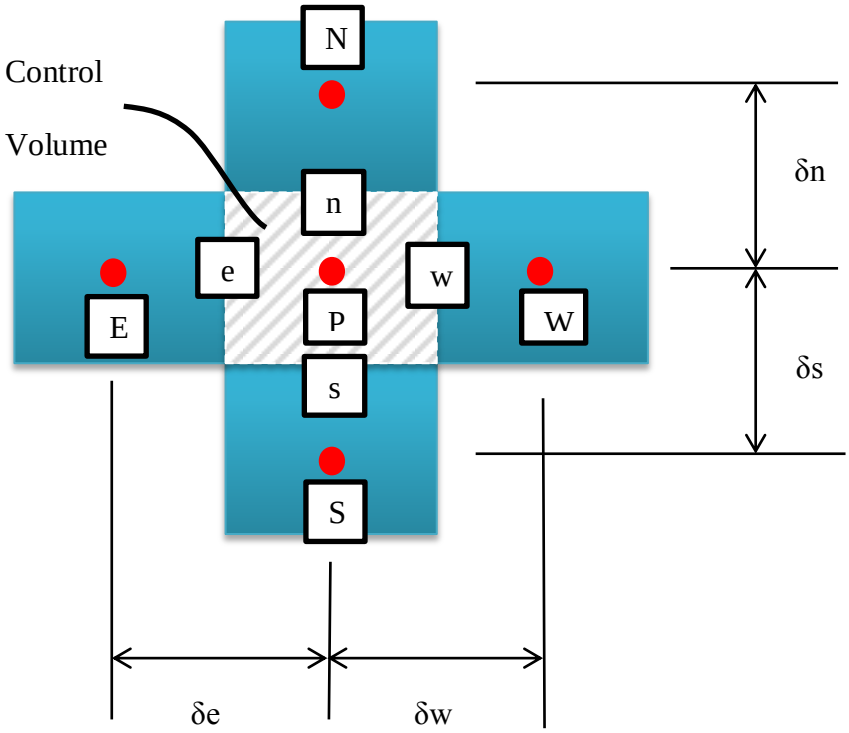

Figure 4. Pressure Control Volume

To derive the discretized equation, the control volume on Figure 4 is used. The control volume faces are placed at e,w,s,n – the mid-point in between the nodes, presented in red.

Primary Derivation

The integration of Equation (4) over the control volume of Figure 4

$$\iint \frac{d}{dx}(\rho u \emptyset)dxdy + \iint \frac{d}{dy}(\rho v \emptyset)dydx = \iint \frac{d}{dx}\left(\Gamma \frac{d\emptyset}{dx}\right)dxdy + \iint \frac{d}{dy}\left(\Gamma \frac{d\emptyset}{dy}\right)dydx \quad (5)$$

Provides

$$(\rho u \emptyset)_e \delta_y - (\rho u \emptyset)_w \delta_y + (\rho v \emptyset)_s \delta_x - (\rho v \emptyset)_n \delta_x = \left(\Gamma \frac{d\emptyset}{dx}\right)_e \delta_y - \left(\Gamma \frac{d\emptyset}{dx}\right)_w \delta_y + \left(\Gamma \frac{d\emptyset}{dy}\right)_s \delta_x - \left(\Gamma \frac{d\emptyset}{dy}\right)_n \delta_x \quad (6)$$

To rearrange Equation (6) more properly, the symbols F and D are defined

$$F_e = (\rho u)_e \delta_y \; and \; F_w = (\rho u)_w \delta_y$$

$$F_s = (\rho v)_s \delta_x \; and \; F_n = (\rho v)_n \delta_x$$

$$D_e = \left(\frac{\Gamma}{\delta_x}\right)_e \delta_y \; and \; D_w = \left(\frac{\Gamma}{\delta_x}\right)_w \delta_y \tag{7}$$

$$D_s = \left(\frac{\Gamma}{\delta_y}\right)_s \delta_x \; and \; D_n = \left(\frac{\Gamma}{\delta_y}\right)_n \delta_x$$

F indicates the strength of convection and D indicates the strength of diffusion. It is important to note that F can assume positive and negative values depending on the direction of fluid flow.

With the assumption of the interfaces being midway of two control volume nodes, the central differencing scheme is achieved. However as mentioned before if the values of F are negative, coefficients of negative value can arise originating unrealistic results, therefore this scheme is only effective for extremely small values of δ_x and δ_y.

For primary derivation, the upwind scheme is a used remedy for the limitations encountered by the central differencing scheme. The upwind scheme proposes a better theory, by reformulating the convective terms – linked to F– yet leaving the diffusivity terms unchanged. The value of $\emptyset$ at the interface of the control volume is equal to the value of $\emptyset$ at the previous grid point upwind of the flow direction.

Thus,

$$\begin{aligned}
&\phi_e = \phi_P \;\; if \; F_e > 0 \;\; and \; \phi_e = \phi_E \;\; if \; F_e < 0 \\
&\phi_w = \phi_W \; if \; F_w > 0 \;\; and \; \phi_w = \phi_P \;\; if \; F_w < 0 \\
&\phi_s = \phi_P \;\; if \; F_s > 0 \;\; and \; \phi_s = \phi_S \;\; if \; F_s < 0 \\
&\phi_n = \phi_N \;\; if \; F_n > 0 \;\; and \; \phi_n = \phi_P \;\; if \; F_n < 0
\end{aligned} \tag{8}$$

Defining the new operator [[A,B]] to denote the greater of A and B, and with following upwind scheme assumptions, Equation (6) becomes:

$$a_P\phi_P = a_E\phi_E + a_W\phi_W + a_S\phi_S + a_N\phi_N \tag{9}$$

$$\begin{aligned}
a_E &= D_e + [[-\mathrm{F}_e, 0]] \\
a_W &= D_W + [[\mathrm{F}_W, 0]] \\
a_S &= D_S + [[-\mathrm{F}_S, 0]] \\
a_N &= D_N + [[\mathrm{F}_N, 0]] \\
a_P &= a_E + \; a_W + a_S + a_N
\end{aligned} \tag{10}$$

It is evident from Equation (10) that no negative coefficients will arise; therefore the solution will be physically realistic.

Numerical Schemes

Equation (4) can be solved for an exact solution if F and D are taken to be constants. As follows the $\phi = f(x)$ profile can be better understood for different Peclet numbers. The ratio of convective strengths by diffusive strengths is defined by the Peclet number

$$P = \frac{F}{D} \tag{11}$$

In a pure diffusion P=0 with $\phi = f(x)$ defined by a linear profile. For positive P the values of ϕ seem to be more influenced by the upstream values of ϕ, and becomes more accurate as higher positive values of P are approached throughout the domain . For negative P the picture is reversed. This proves that the upwind scheme becomes more accurate for higher values of $|P|$.

With the exact solution it is possible to obtain guidance as regarding the use of an appropriate profile $\phi = f(x)$. The primary derivation fails to give a satisfactory formulation, due to our profile not being linear therefore the assumption of the upwind scheme becomes valid only for where $|P|$ is high.

To compensate for all $|P|$ different schemes can be used, for instance as the Exponential Scheme (exact solution), Hybrid Scheme and the power law scheme the latter being used for this simulation.

The schemes are introduced into the discretized equations through a constant $\mathrm{A}(|\mathrm{P}|)$ added to the link coefficients of Equation (10).

Thus,

$$
\begin{aligned}
a_E &= D_e\mathrm{A}(|P|) + [[-\mathrm{F}_e, 0]] \\
a_W &= D_W\mathrm{A}(|P|) + [[\mathrm{F}_W, 0]] \\
a_S &= D_S\mathrm{A}(|P|) + [[-\mathrm{F}_S, 0]] \\
a_N &= D_N\mathrm{A}(|P|) + [[\mathrm{F}_N, 0]] \\
a_P &= a_E + a_W + a_S + a_N
\end{aligned}
\tag{12}
$$

Power Law Scheme

Although the Exponential Scheme provides the most accurate solution, this is a method which is not widely used due to the required computational power and consequently high costs for simulations.

To overcome the previous constraints the power law scheme is the best approximation to the exact curve for a wide range of Peclet numbers and it is a method that does not require excessive computational power.

The constant $\mathrm{A}(|\mathrm{P}|)$ added to the link coefficients presented on the previous numerical scheme section, for the power law scheme is given by

$$\mathrm{A}(|P|) = [[0, (1 - 0.1|P|)^5]] \tag{13}$$

Flow Field – Momentum Equation

The general formulation in the previous sections to solve the differential equation for ϕ is only valid in the presence of a given flow field. For this application it is not possible to guess a flow field as it can vary with flow and geometry conditions therefore the governing equations are used to calculate de flow field. The velocity components of the flow field are governed by the momentum Equation (14).

$$\frac{d\vec{V}}{dt} + \vec{V} \cdot \left(\nabla \vec{V}\right) = \frac{1}{\rho}\left(-\nabla \vec{P} + \nabla \cdot (\mu \nabla \vec{V})\right) + S \tag{14}$$

Considering the flow is a continuum, incompressible, steady state and with no body forces, Equation (14) reduces to Equation (15) in 2D coordinates.

$$\begin{aligned} &: \; u\frac{du}{dx} + v\frac{du}{dy} = \frac{1}{\rho}\left(-\frac{dP}{dx} + \frac{d}{dx}\left(\mu\frac{du}{dx}\right) + \frac{d}{dy}\left(\mu\frac{du}{dy}\right)\right) \\ y&: \; u\frac{dv}{dx} + v\frac{dv}{dy} = \frac{1}{\rho}\left(-\frac{dP}{dy} + \frac{d}{dx}\left(\mu\frac{dv}{dx}\right) + \frac{d}{dy}\left(\mu\frac{dv}{dy}\right)\right) \end{aligned} \tag{15}$$

Equation (15) is a distinct form of the steady state general Equation (4) for the general variable when $\phi = u$ or v is the 2D coordinate velocity, $\Gamma = \mu$ is the dynamic viscosity, and when the pressure gradient source term is added.

Solving the momentum equation is not a concern and could be done by iteration, the difficulty in calculating the velocity field lies in the unknown pressure field. The pressure filed is indirectly specified via the continuity equation, and when the correctly specified the resulting velocity filed satisfies the continuity equation.

Staggered Grid

It is important to recognize that the variables do not have to be calculated in the same grid, this is extremely effective to avoid limitations and difficulties of determining the pressure field.

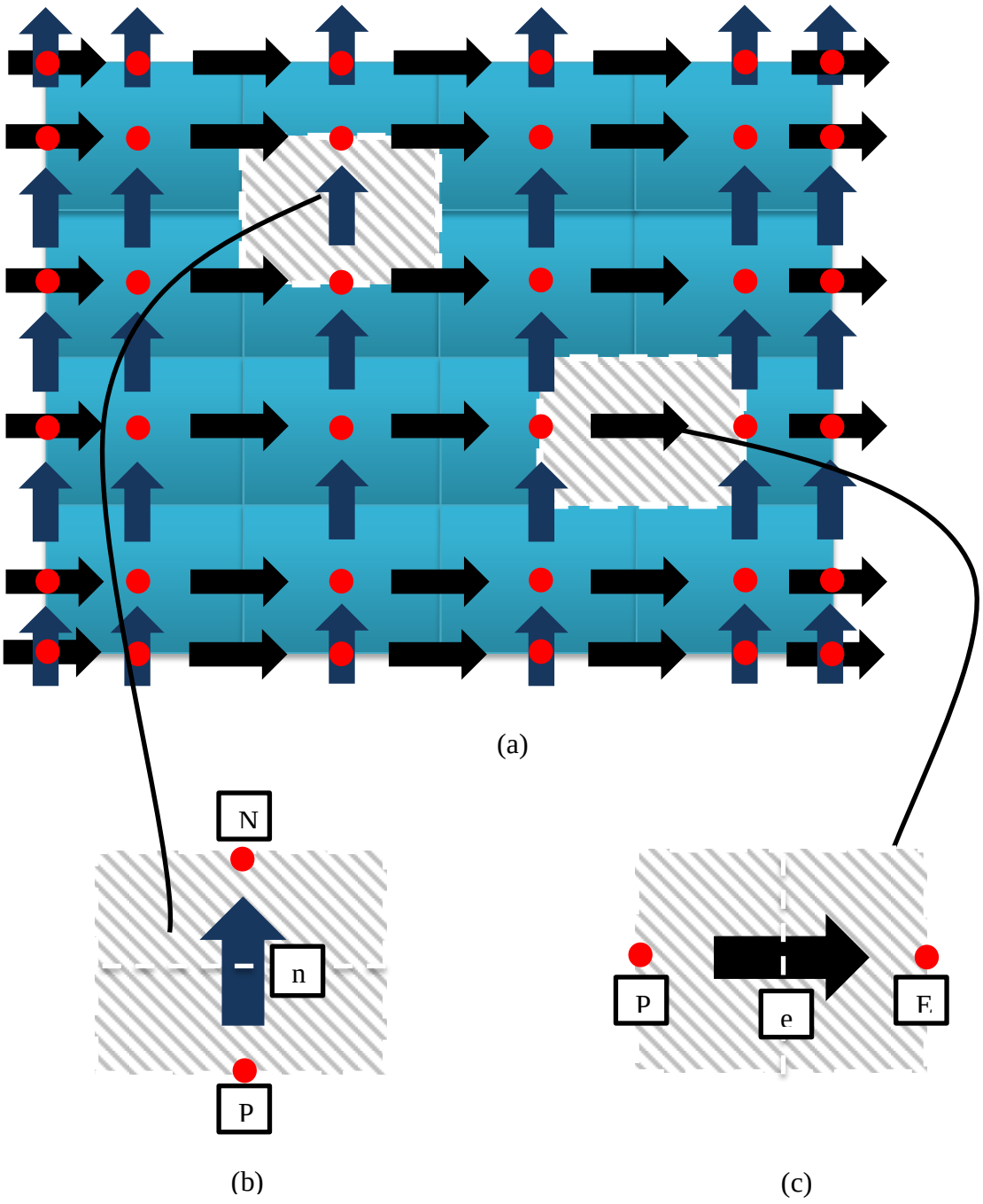

Figure 5. Control Volume Staggered Grid

For the Staggered Grid the velocity components are calculated at the faces of the control volumes of the other variables such as pressure. Assigning a different grid to the velocity components brings a significant benefit to the simulation.

Figure 5 (a) shows the representation of the staggered grid, and Figure 5 (b) and (c) shows two control volumes for the 2D coordinate velocity. The location of the u velocity is shown by the black arrow and the v velocity is shown by the green arrow, while the other variables (pressure, viscosity, density) are arranged at the red center node.

The first most important advantage of the staggered grid is that the discretized continuity equation will contain the differences of adjacent velocities and that prevents a wavy unrealistic velocity field. The second advantage is that the pressure difference between grids becomes the driving force for the velocity as shown by the velocity components control volumes of Figure 5 (b) and (c).

Discretized Momentum Equation

To obtain the discretization of the 2D coordinate momentum Equation (15), an identical formulation used in the previous section for the primary derivations of Equation (9) is used, subsequent to the substitution of the general variable $\emptyset$ for the velocity components, Γ for the absolute viscosity and addition of the source term for the pressure gradient field. Integration is performed over the velocity control volumes containing the pressure nodes at the interface presented in Figure 5 (b) and (c) determines:

$$\begin{aligned} x\text{: } a_e u_e &= \sum a_{nb} u_{nb} + (P_p - P_E)\delta_y \\ y\text{: } a_s v_s &= \sum a_{nb} v_{nb} + (P_p - P_N)\delta_x \end{aligned} \tag{16}$$

The discretized momentum Equation (16) can only be solved when a pressure field is predicted or given by experiment. If the correct pressure filed is not employed, the resulting velocity will not satisfy the continuity equation. The velocity profile obtained by a guessed pressure field is denoted by superscript * is defined by:

$$x:\ a_e {u_e}^* = \sum a_{nb} {u_{nb}}^* + ({P_p}^* - {P_E}^*)\delta_y$$
$$y:\ a_s {v_s}^* = \sum a_{nb} {v_{nb}}^* + ({P_p}^* - {P_N}^*)\delta_x \quad (17)$$

Pressure and Velocity Corrections

Considering the initial pressure field is guessed, it needs to be improved in order for the velocity flow field to become closer to satisfying the continuity equation. To correct the pressure a correction term need to be added to the initial guessed pressure, given by:

$$p = p^* + p' \quad (18)$$

where $p'$is the pressure correction term. Subsequently the velocity is affected by the change in pressure; in that sense it is also necessary to correct the velocities in a similar manner:

$$u = u^* + u'$$
$$v = v^* + v' \quad (19)$$

When subtracting Equation (17) from Equation (16), which is the guessed pressure field momentum equation minus the actual momentum equation provides the discretized correction momentum equation.

$$\begin{aligned} x\text{: } a_e {u_e}' &= \sum a_{nb} {u_{nb}}' + ({P_p}' - {P_E}')\delta_y \\ y\text{: } a_s {v_s}' &= \sum a_{nb} {v_{nb}}' + ({P_p}' - {P_N}')\delta_x \end{aligned} \tag{20}$$

When considering the Semi-Implicit Method the summation of the neighboring nodes is set to zero. The words Semi-Implicit means that by setting the summation term to zero in Equation (20), the neighboring velocity corrections do not influence one another in the correction momentum equation; however since the pressure term is maintained, there is an implicit influence of the pressure correction on the velocity and consequently causes a velocity correction at each node. Equations (20) reduce to:

$$\begin{aligned} x\text{: } a_e {u_e}' &= ({P_p}' - {P_E}')\delta_y \\ y\text{: } a_n {v_n}' &= ({P_p}' - {P_N}')\delta_x \end{aligned} \tag{21}$$

Substituting Equation (21) back into Equation (19) defines:

$$x:\ u_e = u_e{}^* + \frac{\delta_y}{a_e}(P_p{}' - P_E{}')$$
$$y: v_n = v_n{}^* + \frac{\delta_x}{a_n}(P_p{}' - P_N{}') \tag{22}$$

Now, to obtain the pressure correction equation the continuity equation is used. The 2D continuity equation is given by:

$$\frac{d}{dx}(\rho u) + \frac{d}{dy}(\rho v) = 0 \tag{23}$$

To obtain the discretized equation, the same procedure as the general variable is used, where integration of the continuity Equation (23) over the control volume of Figure 4 is performed

Thus,

$$[(\rho u)_e - (\rho u)_w]\delta_y + [(\rho v)_e - (\rho v)_w]\delta_x = 0 \tag{24}$$

If now the velocity correction terms from Equation (22) are substituted into Equation (24):

$$a_P P_P{}' = a_E P_E{}' + a_W P_W{}' + a_N P_N{}' + a_S P_S{}' \tag{25}$$

$$a_E = \rho_e \frac{\delta_y{}^2}{a_e},$$

$$a_W = \rho_w \frac{\delta_y{}^2}{a_e}$$

$$a_N = \rho_n \frac{\delta_x{}^2}{a_n} \tag{26}$$

$$a_S = \rho_s \frac{\delta_x{}^2}{a_n}$$

$$a_P = a_E + a_W + a_N + a_S$$

The SIMPLE Algorithm

The procedure that has been given to calculate the flow field, by guessing a pressure filed and correcting the pressure and the velocities is given the name of SIMPLE algorithm, which stands for semi implicit method for the pressure-linked equations.

The sequence of operation for the SIMPLE algorithm is:

1.) Guess the pressure field P^*.

2.) Solve the momentum Equation (17), to obtain $u^*,\ v^*$.

3.) Solve the pressure correction equation P'.

4.) Calculate P from Equation (18) adding $P'\ to\ P^*$.

5.) Calculate $u\ and\ v$ using the velocity correction Equation (22).

6.) Solve the other discretized Equations for T.

7.) Make the corrected pressure P the new guessed pressure P^*, and start from step 2. Repeat until a converged solution is obtained.

Solution of the System of Equations

To model the flow and heat transfer problems of this simulation either ordinary (ODE) or partial differential (PDE) equations are used, and in order to solve these equations they need to be converted to a system of linear algebraic equations, as shown in a previous section trough discretization.

To solve the system of equations, various methods are available for usage, choosing the correct method is extremely important to guarantee optimal required computational power. The methods are separated into iterative methods and direct methods or a blend of the two methods. The methods used for the scope of this simulation will be presented.

Gauss-Seidel

$$\emptyset_i^{(K+1)} = \frac{1}{a_{ii}}\left[b_i - \sum_{j=1}^{i-1} a_{ij}\emptyset_j^{(K+1)} - \sum_{j=i+1}^{N} a_{ij}\emptyset_j^{(K)}\right] \tag{27}$$

$$\Delta\emptyset_i = \Delta\emptyset_i{}^{(K+1)} - \Delta\emptyset_i{}^{(K)}$$

The Gauss-Seidel method is an implicit iteration method to solve a system of equations.

In Equation (27) the index i is the number of the equation, ∅ is the general variable, K is the number of iterations, a are the coefficients of the equations and b is the source term of each equation.

As a starting point initial values are guessed for all variables ∅ and the value of the first variable is calculated through the first equation. Subsequently, the calculated value for the first variable $\emptyset_j^{(K+1)}$ is used on the subsequent equation $a_{ij}\emptyset_j^{(K)}$ until a converged solution that satisfies all the equations is attained$\Delta\emptyset_i = 0$. The major disadvantage of this method is that convergence is slow, especially when a fine grid is being used.

Tridiagonal Matrix

The tridiagonal matrix is a direct method, which solves a system of equations by the standard Gauss Elimination method. The 1-D discretized equations display the tridiagonal pattern, due to each equation having only the a_w, a_P, a_E coefficients.

Thus, the matrix for the grid of length n becomes:

$$\begin{pmatrix} -a_{W1} & a_{P1} & -a_{E1} & 0 & 0 & 0 \\ 0 & -a_{W2} & a_{P2} & -a_{E2} & 0 & 0 \\ 0 & 0 & \ddots & \ddots & \ddots & 0 \\ 0 & 0 & 0 & -a_{Wn} & a_{Pn} & -a_{En} \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \\ \vdots \\ u_n \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{pmatrix} \qquad (28)$$

The main objective of the tridiagonal matrix is to exploit the pattern by storing only the non-zero elements, which reduces significantly the memory required.

To apply the algorithm, each of the diagonals has to be applied to vectors

$$A = [-a_{W1}, -a_{W2}, \dots, -a_{Wn}]$$
$$B = [a_{P1}, a_{P2}, \dots, a_{Pn}]$$
$$C = [-a_{E1}, -a_{E2}, \dots, -a_{En}] \quad (29)$$
$$D = [b_1, b_2, \dots, b_n]$$

Subsequently the equation is solved through the direct Gauss-Elimination method embedded in the algorithm. It is important to note that the two dimensional grid displays a Penta diagonal matrix. Solving the entire grid by a direct method would not be very effective, requiring excessive memory.

Line by Line Method

A combination of the TDMA direct method and the iterative Gauss-Seidel method is the line by line method. First a grid line in the x or y direction shall be chosen, afterwards the grids along the direction of the chosen line are solved through the TDMA method, while the grid points in the other direction are solved through Gauss-Seidel iteration method. After sweeping through all the grid points, the method can be repeated by switching the directions and sweeping again through the grid points. This helps improve convergence time. The main advantage of this method is that the boundary conditions at the endings are promptly inserted to the grid points also improving convergence time.

When the y direction is chosen to be solved through TDMA, it is called implicit in the y direction, this is because during the iteration calculation the terms the y line and to the east are at the K iteration while the grid points to the west at ae the K+1 iteration. The scheme is shown by Figure 6 for implicit y direction and Figure 7 for implicit in x direction.

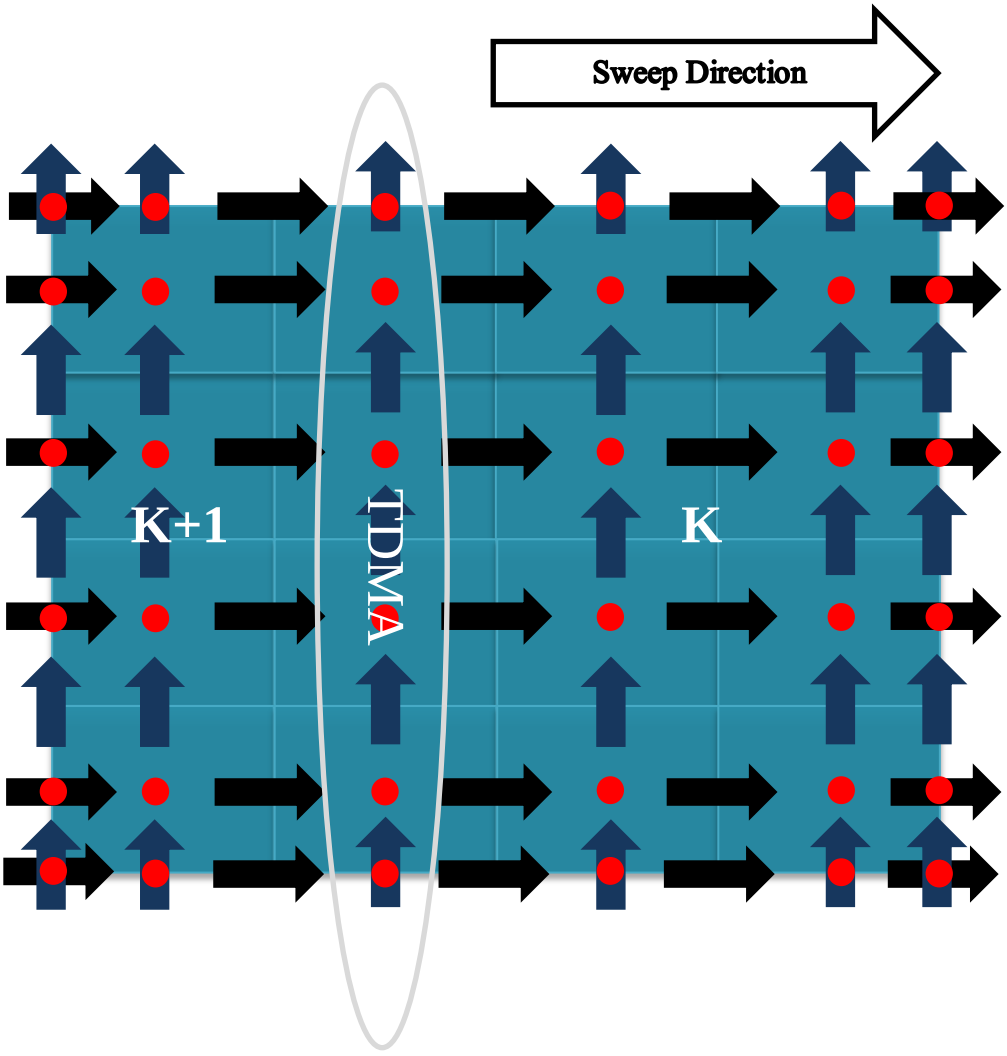

Figure 6. Line by Line Method Implicit y

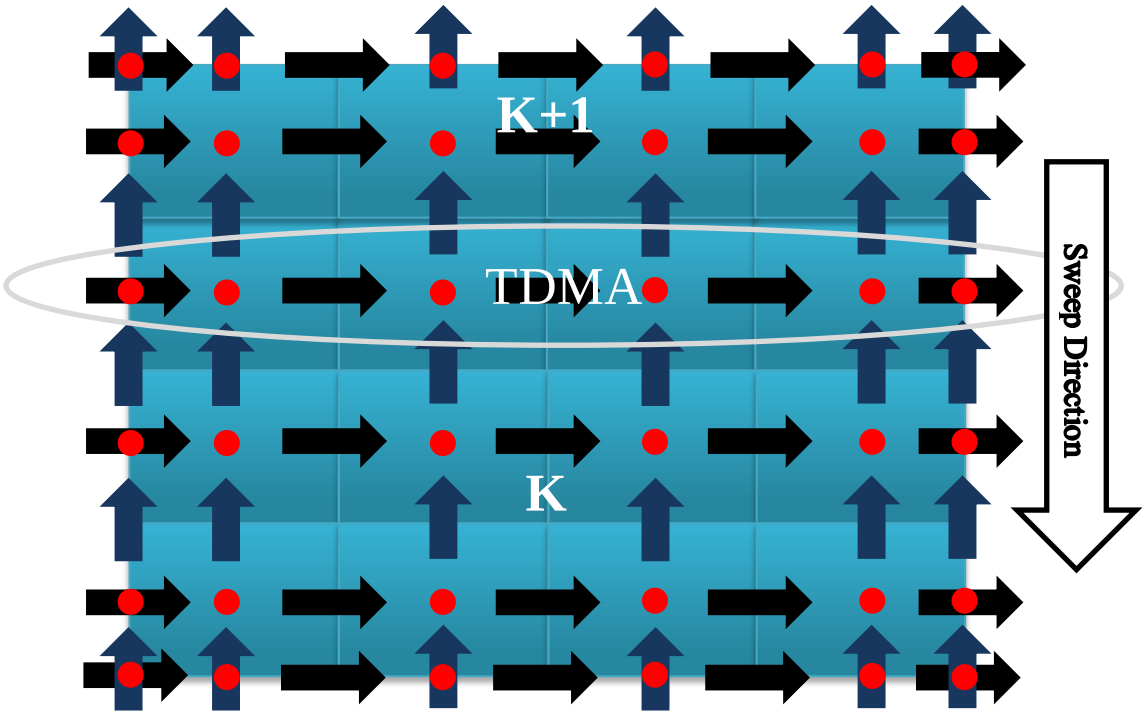

Figure 7. Line by Line Method Implicit x

Overrelaxation and Underrelaxation

During iteration solution -which is used to handle all the nonlinearities- it is often desirable to speed up or slow down the changes from iteration to iteration. Overrelaxation is when the convergence is accelerated and the underrelaxation is when convergence is slowed down. The underrelaxed or overrelaxed general variable discretized equation (9) takes the following form:

$$\frac{a_P}{\alpha}\phi_P = a_E\phi_E + a_W\phi_W + a_S\phi_S + a_N\phi_N + (1-\alpha)\frac{a_P}{\alpha}\phi_P^* \qquad (30)$$

where the term ϕ_P^* from the previous iteration is introduced along with the relaxation factor α. It is important to note that when the relaxation is between 0 and 1 the effect is

underrelaxation. Underrelaxation is only recommended in cases when the non-linearity of the discretized equation is high. There is no general rule for the correct value of the relaxation factor, as it is usually defined by experimenting different values and simulating. The overrelaxation factor that commonly used is between 1 and 1.5.

For this simulation the relaxation method is applied to all discretized equations from the previous section.

Assumptions

The assumptions for this simulation were, flow is a 2D continuum, laminar, steady-state, incompressible, the fluid is considered to be Newtonian and the walls are thin with very high thermal conductivity.

Convergence

The used convergence criterion is the residual method, given by:

$$R_u = \sum \frac{\left| a_P u_P - \sum a_{nb} u_{nb} - (P_W - P_E)\delta_y \right|}{\left| a_P u_P \right|} \leq 10^6$$

$$R_v = \sum \frac{\left| a_P v_P - \sum a_{nb} v_{nb} - (P_S - P_N)\delta_x \right|}{\left| a_P v_P \right|} \leq 10^6$$

$$R_P = \sum \frac{\left| (\rho_w u_w - \rho_e u_e)\delta_y + (\rho_s u_s - \rho_n u_n)\delta_x \right|}{\left| \rho u_{max} L_y \right|} \leq 10^6 \quad (31)$$

$$R_T = \sum \frac{\left| a_P T_P - \sum a_{nb} T_{nb} \right|}{\left| a_P T_P \right|} \leq 10^7$$

Grid Dependence

Different mesh sizes were employed for the conditions of Re=150, h/Dh=0.25 and S/dh= 2.0 the grid size was changed from 25x250 to 28x275 to 30x300 changed by less than 0.05%.

Code Validation

To verify the code of the numerical simulation, the analytical solution for fully developed flow in a smooth channel was used with a constant heat flux boundary condition of 8.235 according to Han (2012). Using a grid size of 30x300 the Nusselt number archived was of 8.232 which is a 0.03% difference from the analytical solution.

A pressure field was guessed of 82 (Pa) for a Re=150, S/Dh=2.0 and h/Dh=0.35 and the final pressure field achieved was only different by 0.073% by the initial zero pressure field, but it is important to note that convergence time was reduced.

CHAPTER IV

RESULTS AND DISCUSSION

In this section performance results are presented for the numerical simulation, followed by a discussion. In this study, it was assumed that the heat recovery system is an air-to-air compact heat exchanger (HRV), with two ducts of equal dimensions, properties and flow conditions. Figure 8. shows the schematic and the fluid properties along with inlet conditions are presented in Table 1:

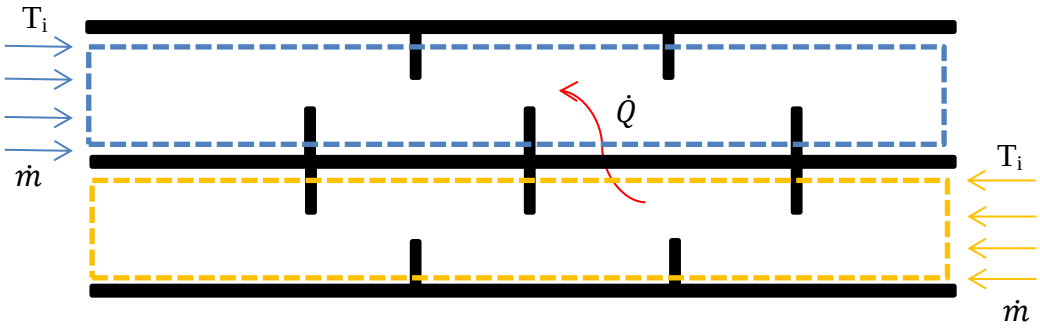

Figure 8. Schematic of Air-to-Air Compact Heat Exchanger

$\rho\ (density)$	$1.177\ \left(\frac{Kg}{m^3}\right)$
μ (Viscosity)	$1.846\ x\ 10^{-5}\ (Pa\ s)$
k (Thermal Conductivity)	$0.02624\ \left(\frac{W}{m^2K}\right)$
c_p (Specific Heat)	$1004.8\ \left(\frac{J}{Kg\ K}\right)$
R_e (Reynolds Number)	$150, 300, 450$
P_r (Prandtl Number)	0.707
T_{inC} (Inlet temperature of the cold channel)	$298\ K$
T_{inH}(Inlet temperature of the hot channel)	$313\ K$

Table 1 Fluid Properties and Inlet Temperature Conditions

Table 2 presents the geometrical conditions of the heat recovery system along with the size of meshing for both ducts.

D_h (Hydraulic Diameter)	$0.004\ m$
$L\ (length)$	$0.280\ m$
EntL (Straight entry length)	$0.045\ m$
ExiL (Straight exit length)	$0.045\ m$
h (Height of the Baffle)	$0.0010\ m, 0.0014\ m, 0.0018\ \ m$
Th (Thickness of the Baffle)	$0.001\ m$
Pit (Pitch of the baffle plate)	$0.002\ m, 0.006\ m, 0.010\ m$
Nx (Number of control volumes x)	300
Ny (Number of control volumes y)	30

Table 2 Geometry and Meshing Size

To determine the effects of baffle height, baffle pitch and Reynolds number on heat transfer enhancement while maintaining a relatively low increase in pressure drop, it was necessary to keep two of the latter variables constant, while varying the other and analyze from the results the impact of each.

Effect of Baffle Height

For this section, the simulation was carried out using a constant Reynolds Number of 150 and a baffle spacing ratio of S/Dh=2.0 while varying the baffle height h/Dh= 0.25-0.35-0.45.

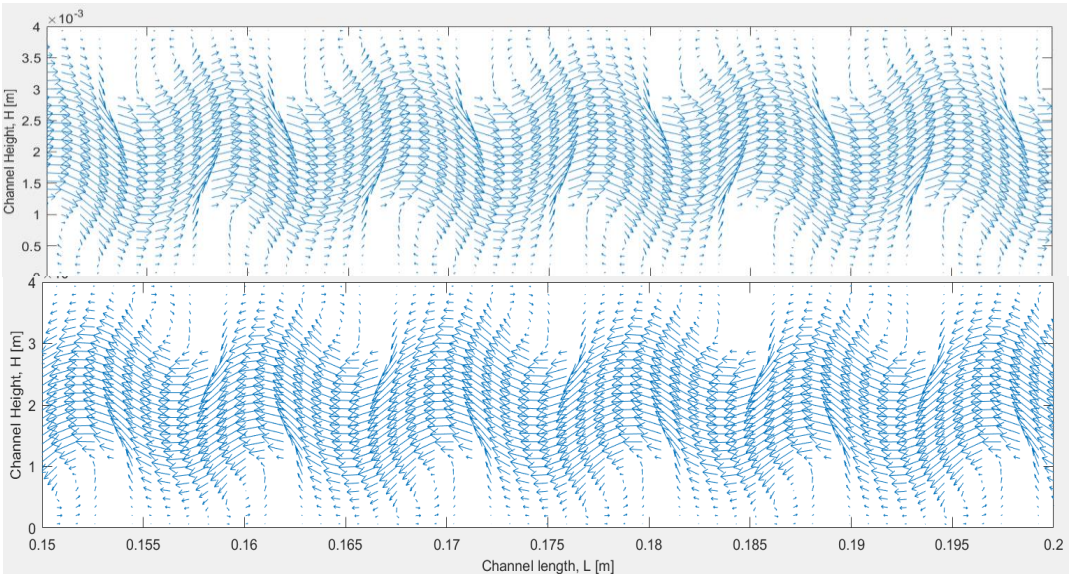

Figure 9. Velocity Vector Plots h/D_h=0.25

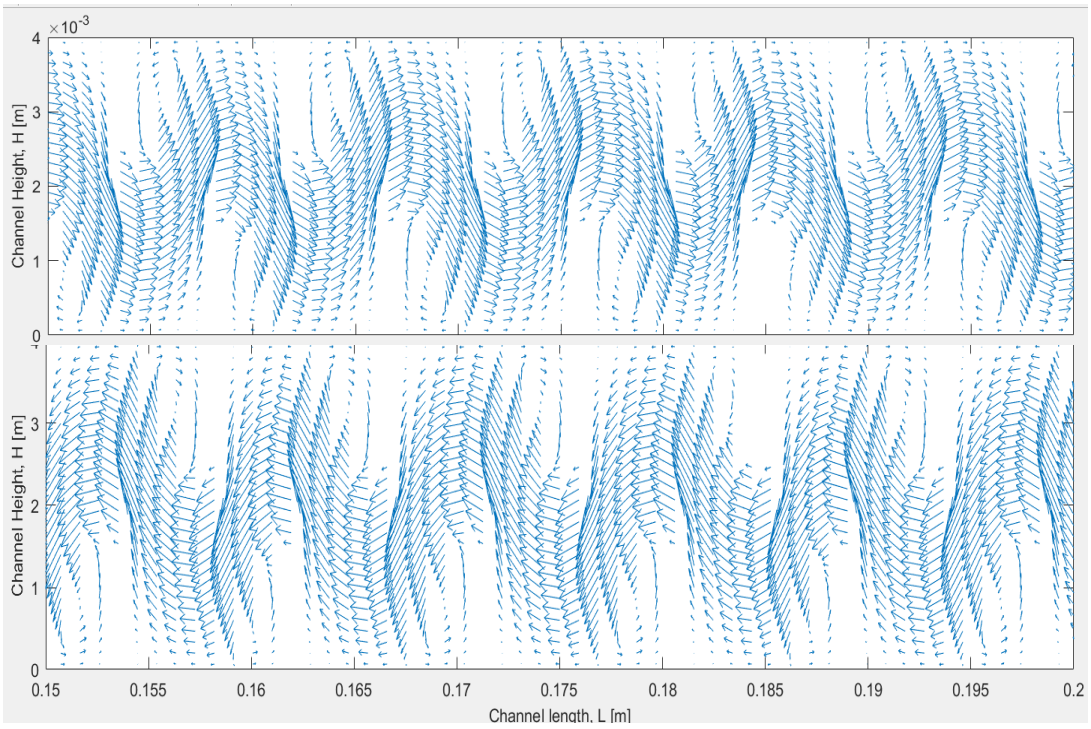

Figure 10. Vector Plots h/D_h=0.35

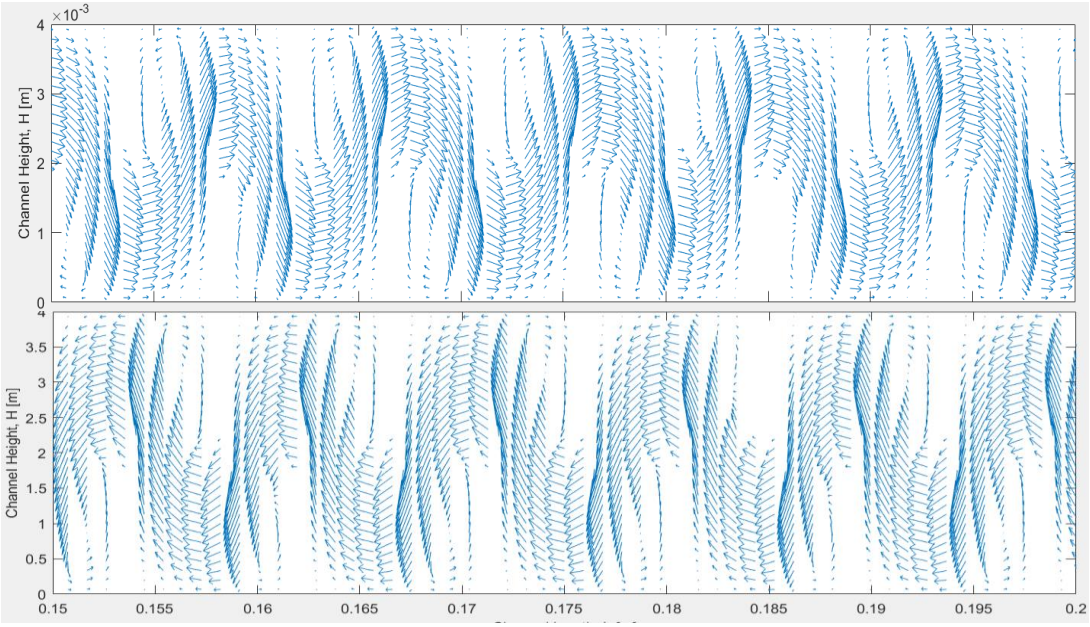

Figure 11. Velocity Vector Plots h/D_h=0.45

Figures 9 to 11 illustrate the effect of baffle height on the hydrodynamic flow profile for the 0.15m to 0.2m section of the ducts. The height of the baffles results in substantial change of the flow profile, creating recirculation zones downstream and also smaller recirculating zones upstream the baffles. Figure 9 – The smallest baffle height– displays only a slight bulk flow deflection, but in turn produces elongating circulating eddies that occupy 67% of the baffle spacing with consequently a longer reattachment length of the flow which is in the order of 4.7mm or 4.7 times the height. Figure 10 – The intermediate baffle height – exhibits more intensive bulk flow deflection, producing shorter downstream baffle circulating eddies that occupy 53% of the baffle spacing with a reattachment length of the flow in the order of 3.7 mm or 2.6 times the height.

Figure 11- The greater baffle height – follows an even more intensive flow deflection trend compared to the other baffle heights, generating the shortest downstream baffle recirculating eddies that occupy 43% of the baffle spacing and reattachment length in the order of 3mm or 1.6 times the baffle height. It is also important to note that as the baffle height is lengthened, velocity is increased in the "core" of the flow and the reattachment length is shortened. The increased flow deflection cause greater impingement on the walls and promotes better mixing of the "core" with the regions near the walls.

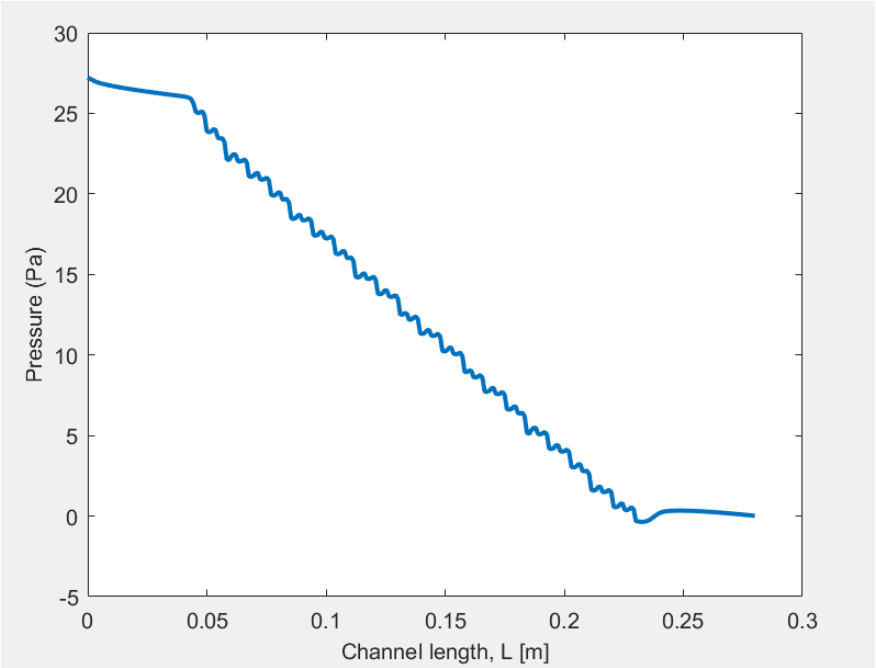

Figure 12. Pressure Drop h/D_h=0.25

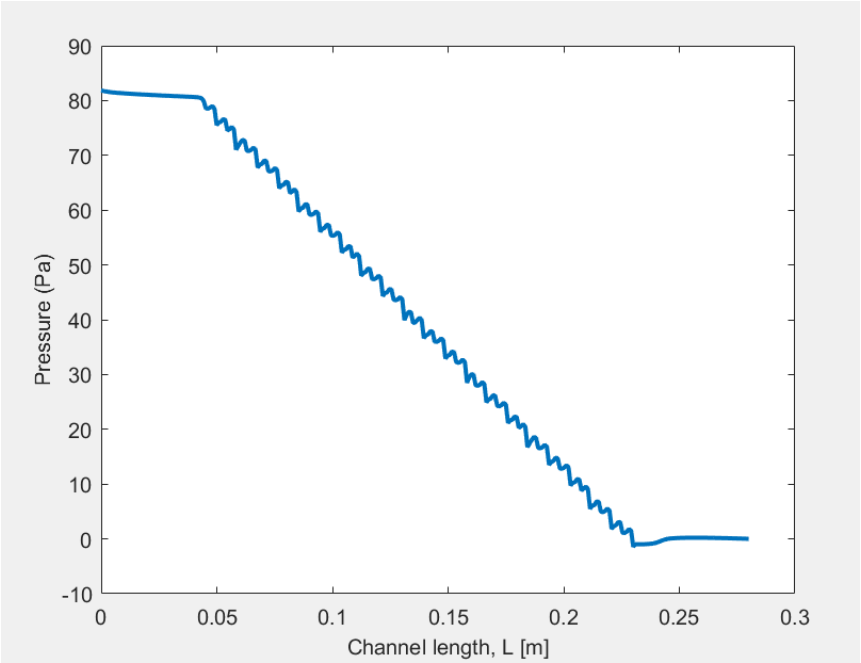

Figure 13. Pressure Drop h/D_h=0.35

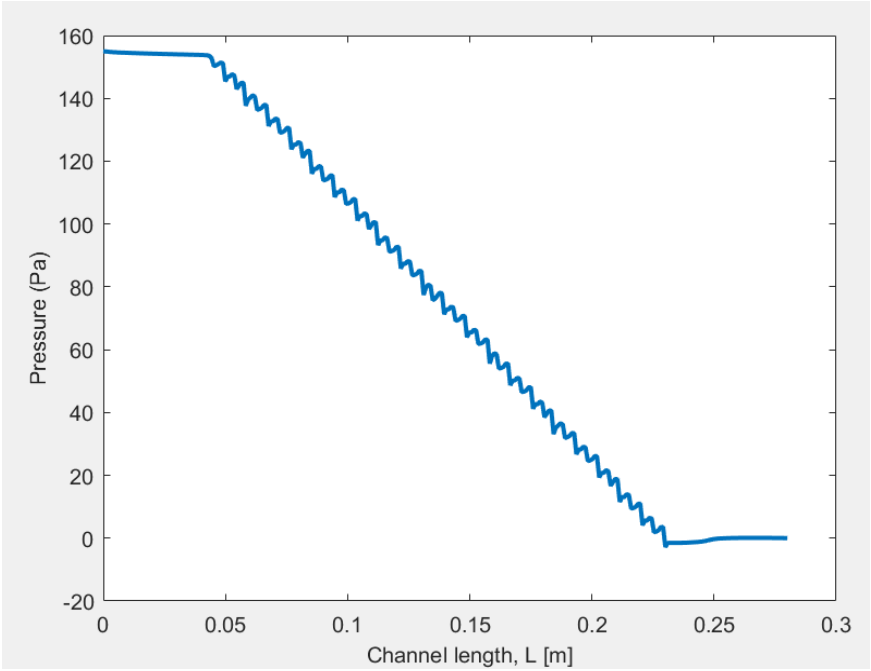

Figure 14. Pressure Drop h/D_h=0.45

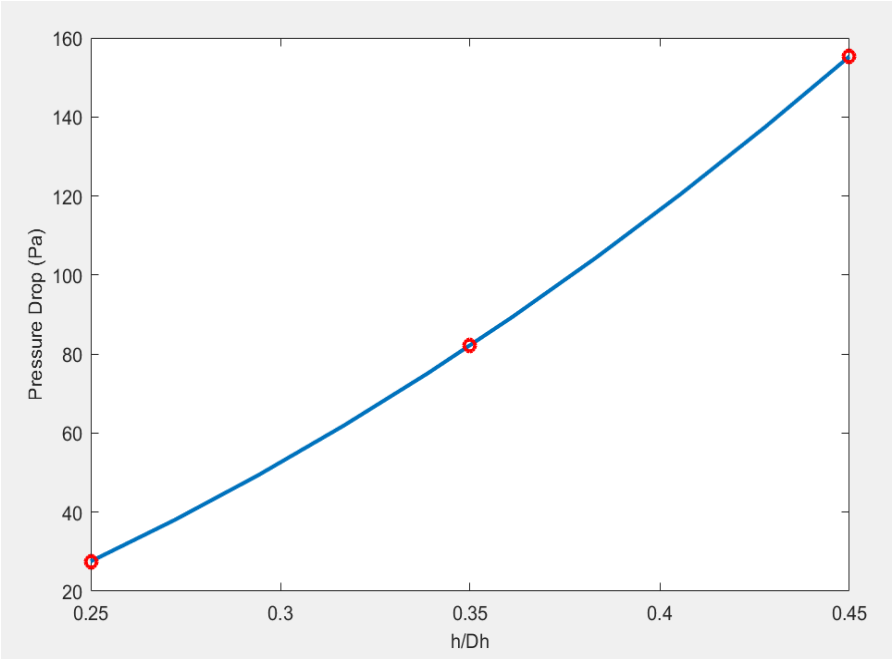

Figure 15. Pressure Drop x h/D_h

Figures 12 to 15 illustrate the effect of baffle height on the total pressure drop. As shown in the previous section, when the height of the baffle is increased, flow is increasingly deflected causing impingement on the walls. Also, as the baffle height is increased, the cross sectional area decreases, promoting greater velocity in the "core" of the flow increasing the pressure drop. The pressure drops for h/Dh=0.25, 0.35, 0.45 were 27.54 (Pa), 82.15 (Pa), and155.31 (Pa) respectively. Varying h/Dh=0.25 to 0.35 increased the pressure drop by 54.6 (Pa), while varying h/Dh=0.35 to 0.45 increased the pressure drop by 73.16 (Pa). Figure 15 displays this trend as the rates of change increase as baffle height h/Dh is lengthened.

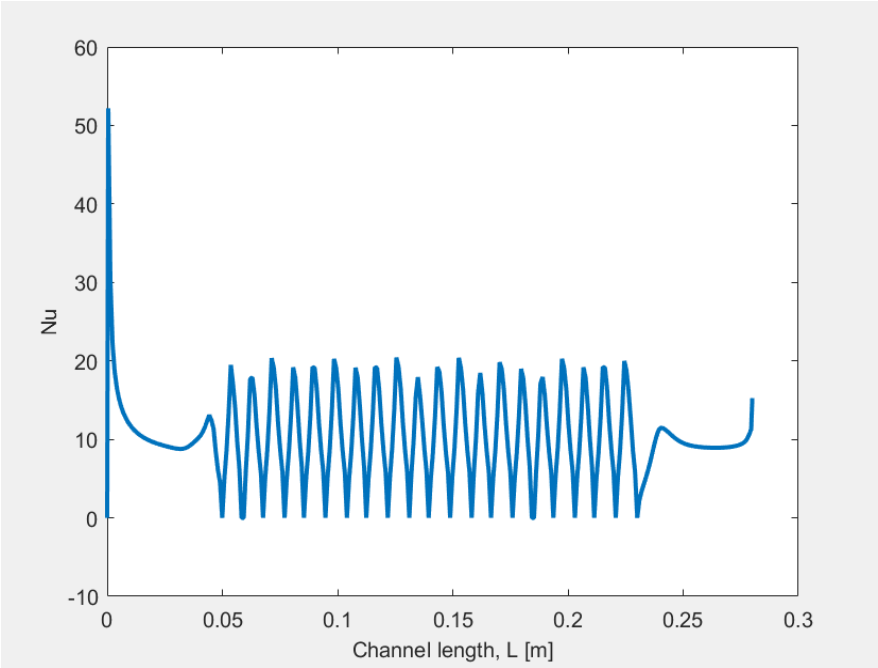

Figure 16. Nusselt Number h/D_h=0.25

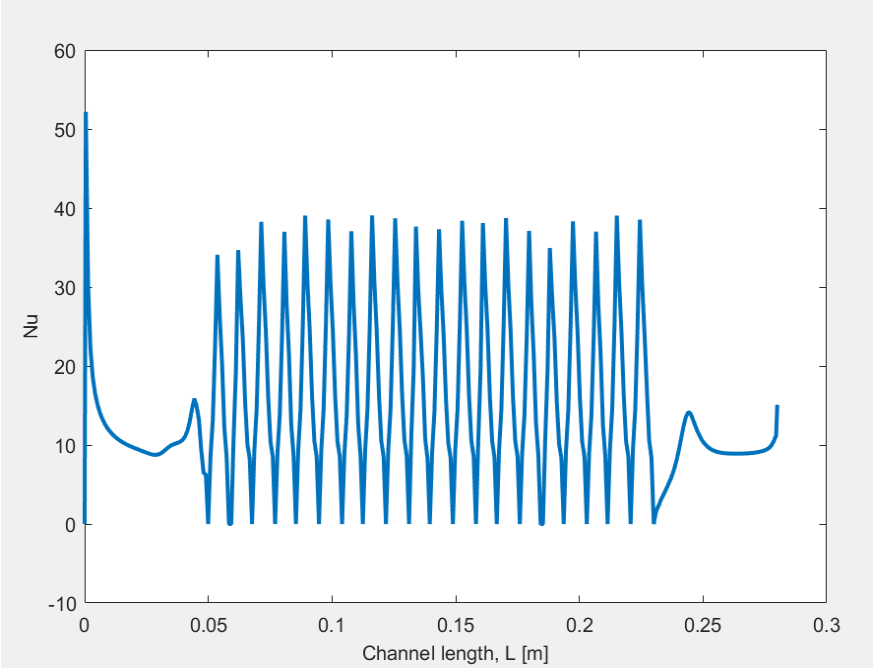

Figure 17. Nusselt Number h/D_h=0.35

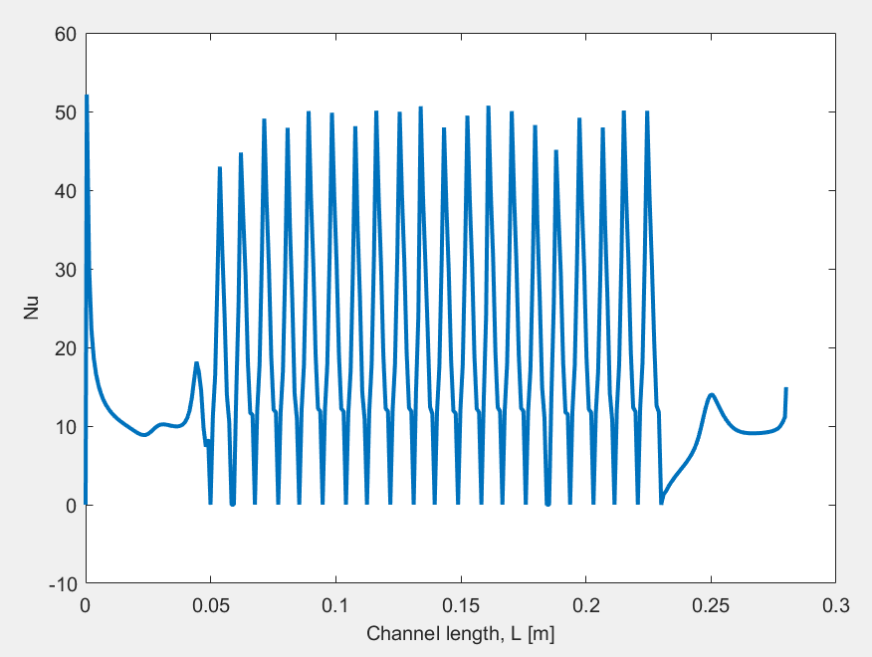

Figure 18. Nusselt Number h/D_h=0.45

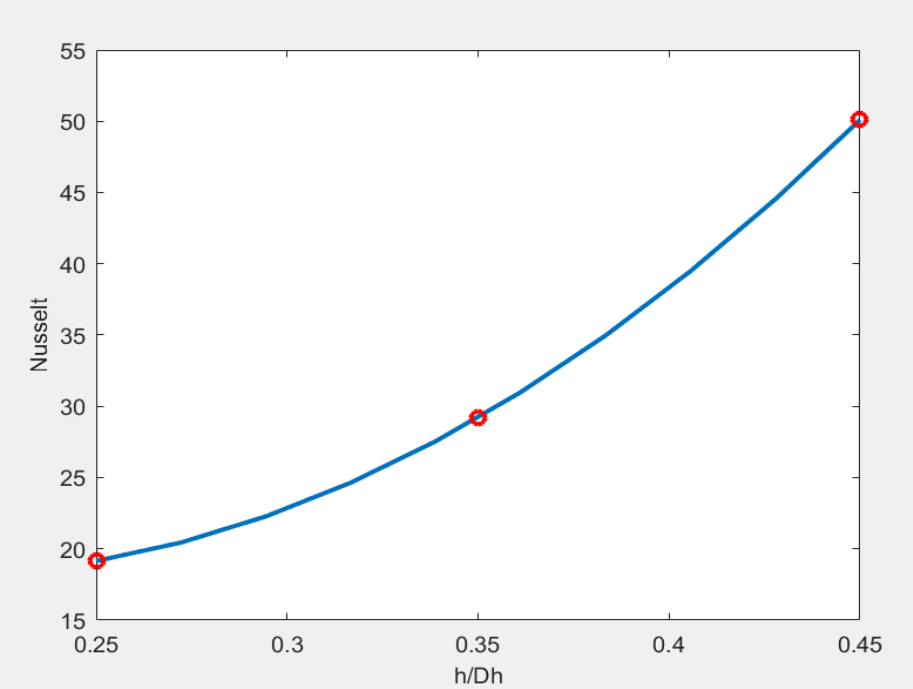

Figure 19. Nusselt x h/D_h

Figures 16 to 19 illustrate the effect of baffle height on the local Nusselt number, which determines the convective heat transfer coefficient. The baffles interrupt not only the flow, but it also the thermal boundary layer. The Nusselt number reaches peak value after the recirculation eddies where flow is reattached. The peak Nusselt numbers for h/Dh=0.25, 0.35, 0.45 were of 19, 29, and 50 respectively. Varying h/Dh=0.25 to 0.35 increased the Nusselt number by 10, while varying h/Dh=0.35 to 0.45 increased the Nusselt number by 21. Figure 19 displays this trend as the Nusselt number rates of change increase as baffle height h/Dh is lengthened. The increasing rates of the Nusselt number are much lower compared to the pressure drop rates, and as h/Dh approaches 0.45 or higher, the pressure drop increases at greater rates than the Nusselt number. When this condition is reached the increased pressure drop does not justify such small gain of heat transfer enhancement.

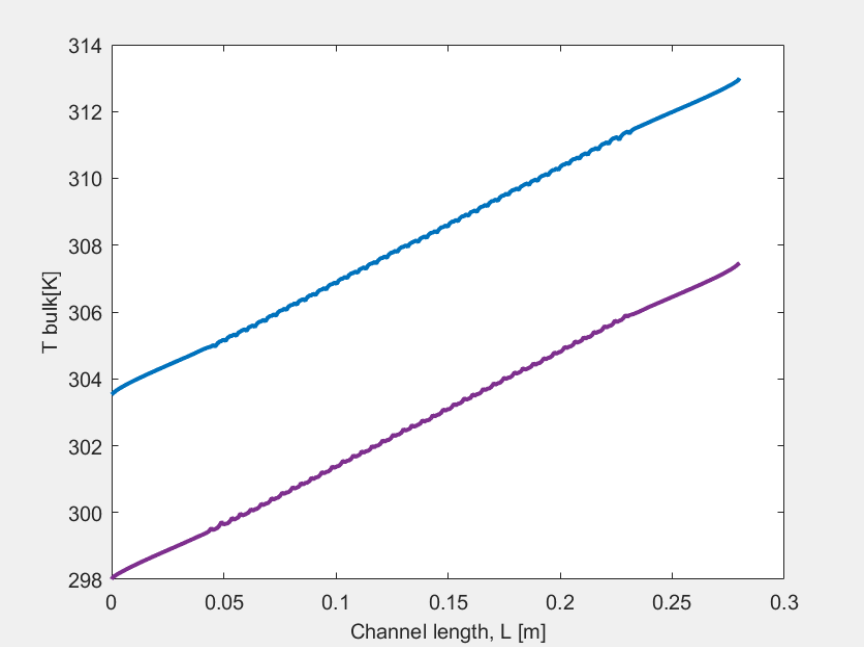

Figure 20. Bulk Temperature h/D_h=0.25

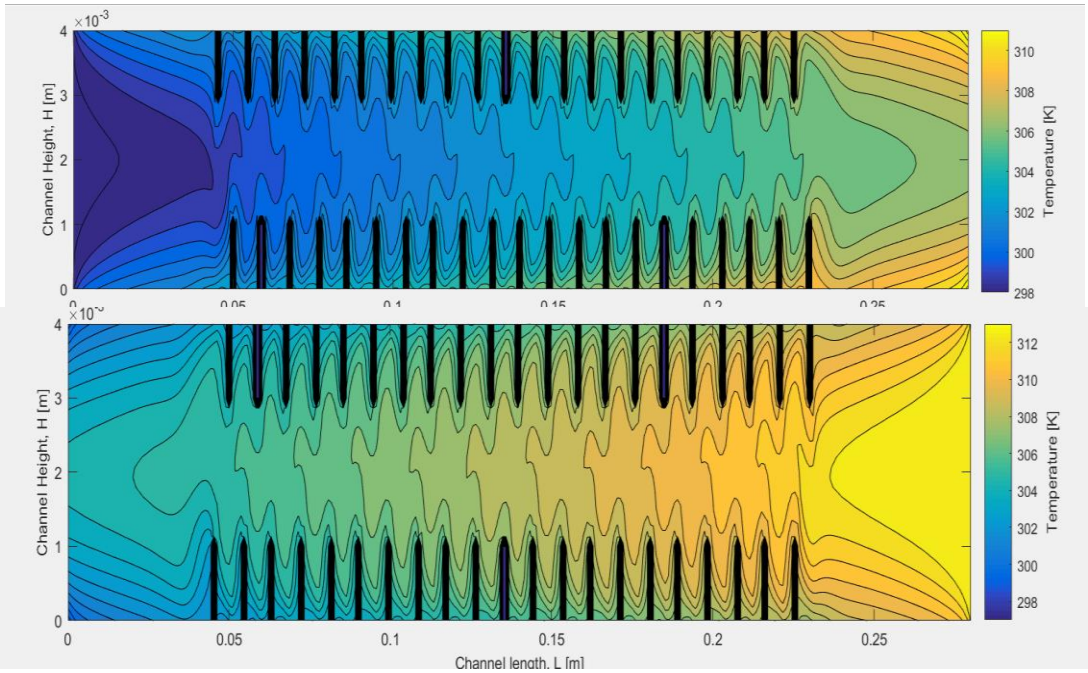

Figure 21. Temperature Profile of Channels h/D_h=0.25

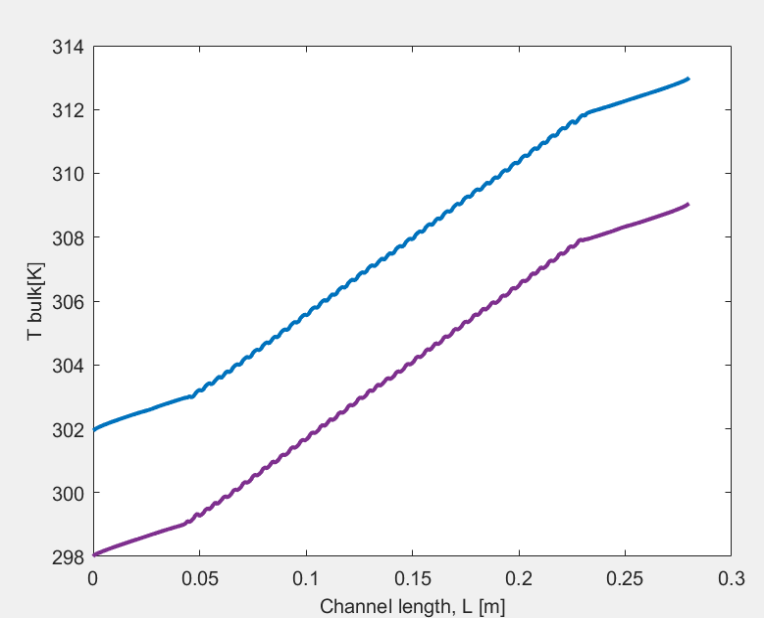

Figure 22. Bulk Temperature h/D_h=0.35

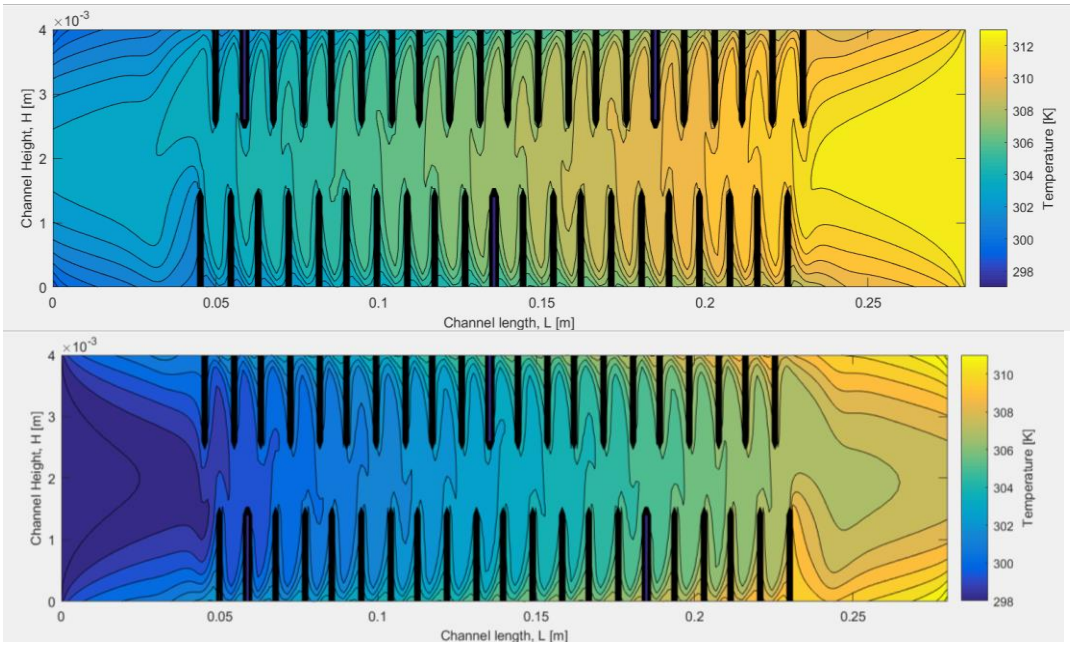

Figure 23. Temperature Profile of Channels h/D_h=0.35

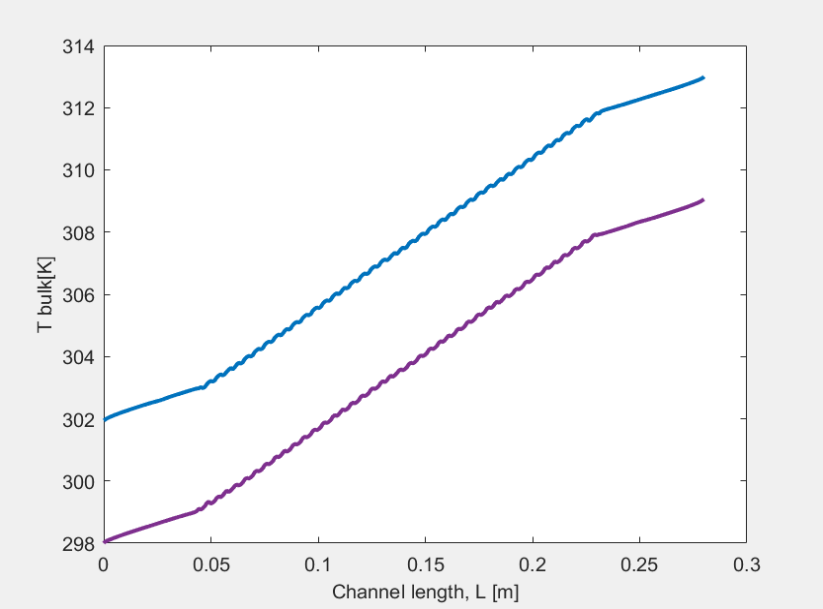

Figure 24. Bulk Temperature h/D_h=0.45

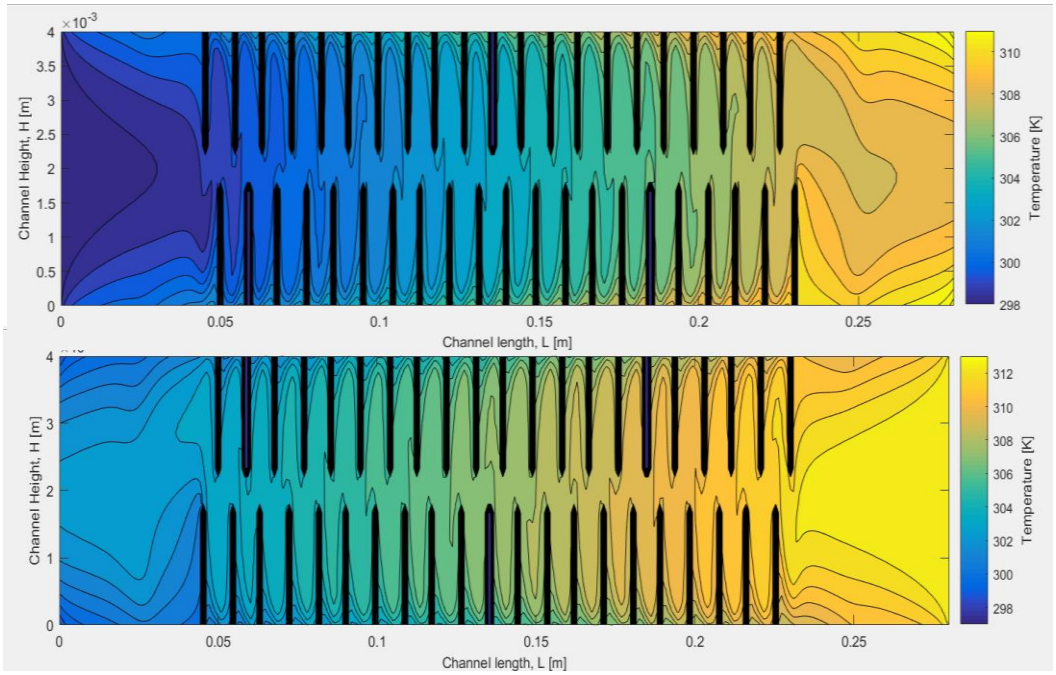

Figure 25. Temperature Profile of Channels h/D_h=0.45

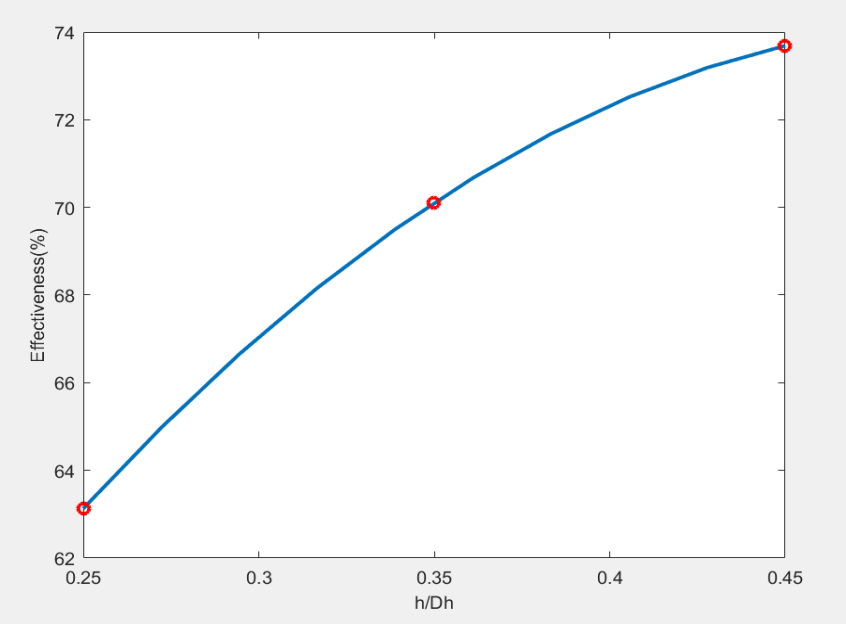

Figure 26. Heat Transfer Effectiveness x h/Dh

Figures 20 to 26 illustrate the effect of baffle height on temperature and heat transfer effectiveness. For h/Dh=0.25 the bulk temperature shift between the inlet and outlet of the two ducts is $\Delta T = 9.5$ K. The slope is similar to of a straight duct; this can be noticed by figure 20, as there is only a slight change of slope relative to the entrance length, where no baffles are present. This behavior, mainly occurs because there is only a slight deflection of the flow, that hardly promotes interruption of the thermal boundary layer. For h/Dh=0.35 the bulk temperature shift is $\Delta T = 10.5$ K, and for h/Dh=0.45 was $\Delta T = 11$ K. For the previous two baffle heights the bulk temperature slope becomes more apparent, as the thermal boundary layer is further interrupted.

Figure 26 presents the heat transfer effectiveness, increasing h/Dh=0.25 to 0.35 increases significantly the effectiveness, but the rates of change start to decay for any further increment in h/Dh, and tends to approach a constant effectiveness after h/Dh=0.45.

The temperature contour plots have a strong link with the velocity vector plots presented before, as the recirculating eddies downstream of the baffle act as an insulation zone. This becomes clear as the contour plot displays a zero temperature gradient tendency at those regions. It was proven that as the baffle height is increased, the length of the recirculating eddies decrease; therefore the insulation effect occurs in a shorter region. Since this simulation is performed in a counter-flow arrangement, it is important to note that the flow profiles of each duct are mirrored, and accordingly while one of the ducts is in the insulation recirculating region of lowest heat transfer coefficient, the other is most likely at a region where flow is reattached in an area of highest heat transfer coefficient.

Recapitulating to the velocity vector plots, the length of the insulation region for h/Dh=0.25, 0.35, 0.45 is of 4.7mm, 3.7 mm, 3mm respectively, while the space between the baffles is only 7mm.

After analyzing the results for the three different heights it is clear the effect of baffle height on the various parameters presented. Now it is possible to designate the most effective height, that will provide heat transfer enhancement with a tolerable pressure drop, and it all points to h/Dh=0.35.

Effect of Baffle Pitch

To determine the effects of baffle pitch, the simulation was carried out using a constant Reynolds Number of 150 and a baffle height ratio of h/Dh=0.35, which was determined as optimum when comparing the three heights in the previous section. While keeping these variables constant the baffle pitch was varied S/Dh= 0.5, 1.5, 2.5.

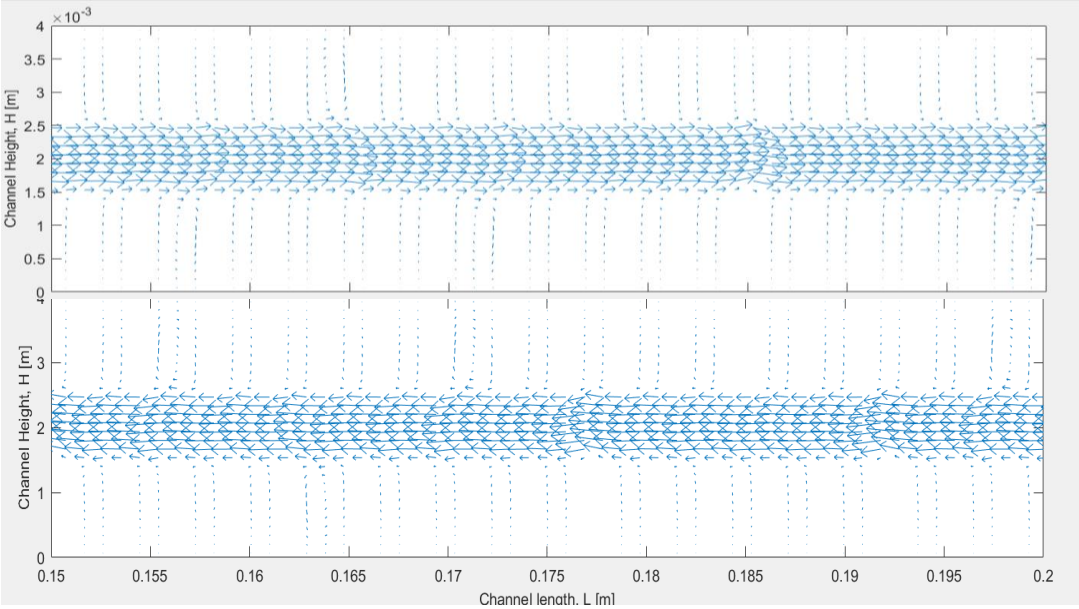

Figure 27. Velocity Vector Plots S/D_h=0.50

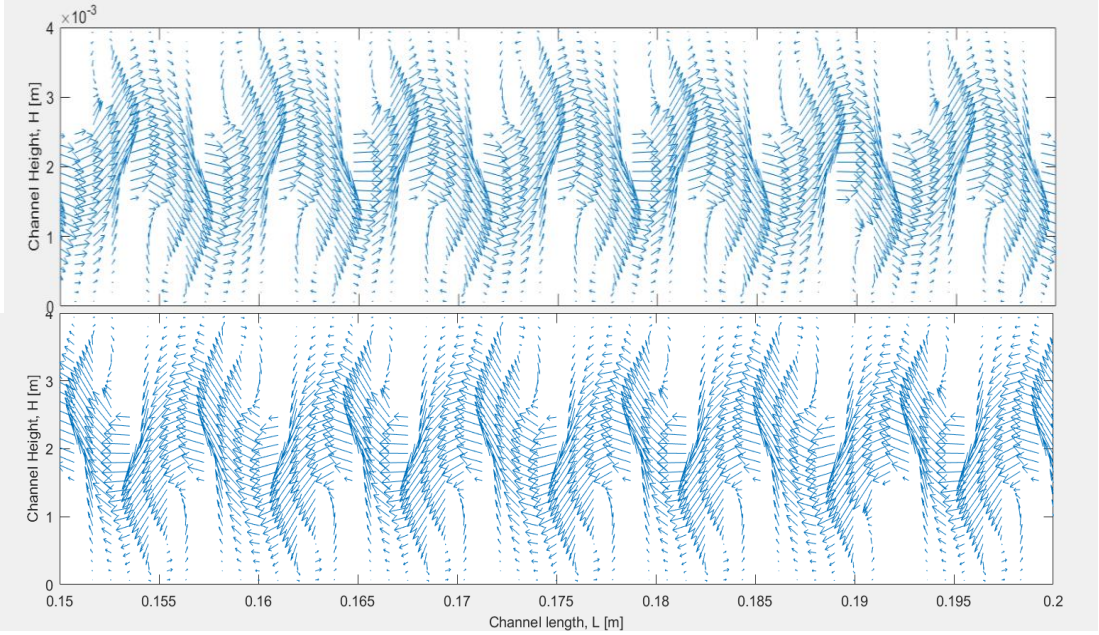

Figure 28. Pressure Drop S/D_h=1.50

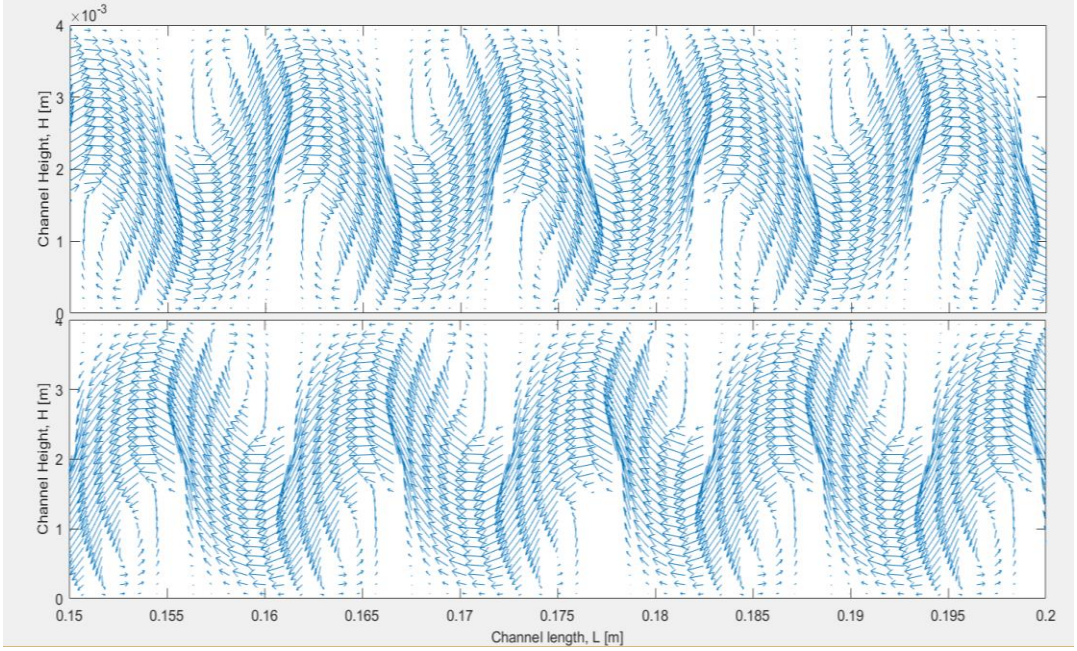

Figure 29. Pressure Drop S/D_h=2.50

Figures 27 to 29 illustrate the effect of baffle pitch on the hydrodynamic flow profile for the 0.15m to 0.2m section of the ducts. The pitch, similarly to the height of the baffles also contribute in substantial change of the flow profile, creating recirculation zones downstream and also smaller recirculating zones upstream the baffles. Figure 27 – The smallest pitch S/Dh=0.50- does not promote any deflection of the flow; as the recirculating eddies occupy 100% of the baffle spacing. Due to this behavior, the flow tends to approach a parallel plate solution, but on a channel with a thinner cross sectional area. Figure 28 – The intermediate baffle pitch S/Dh=1.50 – exhibits mild flow deflection, as flow is barely reattached. The reattachment length is 2.8mm long or 47% of the length of the pitch. Figure 28 –The greater baffle pitch S/Dh=2.50 – Promotes slightly greater flow deflection, and generates a reattachment length of 3.7mm, occupying 37% of the length of the baffle pitch. As the baffle pitch is lengthened the downstream and upstream recirculating vortices become less significant compared to the size of the pitch, and the flow tends to penetrate deeper into the pitch spacing, promoting greater flow deflection and consequently better mixing.

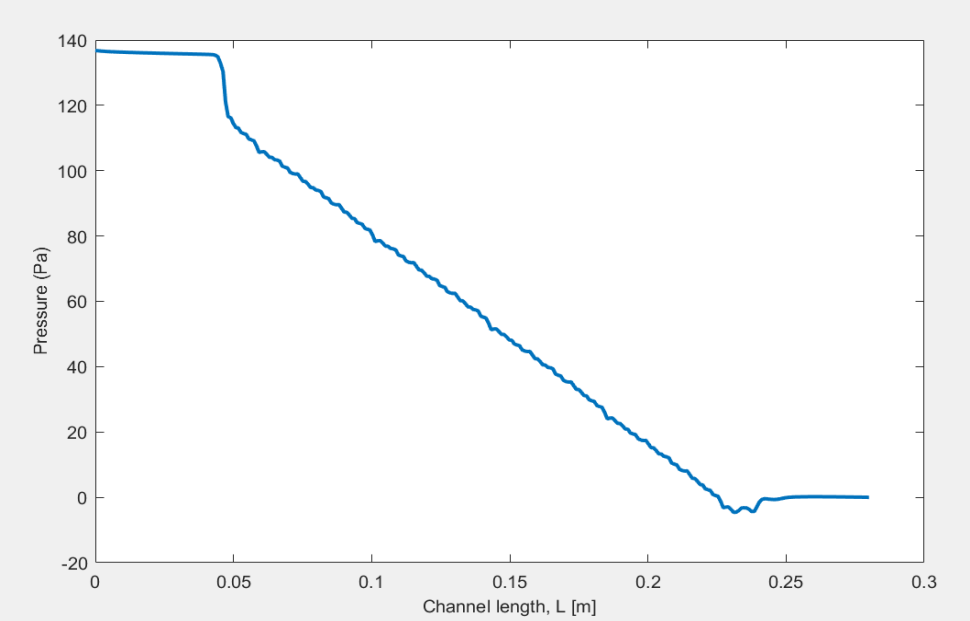

Figure 30. Pressure Drop S/D_h=0.50

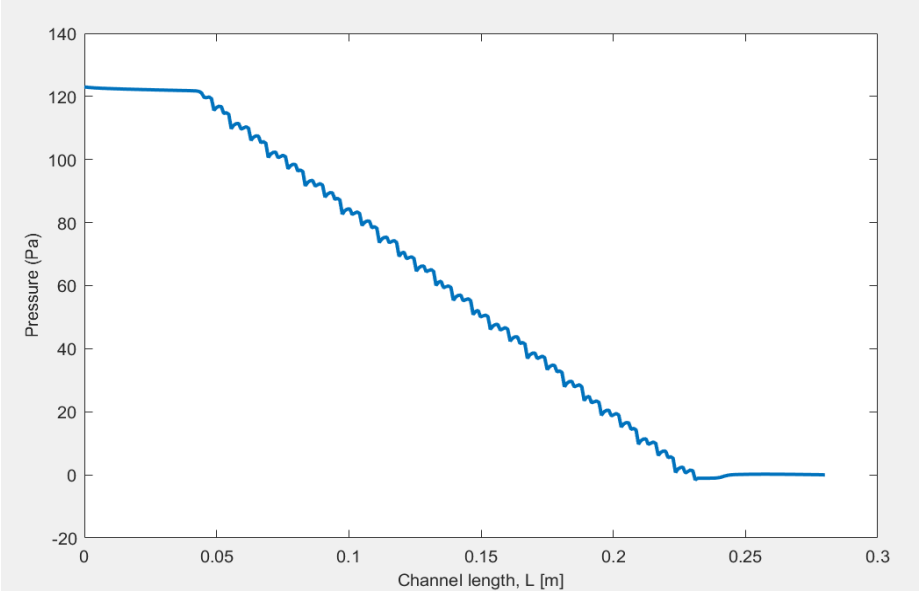

Figure 31. Pressure Drop S/D_h=1.50

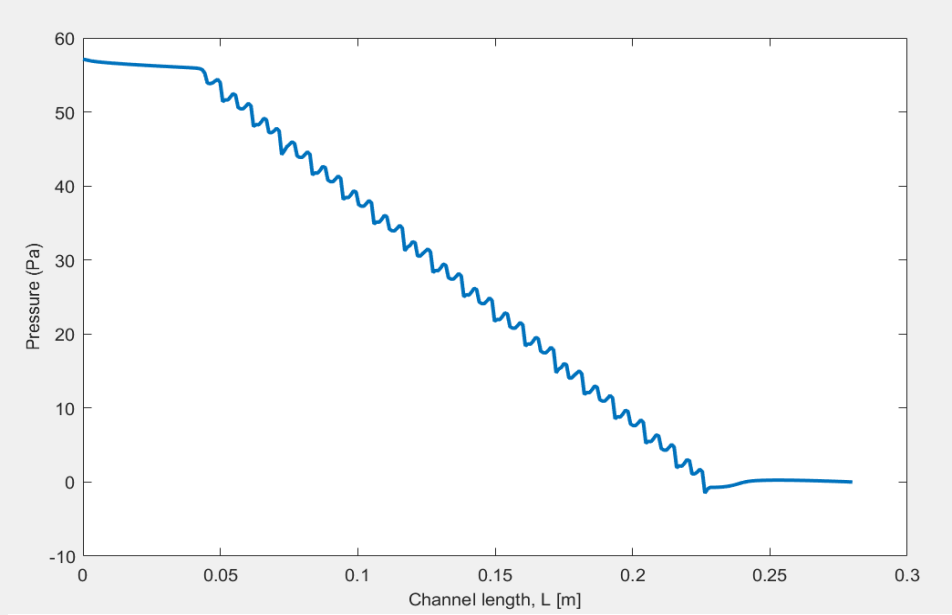

Figure 32. Pressure Drop S/D_h=2.50

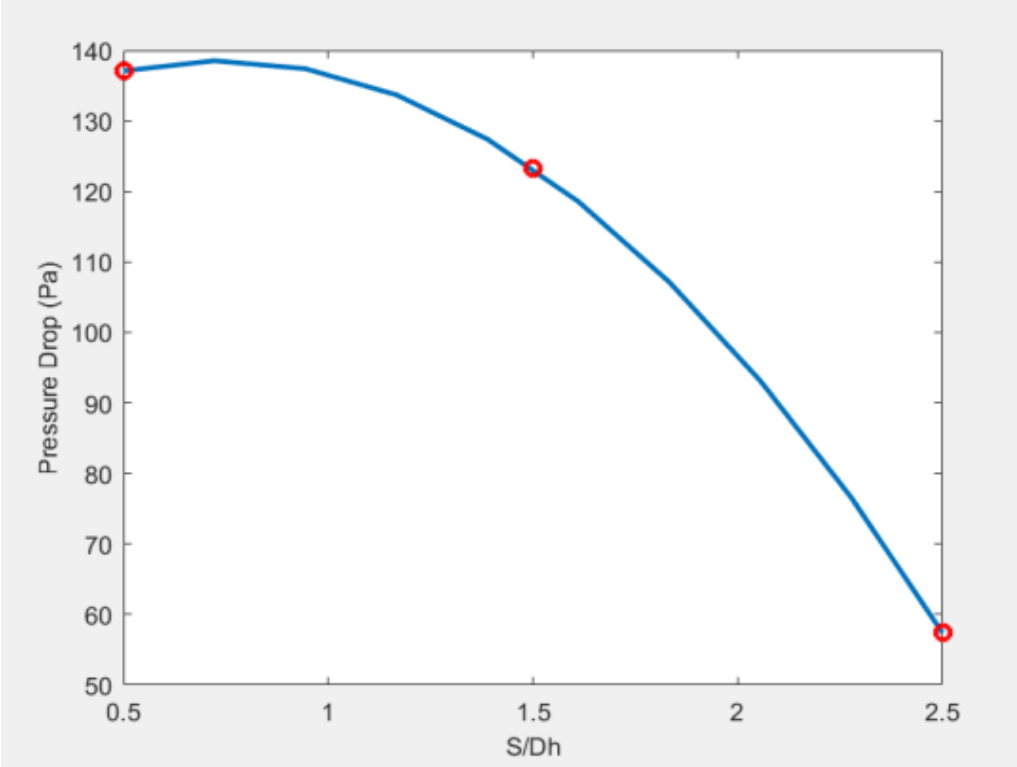

Figure 33. Pressure Drop x S/Dh

Figures 29 to 31 illustrate the effect of pitch on the total pressure drop. It was possible to notice that as baffle spacing is increased, flow has greater deflection, and causes impingent on the walls, but this is not the factor that has the greatest impact on the pressure drop. As the baffle spacing decreases the recirculating vortices fully occupy the pitch, and the "core" of the velocity of the flow can up to 2 times greater at S/Dh=0.5 compared to S/Dh=2.5 causing much more impact on pressure drop. The pressure drop for S/Dh=0.5, 1.5, 2.5 were 137 (Pa), 123.3 (Pa), and 57.44 (Pa) respectively. Varying S/Dh=0.5 to 1.5 decreased the pressure drop by 13.7 (Pa), while varying S/Dh=1.5 to 2.5 65.9 (Pa). It becomes clear that as pitch spacing increases the pressure drop is reduced at increasing rates, this is due to lower velocities in the core of the flow. Figure 31 displays the decay of pressure drop with increased pitch spacing S/Dh, showing an increasing rate of change trend. With the results of baffle height and baffle spacing, it is clear that the baffle spacing has a much greater effect on pressure drop.

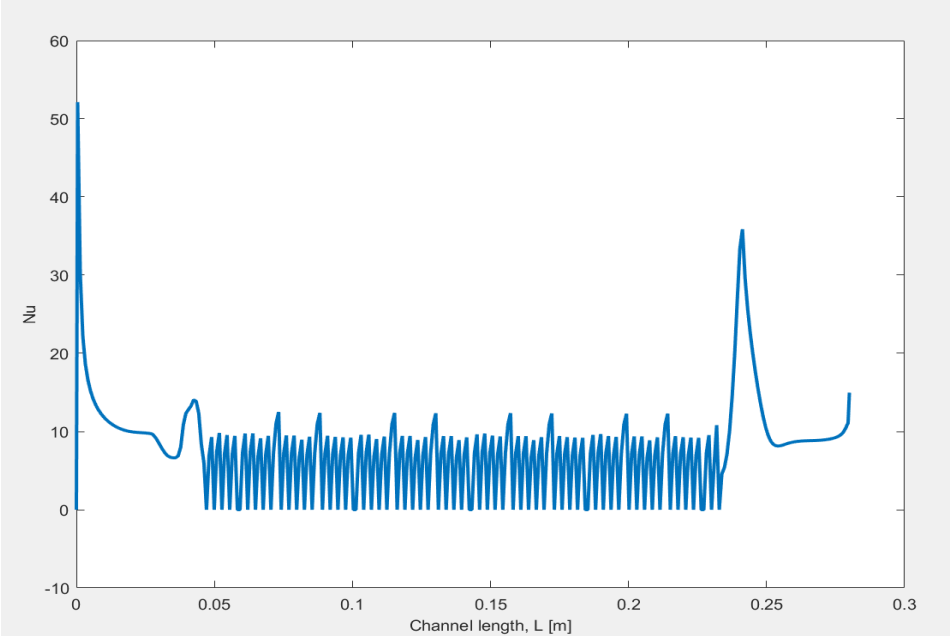

Figure 34. Nusselt S/D_h=0.50

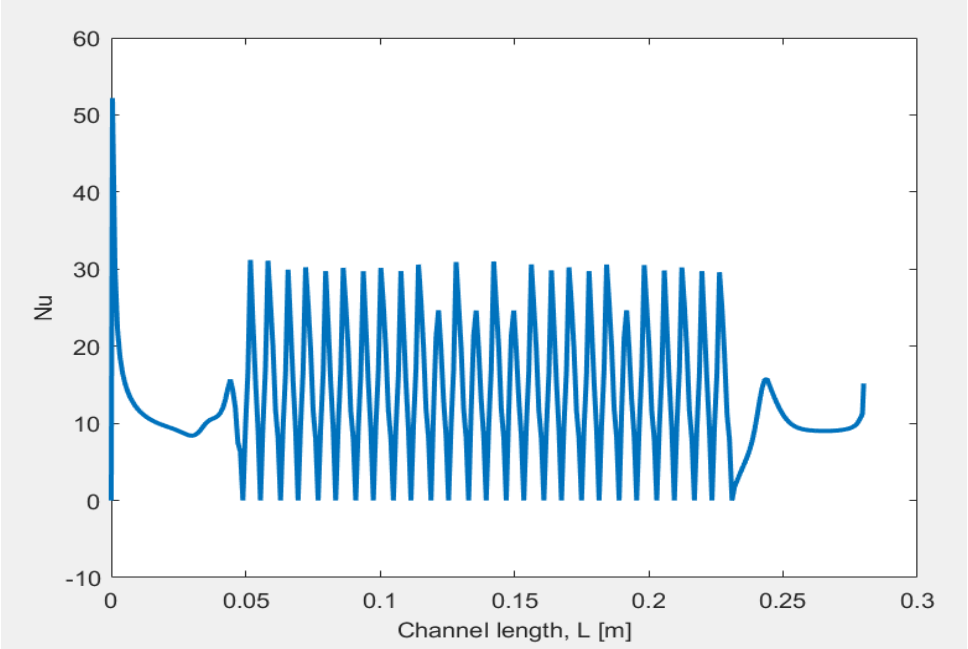

Figure 35. Nusselt S/D_h=1.50

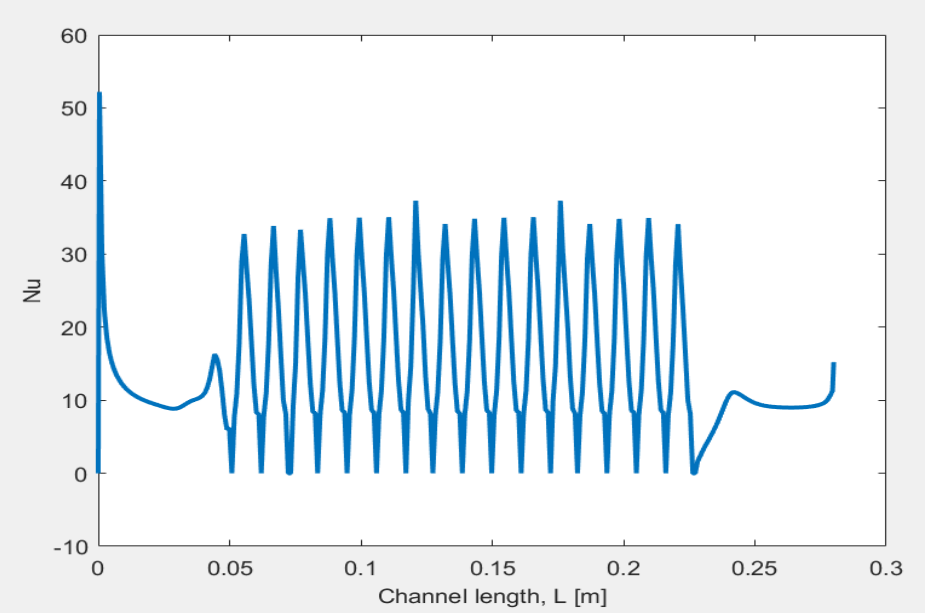

Figure 36. Nusselt S/D_h=2.50

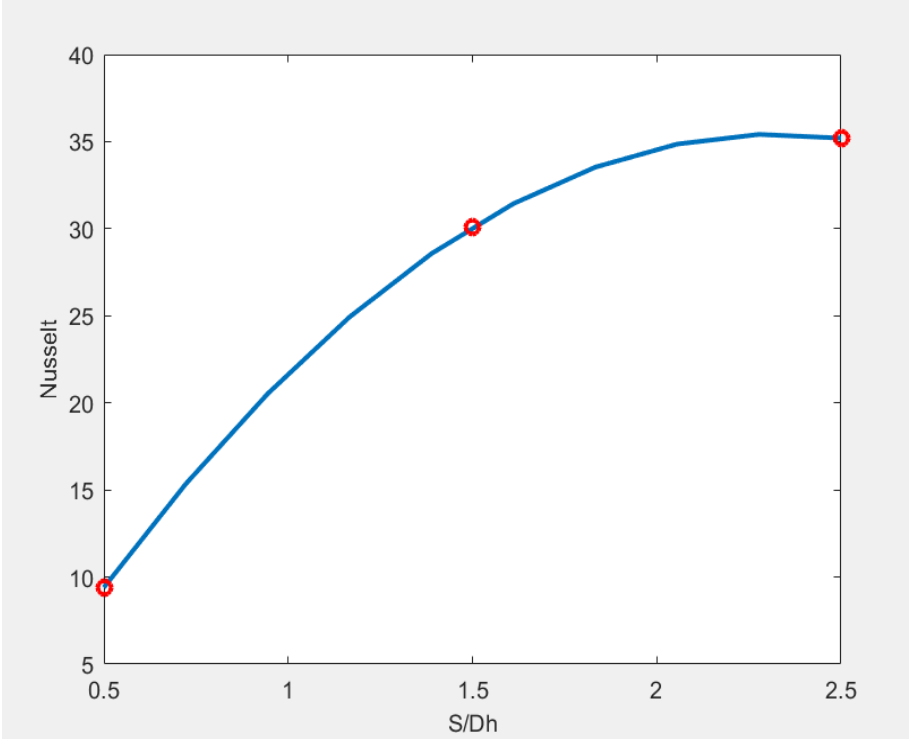

Figure 37. Nusselt x S/D_h

Figures 34 to 37 illustrate the effect of baffle pitch on the local Nusselt number. The baffles interrupt not only the flow, but also the thermal boundary layer. The Nusselt number reaches peak value after the recirculation eddies where flow is reattached. The peak Nusselt numbers for S/Dh=0.50, 1.50, 2.50 were 13, 30, 35. Varying S/Dh=0.50 to 1.50 increased the Nusselt number by 17 while varying S/Dh=1.50 to 2.50 only increased the Nusselt number by 5. Figure 37 illustrates this trend, as the rate of change decays approaching a constant after about S/Dh=2.00, where the Nusselt number reaches a maximum of 35.

Unlike the baffle height h/Dh, increasing the pitch S/Dh has a decreasing trend for the pressure drop, with an increasing trend of the Nusselt number during a short interval. This is extremely advantageous, as in the previous section heat transfer enhancement was associated with great increase in pressure drop. It is important to find equilibrium of both parameters, because even though this method has great advantage, the stabilization occurs in short ranges, while increasing baffle height results in much greater Nusselt numbers.

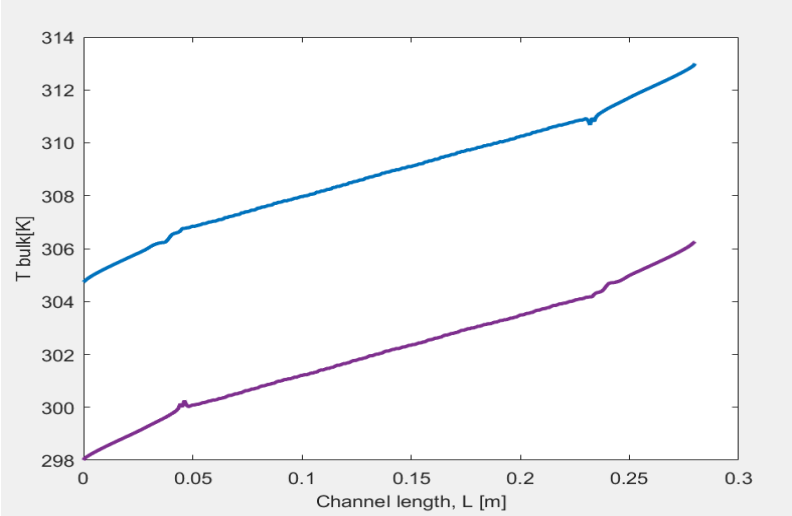

Figure 38. Bulk Temperature S/D_h=0.50

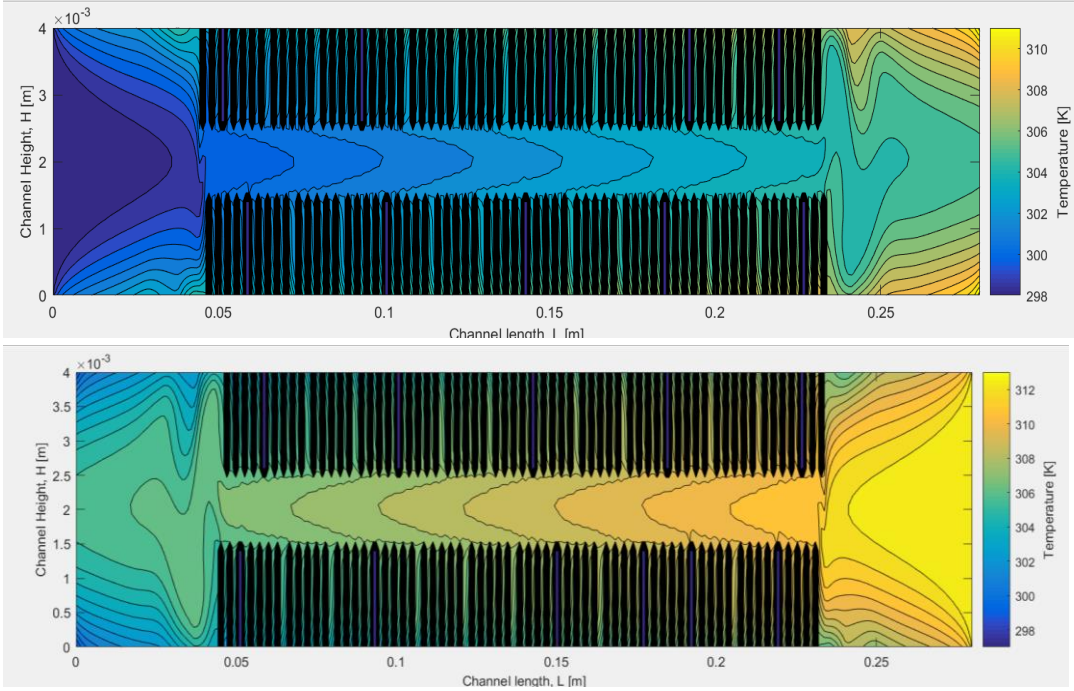

Figure 39. Temperature Profile of Channels S/D_h=0.50

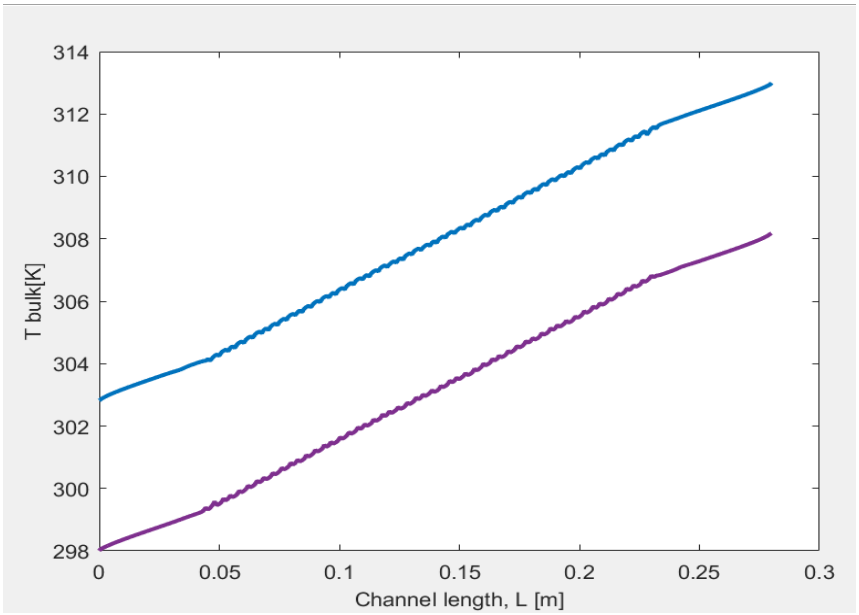

Figure 40. Bulk Temperature S/D_h=1.50

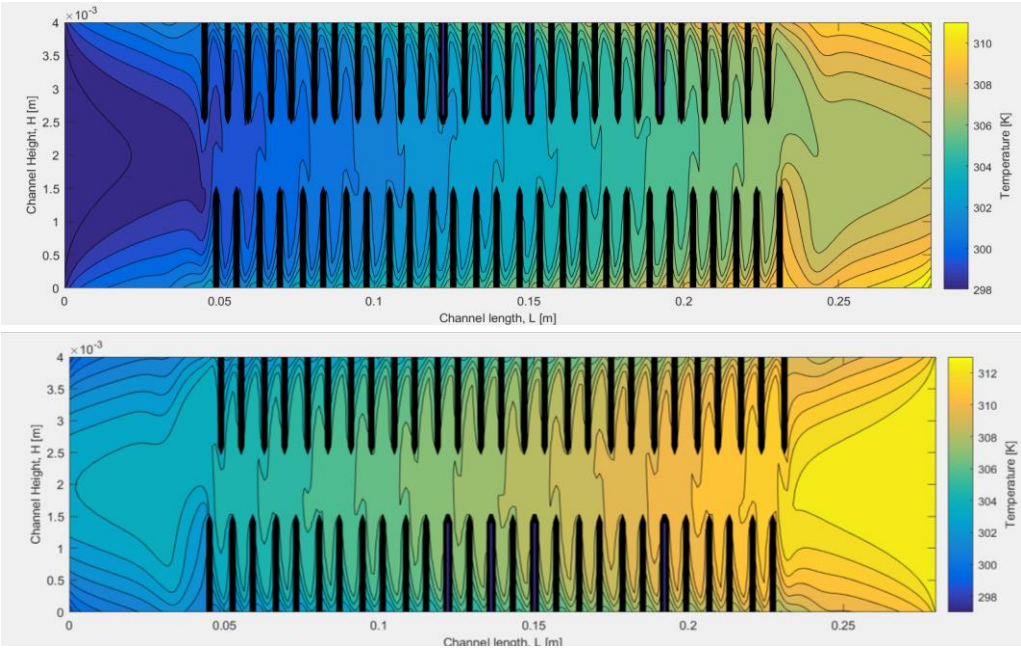

Figure 41. Temperature Profile of Channels S/D_h=1.50

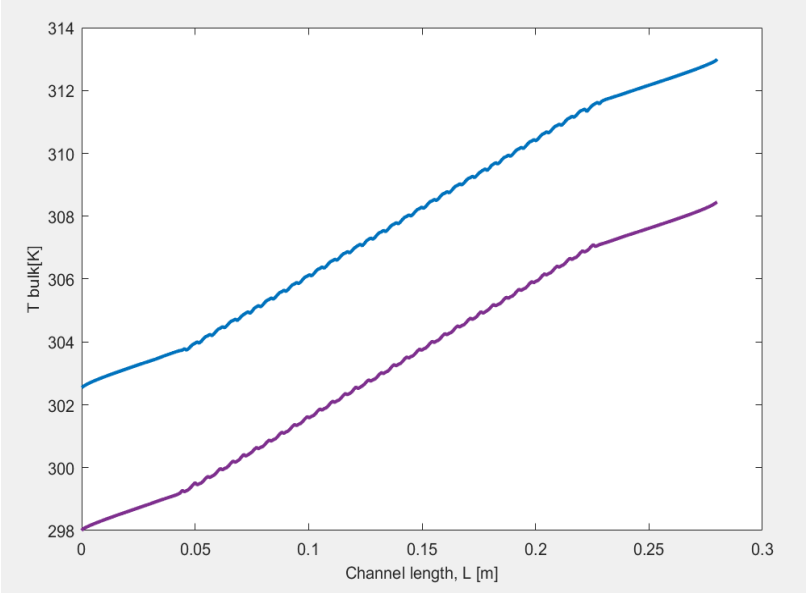

Figure 42. Bulk Temperature S/D_h=2.50

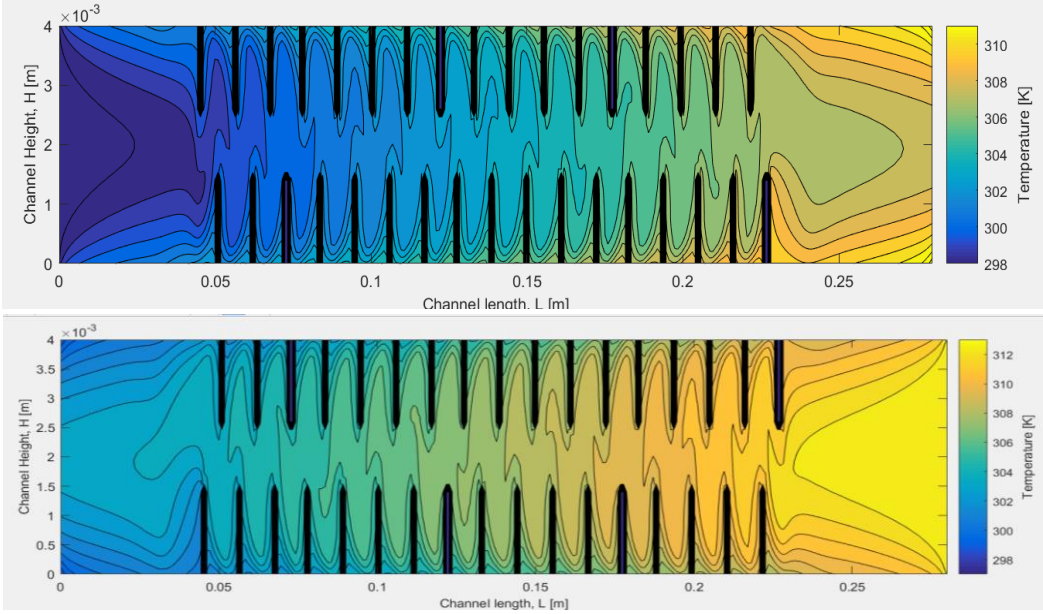

Figure 43. Temperature Profile of Channels S/D_h=2.50

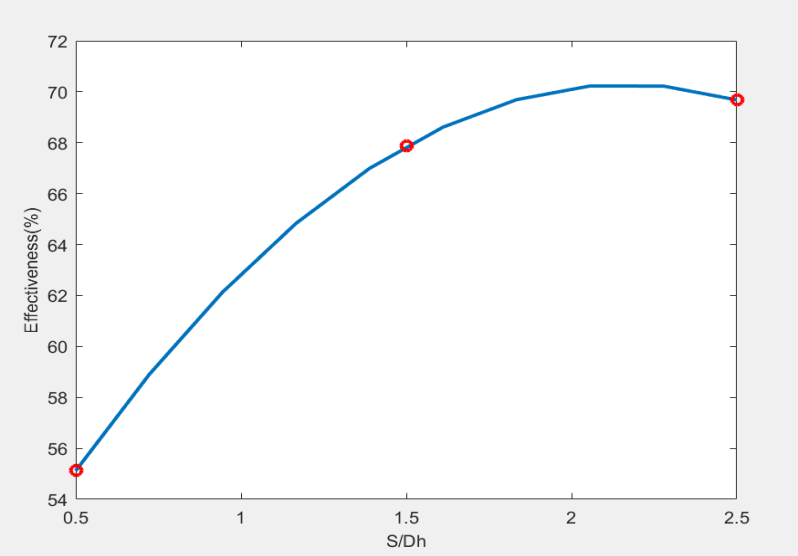

Figure 44. Heat Transfer Effectiveness x S/Dh

Figures 38 to 44 illustrate the effect of baffle pitch on temperature and heat transfer effectiveness. For S/Dh=0.50, 1.50, 2.50 the bulk temperature shift between the inlet and outlet of the two ducts were $\Delta T = 8.3$ K, 10.2 K and 10.5 K respectively. At S/Dh=0.50, there is very little mixing of the flow, consequently no interruption of the thermal boundary layer due to almost no flow deflection being present. As S/Dh=1.50 flow has much greater deflection since the recirculating vortices do not occupy the pitch fully like the first case, promoting much greater mixing of the thermal boundary layer. For S/Dh=2.50 the hydrodynamic reattachment length is only slightly longer than S/Dh=1.50 by less than 1 mm, this indicates that the recirculating vortices have about equivalent lengths even though we are dealing with a greater pitch length, and this explains why the shift after S/Dh=1.50 to a greater pitch does not promote significant temperature change.

Figure 44 illustrates clearly this effect by the heat transfer effectiveness plot. Increasing S/Dh=0.50 to 1.50 increases significantly the effectiveness, but the rates of change start to decay for any further increase in S/Dh, and tends to approach a constant effectiveness after S/Dh=2.0.

As mentioned in the previous section, the temperature contour plots have a strong link with the velocity vector plots, and recirculating vortices downstream of the vortices behave as an insulation zone, shown by the zero temperature gradient at those regions.

This section proves that increasing the baffle pitch can sometimes be limited at very short ranges, and to get the more heat transfer enhancement, the height has much greater impact. A good combination of height and pitch is important as the baffle height has great impact on the downstream vortices and an adequate pitch could minimize the pressure drop.

After analysis of results, it is clear that the Nusselt number, and effectiveness reaches the peak value and stabilizes after S/Dh=2, but the pressure drop continues to decrease until S/Dh=2.5, making it the most effective pitch for this application.

Effect of Reynolds Number

To determine the effects of Reynolds number, the simulation was carried out using a constant baffle spacing of S/Dh=2.50 and a baffle height ratio of h/Dh=0.35, which was determined as optimum in the previous section. While keeping these variables constant the Reynolds number was varied from Re= 200,300, 400.

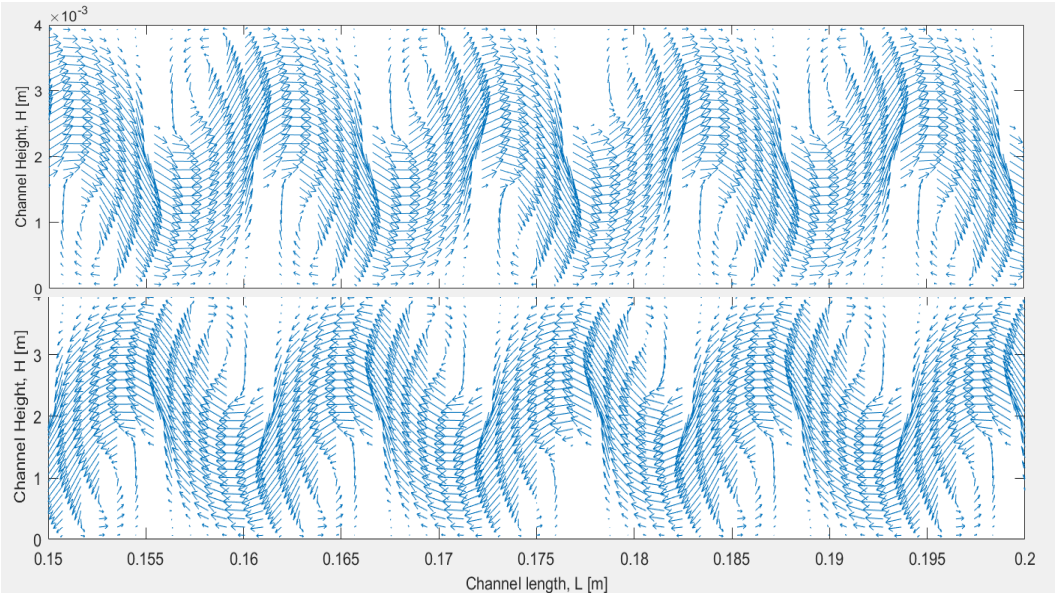

Figure 45. Velocity Vector Plots Re=200

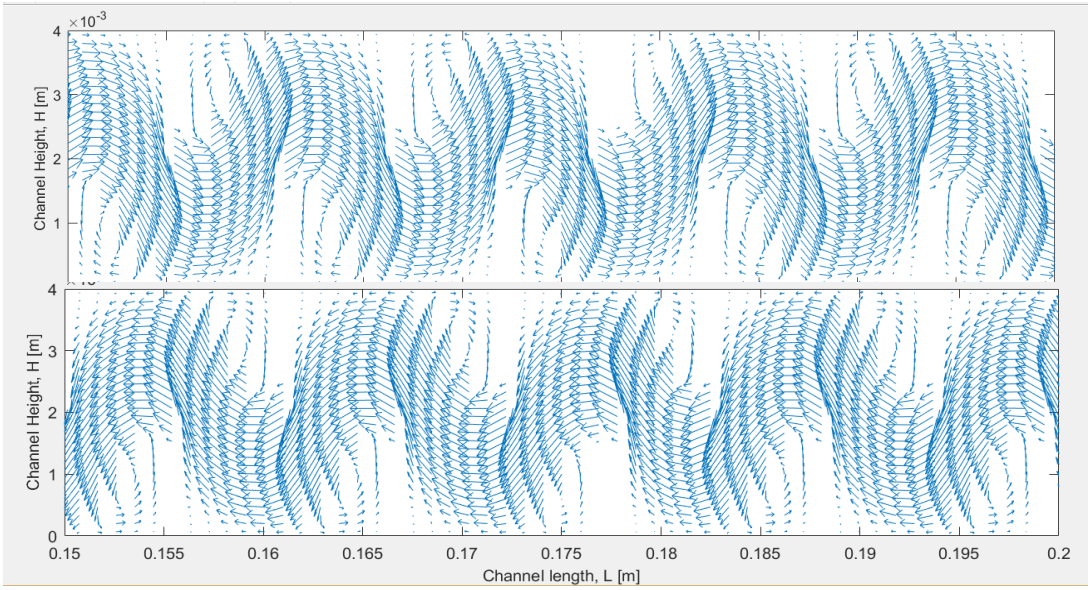

Figure 46. Velocity Vector Plots Re=300

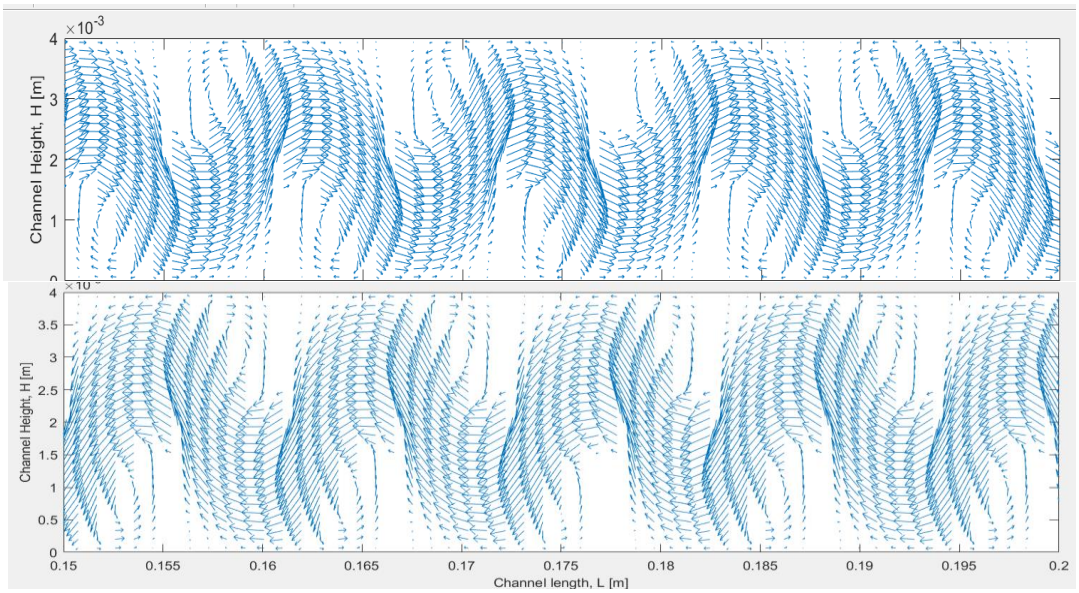

Figure 47. Velocity Vector Plots Re=400

Figures 45 to 46 illustrate the effect of Reynolds number on the hydrodynamic flow profile for the 0.15m to 0.2m section of the ducts. From the results above Reynolds number is varied while all other parameters remain constant, including the geometric and fluid properties, so in other words the Reynolds number is only changing through velocity increase. The Reynolds number was varied Re 200, 300 and 400, and it is seen that the velocity has no impact on the reattachment length or the length of the recirculating vortices which for all was 3.7 mm. So for this manner the only visible change in the hydrodynamic profile is the magnitude of the velocity vectors.

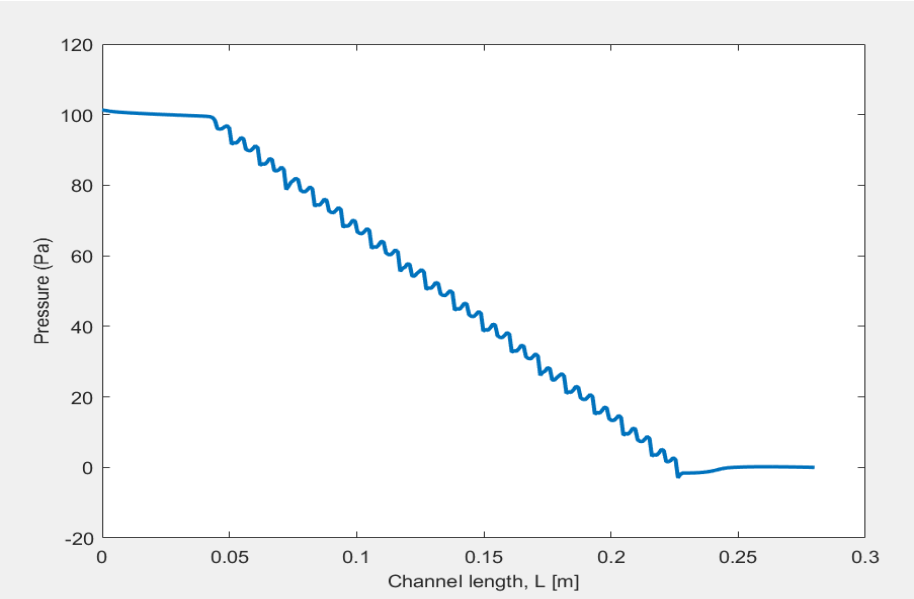

Figure 48. Pressure Drop Re=200

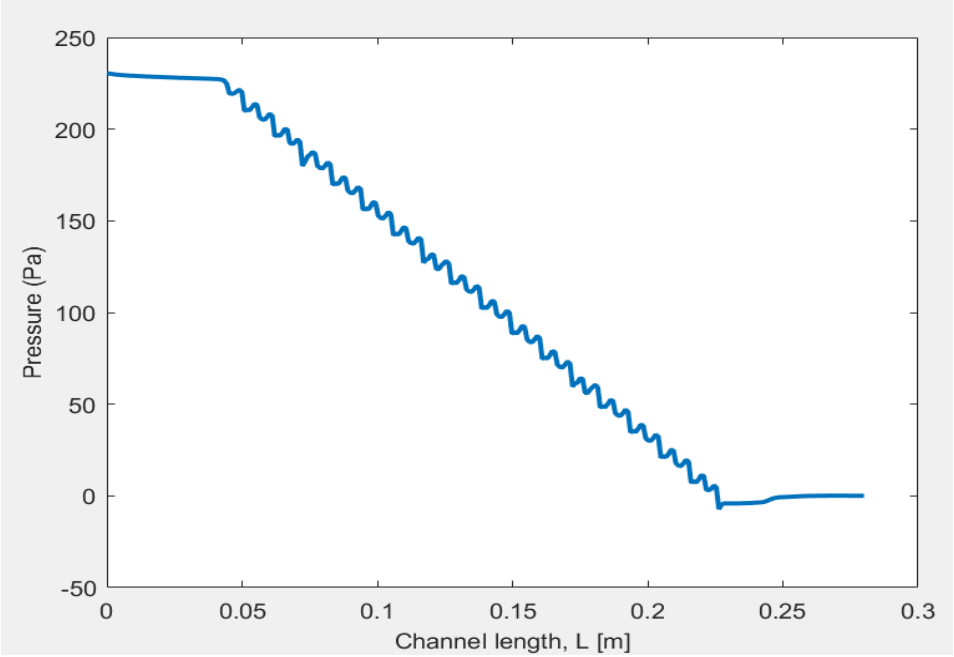

Figure 49. Pressure Drop Re=300

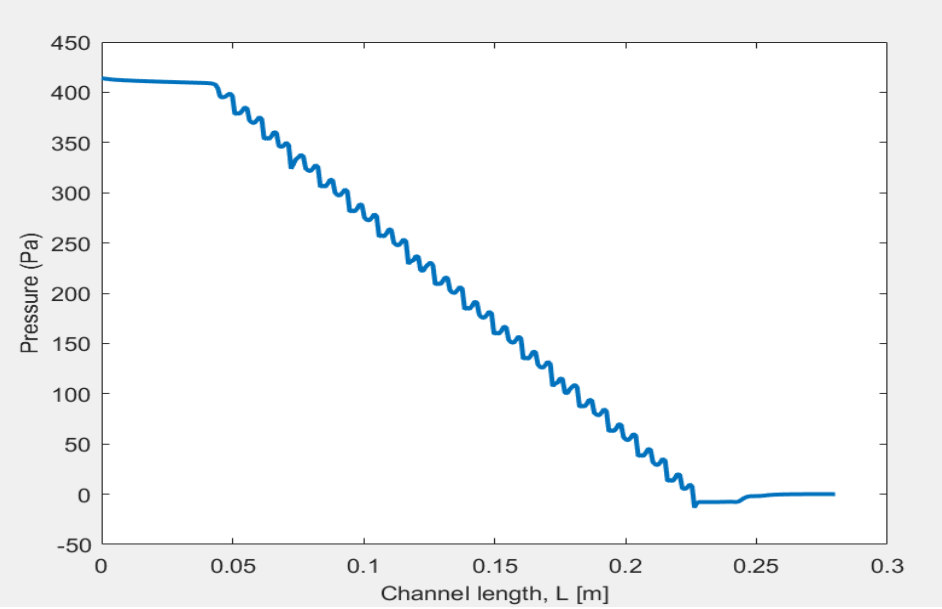

Figure 50. Pressure Drop Re=400

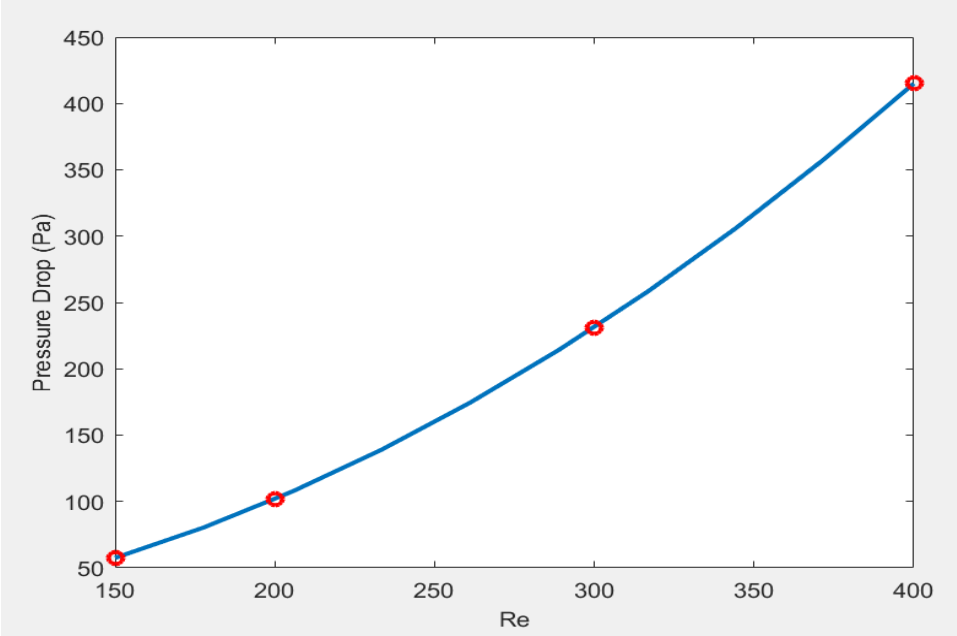

Figure 51. Pressure Drop x Re

Figures 48 to 51 illustrate the effect of Reynolds number on the total pressure drop. As mentioned previously the only parameter changing the Reynolds number is the velocity, and as it is known, the pressure drop is proportional to the velocity squared. For Re= 200, 300, 400 the pressure drops were 101.81 (Pa), 231.28 (Pa), 415.06 (Pa). Figure 51 displays the trend for the Re x Pressure drop.

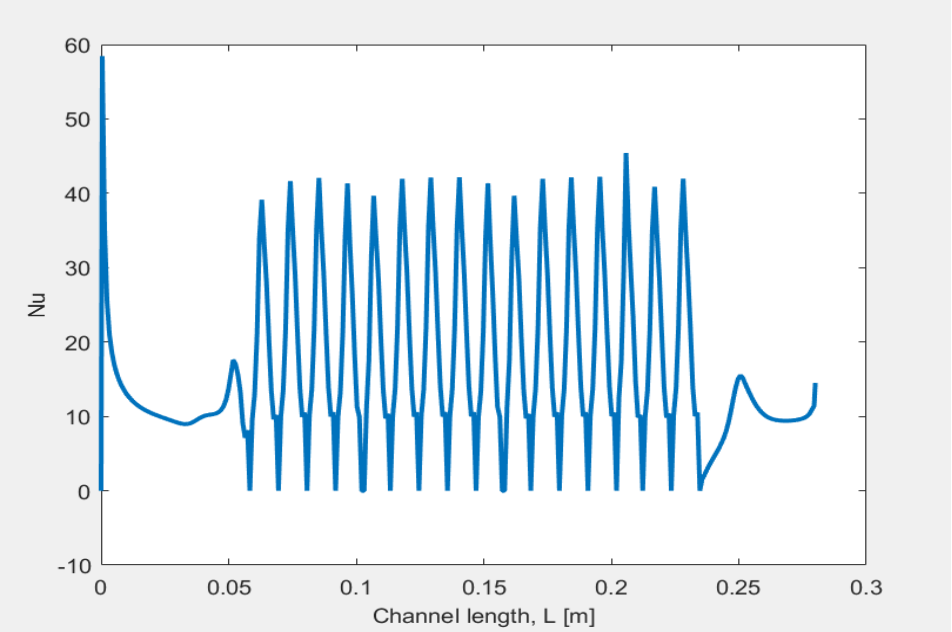

Figure 52. Nusselt Re=200

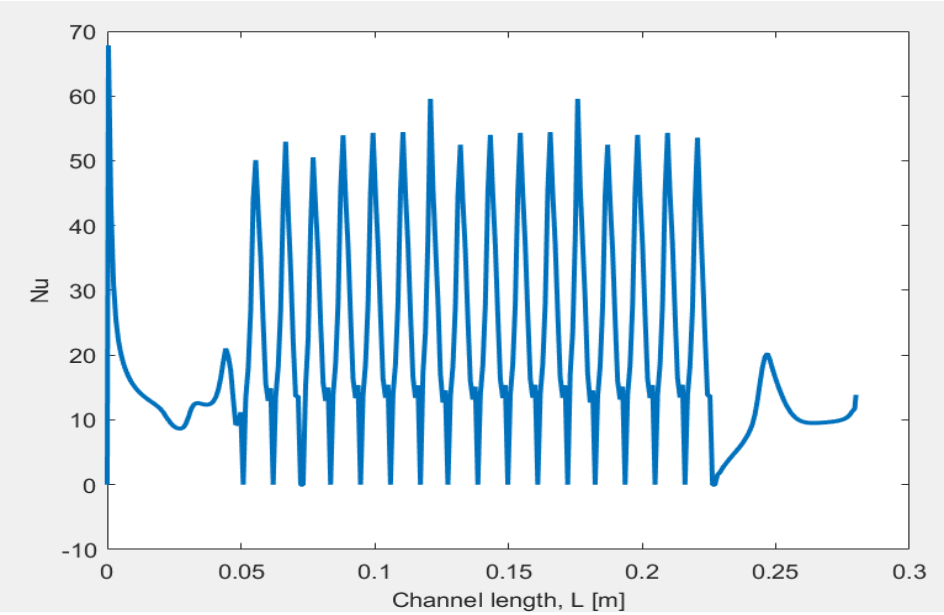

Figure 53. Nusselt Re=300

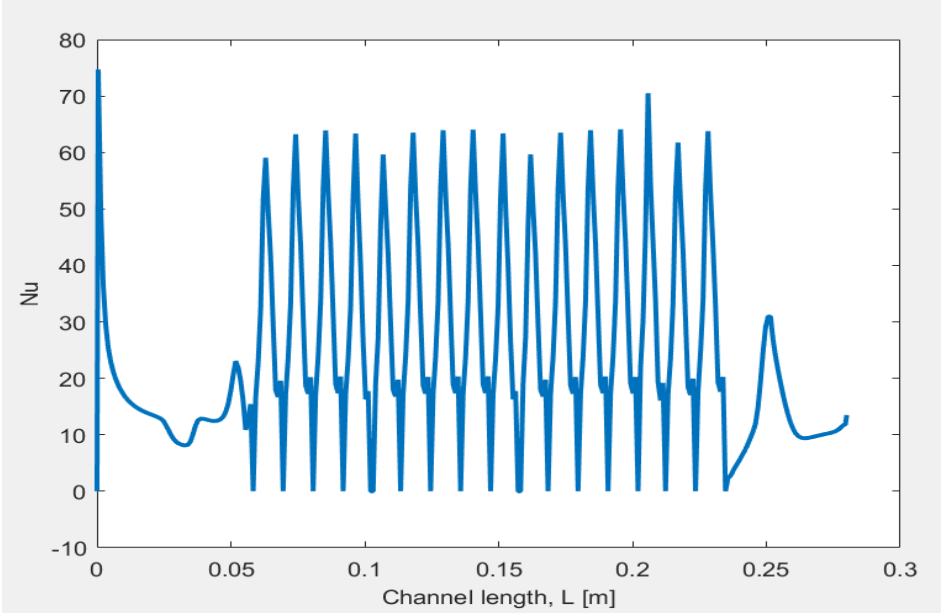

Figure 54. Nusselt Re=400

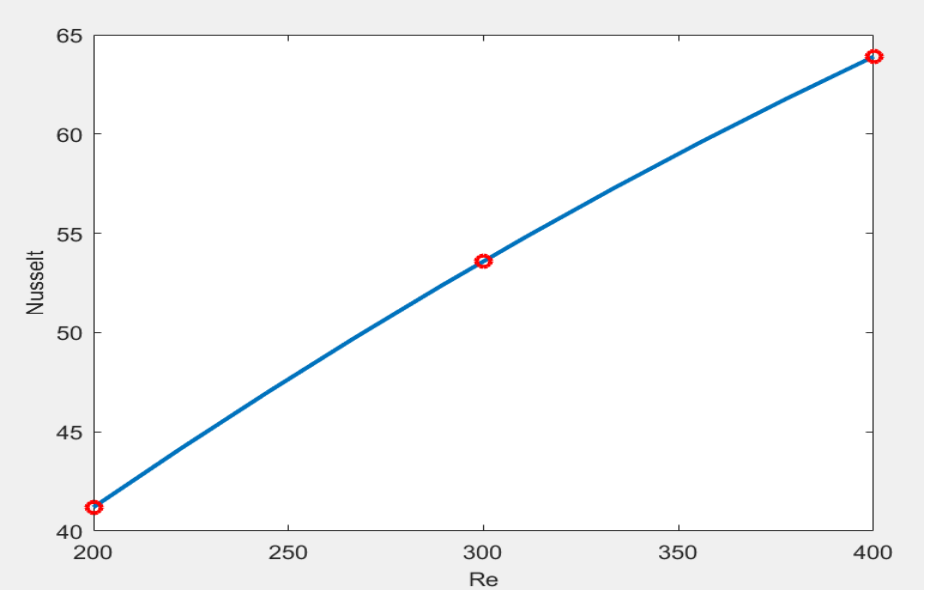

Figure 55. Nusselt x Re

Figures 52 to 55 illustrate the effect of Reynolds number on the local Nusselt number. The baffles interrupt not only the flow, but also the thermal boundary layer. The peak Nusselt numbers for Re=200, 300, 400 were 41.2, 53.6 and 63.9. Varying the Reynolds number from Re=200 to Re=300 increased the Nusselt number by 12.4 and while increasing the Re=300 to Re=400 the Nusselt number was increased by 10.3.

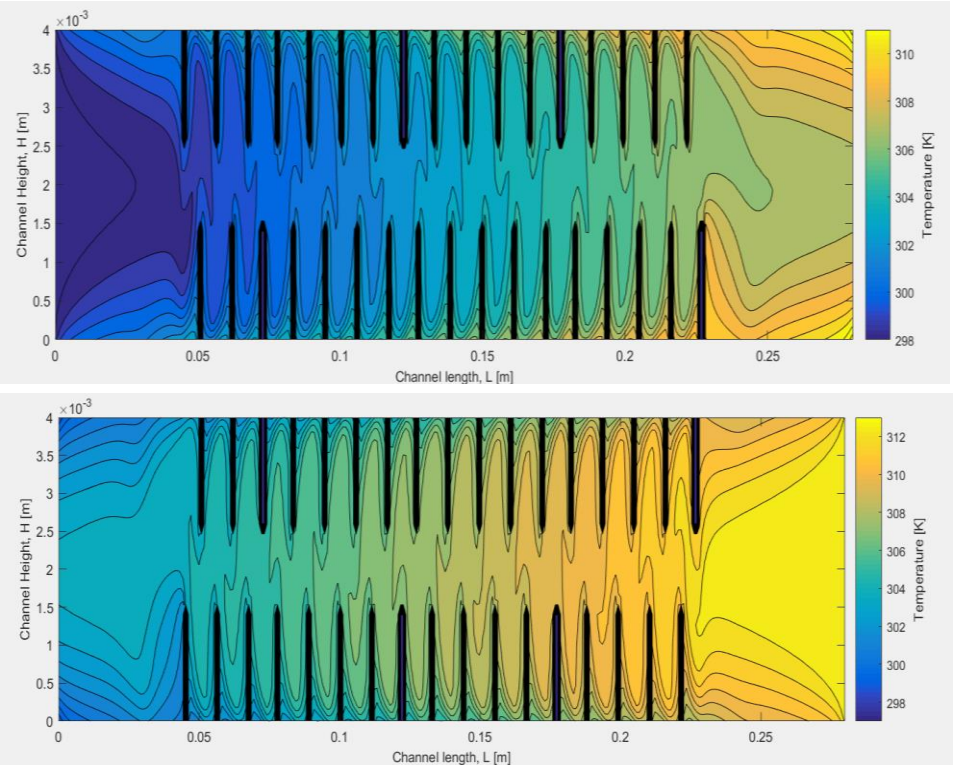

Figure 56. Temperature Profile of Channels Re=200

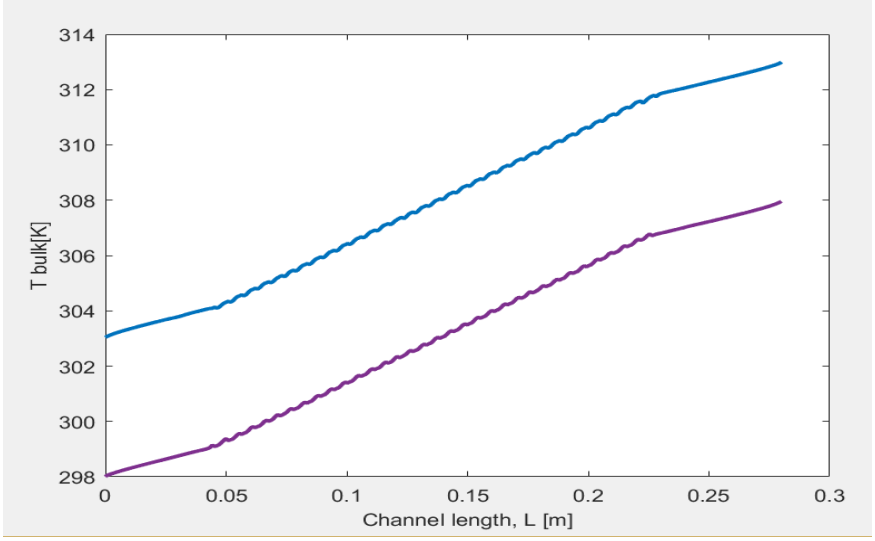

Figure 57. Bulk Temperature Re=200

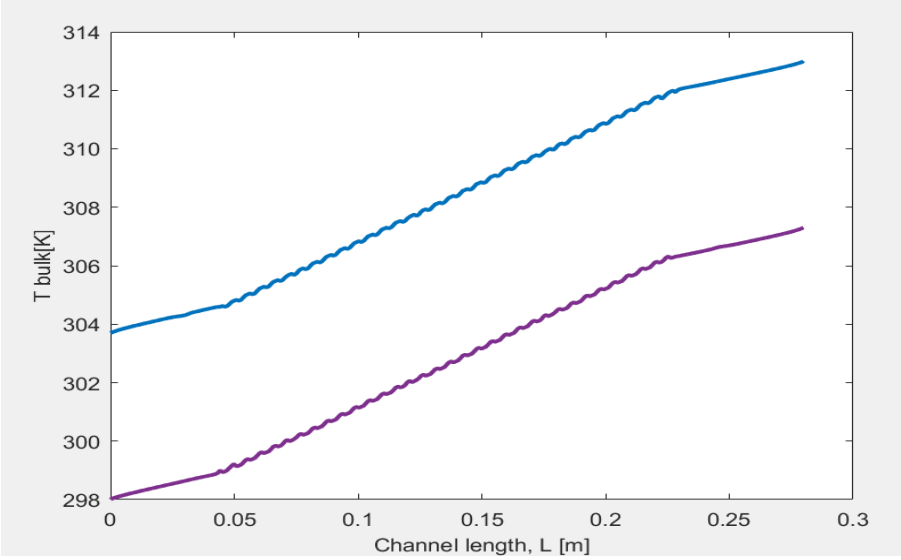

Figure 58. Bulk Temperature Re=300

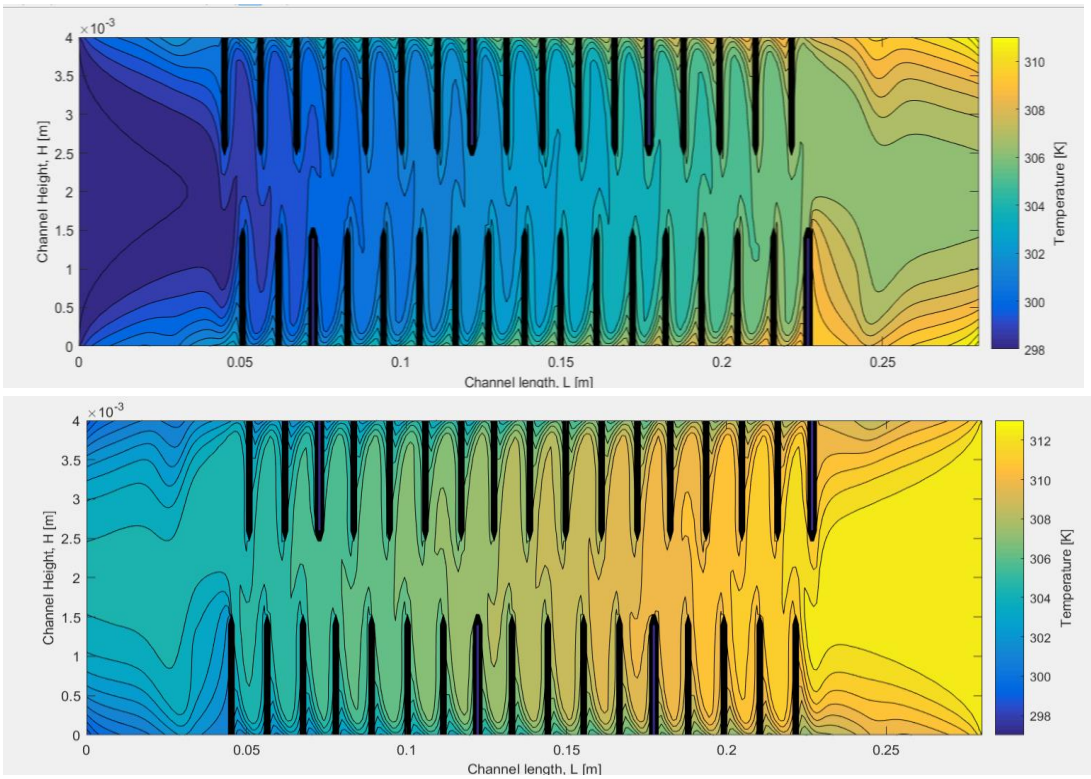

Figure 59. Temperature Profile of Channels Re=300

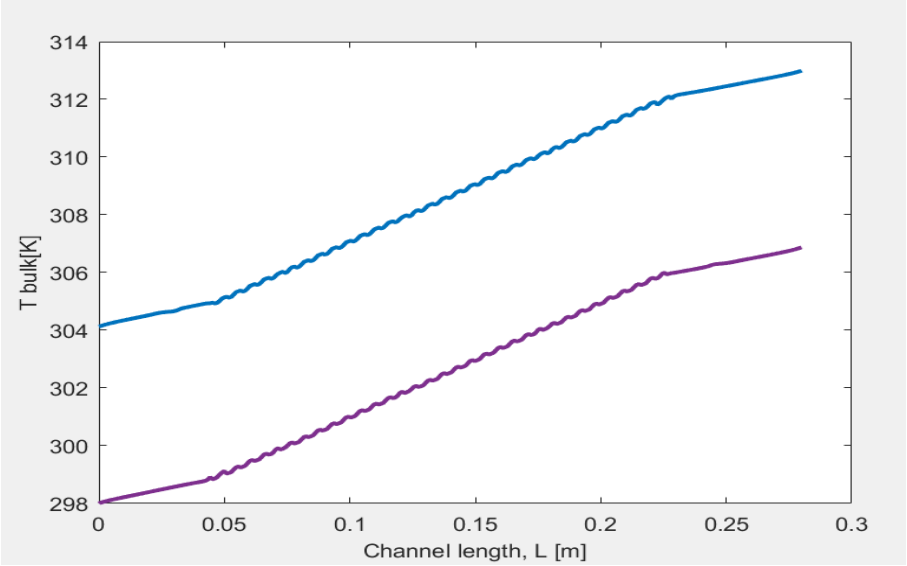

Figure 60. Bulk Temperature Re=400

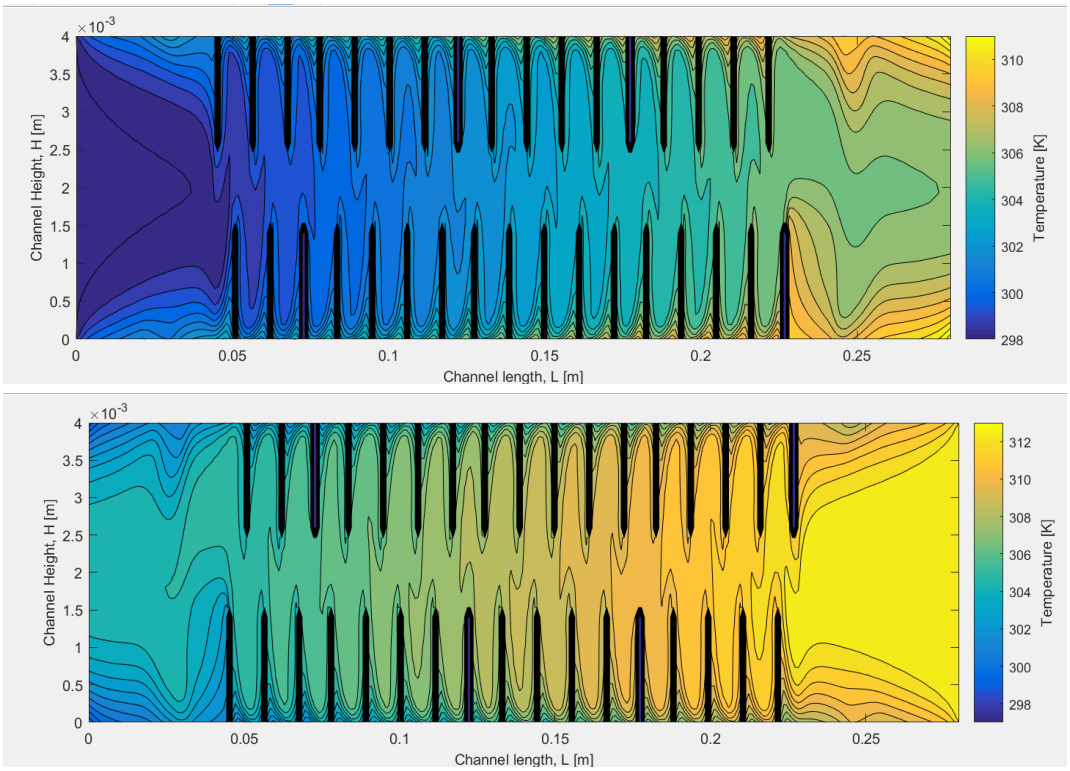

Figure 61. Temperature Profile of Channels Re=400

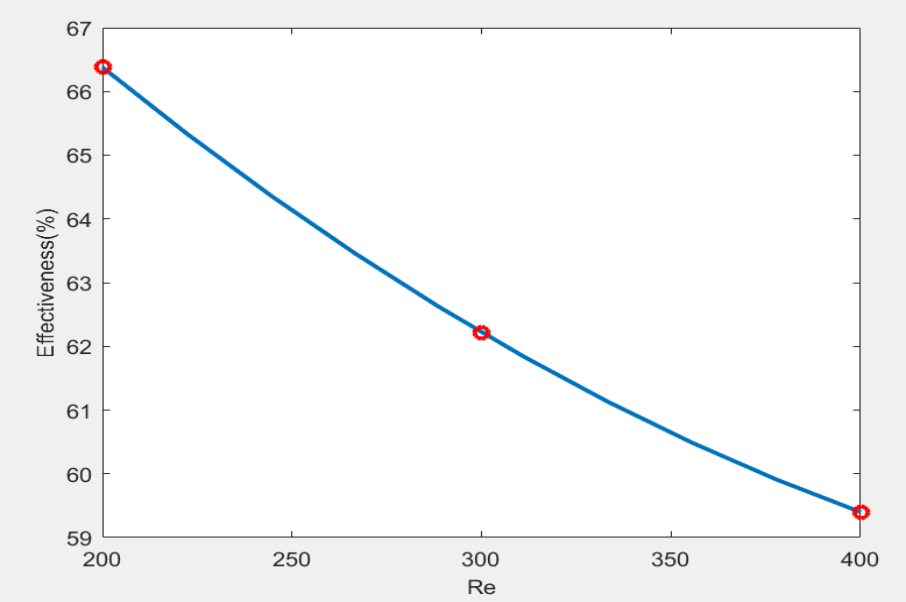

Figure 62. Heat Transfer Effectiveness x Re

Figures 57 to 62 illustrate the effect of Reynolds number on temperature and heat transfer effectiveness. For Re= 200,300,400 the bulk temperature shift between the inlet and outlet of the two ducts were $\Delta T = 9.0$ K , 9.3 K, 8.9 K respectively. Even though with increasing Reynolds number the heat transfer coefficient also increases -shown by figure 55 – the temperature change between the inlet and outlet of the duct slightly decreases. This happens even though you have a heat transfer enhancement, because as the velocity is increases with Reynolds number and the residence time of the air in the duct is lower, providing less time for heat to be exchanged. Figure 62 shows that heat transfer effectiveness decreases with increasing Reynolds number, and this also occurs because of the air residence time, for lower velocities the residence time is longer and as velocity increases the residence time decreases.

After analysis of results, it is clear that the increase in Reynolds number contributes to greater pressure drop, while providing only a marginal increase in the Nusselt number. Also with the increase of Reynolds number the residence time decreases along with the heat transfer effectiveness. The original value of Re=150 is the best option for this simulation.

CHAPTER V

CONCLUSION

A numerical study was performed to determine the effects of baffle height, baffle pitch and Reynolds number on the overall performance of a heat recovery ventilator. A numerical model using the steady state 2D convective diffusion equation using finite difference was developed.

After analysis of the results it became evident that the baffle height has the greatest effect on the overall performance. At greater baffle heights the flow deflection, velocity of "core" flow, pressure drop, the Nuselt number, and heat transfer effectiveness all increased while the reattachment length decreased. The pressure drop increased at much greater rates than the Nusselt number, so it was important to define an optimal point for the height. The optimum height for the selected range was defined as h/dh=0.35, because when setting h/dh=0.45 the pressure drop nearly doubled for only a small increment of only 3% of the heat transfer effectiveness.

The results showed that the baffle pitch has less effect on the overall performance after flow is reattached. As the pitch spacing is occupied fully by the recirculating vortices, the pressure drop is the highest and as the pitch is lengthened, flow starts to reattach increasing the Nusselt number, reattachment length, heat transfer effectiveness until becoming stable, while the pressure drop has a continuous decreasing trend and stabilization occurs at greater pitch lengths.

The change of Reynolds number had no impact on the reattachment length, but since it was entirely linked to a velocity change the pressure drop was greatly influenced. As the Nusselt number was increased, the velocity, pressure drop and Nusselt number increased, while the heat transfer effectiveness decreased, due to the lower residence time. The results show that the lowest Nusselt numbers present the greatest heat transfer effectiveness, with lower pressure drops. The Reynolds number affects the pressure drop, much more than the baffle height or baffle spacing, without providing significant increase in the Nusselt number, while also decreasing the heat transfer effectiveness.

For further work it is recommended to experiment this model and confront results, to determine if the numerical model is accurate with a practical application. Also since the rectangular baffles interrupt the flow abruptly, a future simulation with a baffle geometry that can offer a smooth transition producing flow deflection without large recirculating vortices and pressure drops.

REFERENCES

Albakhit, H. and Fakheri A., 2005 “A hybrid approach for full numerical simulation of heat exchangers” ASME Heat Transfer Summer Conference, July17–22, 2005, San Francisco, CA, USA, 2005.

ASHRAE., 2005 “Handbook of Fundamentals, American Society of Heating,” Refrigerating and Air-Conditioning Engineers, Inc, Atlanta, GA.

Cheng,C.H. and Huang,W.H., 1990 “Numerical prediction for laminar forced convection in parallel-plate channels with transverse fin arrays” International Journal of Heat and Mass Transfer Vol. 34 (11), pp. 2739-2749.

Chompookham,T. , Thianpong,C., Kwankaomeng,S., and Promvonge,P., 2010 “Heat transfer augmentation in a wedge-ribbed channel using winglet vortex generators” International Communications in Heat and Mass Transfer Vol. 37 , pp. 163–169.

F. Incropera, P.D. Dewitt, 1996, “Introduction to Heat Transfer”, 3rd ed, John Wiley & Sons Inc.

Fullerton, T.L. and Anand, N.K. 2010, “An alternative approach to study periodically fully-developed flow and heat transfer problems subject to isothermal heating conditions” International Journal of Engineering Science Vol. 48 , pp. 1253–1262.

Habib, M.A., Mobarak, A.M., Sallak, M.A., Abdel Hadi ,E.A., and Affify, R.I., 1994, “Experimental investigation of heat transfer and flow over baffles of different heights, Trans.ASME J. Heat Transfer Vol.116, pp. 363–368.

Han, H., Choo, YB. and Kwon, YI., 2007 “An Experimental Study on the Effect of Outdoor Temperature and Humidity Conditions on the Performance of a Heat Recovery Ventilator” Proceedings of Clima 2007 Well Being Indoors.

Hasan,M., Rageb, A.A., Yaghoubi, M., and Homayoni,H., 2009, “Influence of channel geometry on the performance of a counter flow microchannel heat exchanger” International Journal of Thermal Sciences Vol.48, pp. 1607–1618.

Hoon Ko,K., and Anand,N.K., 2003, “Use of porous baffles to enhance heat transfer in a rectangular channel” International Journal of Heat and Mass Transfer Vol.46, pp. 4191–4199.

Idayu, A. and Riffat, S.B., 2012“Review on Heat Recovery Technologies for Building Applications,” Renewable and Sustainable Energy Reviews, Elsevier; vol.16 (2):pp.1241–1255.

Jedsadaratanachai,W., Suwannapan,S., and Promvonge,P., 2011, “Numerical study of laminar heat transfer in baffled square channel with various pitches” Energy Procedia Vol. 9, pp. 630 – 642.

Kelkar, K. M. and Patankar, S.V.,1987 “Numerical Prediction of Flow and Heat Transfer in a Parallel Plate Channel With Staggered Fins” J. Heat Transfer 109(1):pp.25-30.

Kim,G.W., Lim,H.M., Rhee.,G.H., 2016, “Numerical studies of heat transfer enhancement by cross-cut flow control in wavy fin heat exchangers” International Journal of Heat and Mass Transfer Vol.96, pp. 110–117.

Kundu, P., Cohen, I., Dowling,D., 2012 “Fluid Mechanics”, Elsevier Inc,Waltham-MA, 5th ed.

Li, Q., Flamant,G., Yuana,X., Neveub,P. and Luo,L., 2011 “A review and future applications for a new generation of high temperature solar receivers,” Renewable and Sustainable Energy Reviews, Elsevier; vol.15:pp.4855–4875.

Mousavi, S.S. and Hooman, K., 2005 “Heat and fluid flow in entrance region of a channel with staggered baffles” Energy Conversion and Management, Vol 47(15-16): pp. 2011–2019.

Patankar,S. V., Liu C. H. and Sparrow E. M.,1977 “Fully developed flow and heat transfer in ducts having streamwise-periodic variation of cross-sectional area” Am. Sot. Mech. Enyrs, Series C, J. Heat Transfer 99, 180-186.

Patankar S. V., 1980 “Numerical heat transfer and fluid flow,” Hemisphere Publishing Corporation, New York-NY.

Promvonge,P., and Thianpong,C., 2008 “Thermal performance assessment of turbulent channel flows over different shaped ribs” International Communications in Heat and Mass Transfer Vol. 35 , pp. 1327-1334.

Rowley, G.J. and Patankar, S.V.,1984 “Analysis of laminar flow and heat transfer in tubes with internal circumferential fins” International Journal of Heat and Mass Transfer 27(4):553-560.

Shurcliff, W.,1988 “Air-to-air heat exchangers for houses,” Annual Review Energy;(13):pp.1–22.

Webb, B. W. and Ramadhyani, S., 1985 “Conjugate heat transfer
in a channel with staggered ribs” , ht. J. Heat Mass Transfer 28, 1679-1687.

Yaïci,W., Ghorab,M., and Entchev,E., 2012 “Numerical analysis of heat and energy recovery ventilators performance based on CFD for detailed design” Applied Thermal Engineering, Vol 51(1-2):pp 770–780.

Zhang, L.Z., 2008 “Total Heat Recovery, Heat and Moisture Recovery from Ventilation Air,” Nova Science Publishers, Inc., New York.

APPENDIX A

MATLAB CODE

```
Final 2

% Thesis Felipe Assuncao
% SIMPLE algorithm applied to stagerred baffled channel


clear all
close all
clc
% calculate CPU time
tic
% Call Properties
global rho mu k cp q Re lx ly nx ny umax k_vec EntL h Th Pit ExiL D D_vec
[rho,mu,k,cp,q,Re,lx,ly,nx,ny,umax,EntL,h,Th,Pit,ExiL,D] = property;


% - x,y            : location of grid points
% - dx,dy          : diffusion length
% - aW,aE,aS,aW,aP : coefficient

% defines an empty struct array for pressure and velocity
ps1 = struct('x',[],'y',[],'dx',[],'dy',[],...
    'aW',[],'aE',[],'aS',[],'aN',[],'aP',[]);
us1 = ps1; vs1 = ps1; ts1=ps1;
ps2=ps1; us2=ps1; vs2=ps1; ts2=ts1;
rhos1=ps1;rhos2=ps2;

% Call Geometry
[ps1] = geometry(ps1);
[us1] = geometry(us1,0);
[vs1] = geometry(vs1,1);
[ts1] = geometry(ts1);

[ps2] = geometry(ps2);
[us2] = geometry(us2,0);
[vs2] = geometry(vs2,1);
[ts2] = geometry(ts2);


% Call Boudary Conditions
[p1,p2,pc1,pc2,u1,u2,v1,v2,T1,T2] = bcic2();
% Call Boudary Conditions for Baffles
[mu,k_vec,D_vec]=blockage(mu);
mu1=mu;
mu2=fliplr(mu);
mu2=flipud(mu2);

k_vec1=k_vec;
k_vec2=fliplr(k_vec);
k_vec2=flipud(k_vec2);

D_vec1=D_vec;
D_vec2=fliplr(D_vec);
D_vec2=flipud(D_vec2);

% Initialize Residue and Iteration
Ru1=ones; Ru2=ones; Rv1=ones; Rv2=ones; Rp1=ones;Rp2=ones; R1=zeros; IT1=zeros;
Rt1=ones;Rt2=ones;Rrho1=ones;Rrho2=ones;R2=zeros; IT2=zeros;
%Define Relaxation Factors
w=0.4; wp=0.7; wt=1; wrho=1;   % each represent momentum, pressure, temperature
```

```
% Monitor residual
disp('IT            Ru1          Ru2          Rv1          Rv2          Rp1          Rp2
Rt1          Rt2')

while Ru1 > 1e-06 | Ru2 > 1e-06 | Rv1 > 1e-06 | Rv2 > 1e-06 | Rp1 > 1e-06 | Rp2 > 1e-
06...
    | Rt1 > 1e-07 | Rt2 > 1e-07

    % Duct 1
    % Call u coefficient and solve u
    [us1]=u_coefficient1(us1,u1,v1,mu1);
    [u1]=imp_x_u(us1,u1,p1,w);
    [u1]=imp_y_u(us1,u1,p1,w);
    [u1]=imp_x_u(us1,u1,p1,w);
    [u1]=imp_y_u(us1,u1,p1,w);
    % Duct 2
    % Call u coefficient and solve u
    [us2]=u_coefficient1(us2,u2,v2,mu2);
    [u2]=imp_x_u(us2,u2,p2,w);
    [u2]=imp_y_u(us2,u2,p2,w);
    [u2]=imp_x_u(us2,u2,p2,w);
    [u2]=imp_y_u(us2,u2,p2,w);


    % Duct 1
    % Call v coefficient and solve v
    [vs1]=v_coefficient1(vs1,u1,v1,mu1);
    [v1]=imp_x_v(vs1,v1,p1,w);
    [v1]=imp_y_v(vs1,v1,p1,w);
    [v1]=imp_x_v(vs1,v1,p1,w);
    [v1]=imp_y_v(vs1,v1,p1,w);
    % Duct 2
     % Call v coefficient and solve v
    [vs2]=v_coefficient1(vs2,u2,v2,mu2);
    [v2]=imp_x_v(vs2,v2,p2,w);
    [v2]=imp_y_v(vs2,v2,p2,w);
    [v2]=imp_x_v(vs2,v2,p2,w);
    [v2]=imp_y_v(vs2,v2,p2,w);


    % Call pc coefficient and solve pc
    % Duct 1
    [ps1]=p_coefficient(ps1,us1,vs1); pc1=zeros(nx+2,ny+2);
    [pc1]=imp_x_p(ps1,u1,v1,pc1);
    [pc1]=imp_y_p(ps1,u1,v1,pc1);
    [pc1]=imp_x_p(ps1,u1,v1,pc1);
    [pc1]=imp_y_p(ps1,u1,v1,pc1);
    % Duct 2
    [ps2]=p_coefficient(ps2,us2,vs2); pc2=zeros(nx+2,ny+2);
    [pc2]=imp_x_p(ps2,u2,v2,pc2);
    [pc2]=imp_y_p(ps2,u2,v2,pc2);
    [pc2]=imp_x_p(ps2,u2,v2,pc2);
    [pc2]=imp_y_p(ps2,u2,v2,pc2);


    % Calculate p
    % Duct 1
    p1=p1+wp*pc1;
    % Duct 2
    p2=p2+wp*pc2;

    % Calculate the velocity correction
    % Duct 1
    [u1,v1]=vel_correction(us1,vs1,u1,v1,pc1);
    % Duct 1
    [u2,v2]=vel_correction(us2,vs2,u2,v2,pc2);
```

```
    % Mass continuity
    % Duct 1
    min=rho*u1(1,:)*(ly/ny);
    mout=rho*u1(end-1,:)*(ly/ny);
    min=sum(min);
    mout=sum(mout);
    mrat=min/mout;
    u1(end,:)=mrat*u1(end-1,:);
    % Duct 2
    min=rho*u2(1,:)*(ly/ny);
    mout=rho*u2(end-1,:)*(ly/ny);
    min=sum(min);
    mout=sum(mout);
    mrat=min/mout;
    u2(end,:)=mrat*u2(end-1,:);

    % Call T coefficient and solve T
    [ts1]=t_coefficient1(ts1,u1,v1,T1,k_vec1);
    [T1]=imp_x_t1(ts1,T1,T2,wt);
%   [T1]=imp_y_t1(ts1,T1,T2,wt);
    [T1]=imp_x_t1(ts1,T1,T2,wt);
%   [T1]=imp_y_t1(ts1,T1,T2,wt);
    [ts2]=t_coefficient2(ts2,u2,v2,T2,k_vec2);
    [T2]=imp_x_t2(ts2,T2,T1,wt);
%   [T2]=imp_y_t2(ts2,T2,T1,wt);
    [T2]=imp_x_t2(ts2,T2,T1,wt);
%   [T2]=imp_y_t2(ts2,T2,T1,wt);


    % Calculate Residual

    IT1=IT1+1; % Count iteration
    [Ru1,Rv1,Rp1,Rt1,R1]=residuals(us1,vs1,ts1,u1,v1,p1,T1,IT1,R1);
    IT2=IT2+1;
    [Ru2,Rv2,Rp2,Rt2,R2]=residuals(us2,vs2,ts2,u2,v2,p2,T2,IT2,R2);
%
    % Monitor residual
    fprintf('%d      %.4e     %.4e     %.4e     %.4e     %.4e     %.4e     %.4e
%.4e\n',IT1,Ru1,Ru2,Rv1,Rv2,Rp1,Rp2,Rt1,Rt2)

end

% Calculate nodal points velocities
[uc1,vc1] = postprocess(us1,vs1,u1,v1);
[uc2,vc2] = postprocess(us2,vs2,u2,v2);

% Temperature Plots
[T_bulk1,rho_bulk1,Nu1]=Tbulk1(T1,rho1,ts1,rhos1,uc1);
[T_bulk2,Nu2]=Tbulk2(T2,ts2,uc2);
effec=(T_bulk1(end)-T_bulk1(1))/(T_bulk2(1)-T_bulk1(1));
figure(3)
plot(ts1.x,T_bulk1,ts1.x,fliplr(T_bulk2),'LineWidth',2)
xlabel('Channel length, L [m]','FontSize',10)
ylabel('T bulk[K]','FontSize',10)

% Velocty vector plots
%Duct 1
figure(4)
scale=0.18;
quiver(ps1.x,ps1.y,uc1,vc1,scale)
xlabel('Channel length, L [m]','FontSize',10)
ylabel('Channel Height, H [m]','FontSize',10)
axis([0.15 0.20 0 0.004]);
%Duct 2
figure(5)
```

```
uc2=flipud(uc2);
vc2=flipud(vc2);
quiver(ps2.x,ps2.y,-uc2,vc2,scale)
xlabel('Channel length, L [m]','FontSize',10)
ylabel('Channel Height, H [m]','FontSize',10)
axis([0.15 0.20 0 0.004]);

% Temperature Countour
%Duct 1
figure(6)
 contourf(ps1.x,ps1.y,T1,500)
 caxis([298,311])
 xlabel('Channel length, L [m]','FontSize',10)
 ylabel('Channel Height, H [m]','FontSize',10)
 c=colorbar;
 ylabel(c,'Temperature [K]','FontSize',10)
 %Duct 2
figure(7)
 Tc2=flipud(T2);
 contourf(ps2.x,ps2.y,Tc2,500)
 caxis([297,313])
 xlabel('Channel length, L [m]','FontSize',10)
 ylabel('Channel Height, H [m]','FontSize',10)
 c=colorbar;
 ylabel(c,'Temperature [K]','FontSize',10)

%Pressure drop
%Duct 1
figure(8)
 plot(ps1.x(:,ny/2),p1(:,ny/2),'LineWidth',2)
xlabel('Channel length, L [m]','FontSize',10)
ylabel('Pressure (Pa)','FontSize',10)


time=toc; toc

% Heat transfer effectiveness
figure(10)
xx=[0.5;1.5;2.5];
yy=[55.12,67.87,69.68]; % Values were added manually after simulation
xxx=linspace(0.5,2.5,10);
plot(xxx,csapi(xx,yy,xxx),xx,yy,'ro','LineWidth',2)
xlabel('S/Dh','FontSize',10)
ylabel('Effectiveness(%)','FontSize',10)

% Pressure drop x baffle height
figure(11)
xx=[0.25;0.35;0.45];
yy=[27.54,82.15,155.31]; % Values were added manually after simulation
xxx=linspace(0.25,0.45,10);
plot(xxx,csapi(xx,yy,xxx),xx,yy,'ro','LineWidth',2)
xlabel('h/Dh','FontSize',10)
ylabel('Pressure Drop (Pa)','FontSize',10)

% Nusselt x baffle height
figure(12)
xx=[0.25;0.35;0.45];
yy=[19.15,29.2,50.1]; % Values were added manually after simulation
xxx=linspace(0.25,0.45,10);
plot(xxx,csapi(xx,yy,xxx),xx,yy,'ro','LineWidth',2)
xlabel('h/Dh','FontSize',10)
ylabel('Nusselt','FontSize',10)

% Pressure drop x baffle spacing
figure(13)
xx=[0.50;1.50;2.0;2.50];
```

```
yy=[137.1,123.31,82.14,57.41]; % Values were added manually after simulation
xxx=linspace(0.50,2.50,10);
plot(xxx,csapi(xx,yy,xxx),xx,yy,'ro','LineWidth',2)
xlabel('S/Dh','FontSize',10)
ylabel('Pressure Drop (Pa)','FontSize',10)

% Nusselt number x baffle spacing
figure(14)
xx=[0.50;1.50;2.50];
yy=[9.4,30.1,35.2]; % Values were added manually after simulation
xxx=linspace(0.50,2.50,10);
plot(xxx,csapi(xx,yy,xxx),xx,yy,'ro','LineWidth',2)
xlabel('S/Dh','FontSize',10)
ylabel('Nusselt','FontSize',10)

% Pressure drop x Re
figure(15)
xx=[150;200;300;400];
yy=[57.45,101.81,231.28,415.06];
xxx=linspace(150,400,10);
plot(xxx,csapi(xx,yy,xxx),xx,yy,'ro','LineWidth',2)
xlabel('Re','FontSize',10)
ylabel('Pressure Drop (Pa)','FontSize',10)

% Nusselt number x Re
figure(16)
xx=[200;300;400];
yy=[41.2,53.6,63.9];
xxx=linspace(200,400,10);
plot(xxx,csapi(xx,yy,xxx),xx,yy,'ro','LineWidth',2)
xlabel('Re','FontSize',10)
ylabel('Nusselt','FontSize',10)

% Heat transfer effectiveness x Re
figure(17)
xx=[200;300;400];
yy=[66.38,62.22,59.40];
xxx=linspace(200,400,10);
plot(xxx,csapi(xx,yy,xxx),xx,yy,'ro','LineWidth',2)
xlabel('Re','FontSize',10)
ylabel('Effectiveness(%)','FontSize',10)
```

```
% Geometry Function

function [ss] = geometry(ss,verbose)

% call global variables
global lx ly nx ny

if nargin < 2, xmax=nx+2; ymax=ny+2;                % p control volumes
elseif verbose < 1, xmax=nx+1; ymax=ny+2;           % u control volumes
else xmax=nx+2; ymax=ny+1;                          % v control volumes
end

% Initial variables
x=zeros(xmax,ymax); y=zeros(xmax,ymax);
dx=zeros(xmax-1,ymax); dy=zeros(xmax,ymax-1);

% Grid distribution
if nargin < 2, dx(1,:)=lx/nx/2; dy(:,1)=ly/ny/2;    % p control volumes
elseif verbose < 1, dx(1,:)=lx/nx; dy(:,1)=ly/ny/2; % u control volumes
else dx(1,:)=lx/nx/2; dy(:,1)=ly/ny;                % v control volumes
end

% last points
```

```
dx(xmax-1,:)=dx(1,:);
dy(:,ymax-1)=dy(:,1);

% diffusion length
dx(2:xmax-2,:)=lx/nx;
dy(:,2:ymax-2)=ly/ny;

% Boundary
for i=2:xmax, for j=1:ymax-1, x(i,j)=x(i-1,j)+dx(i-1,j); end, end
for i=1:xmax-1, for j=2:ymax, y(i,j)=y(i,j-1)+dy(i,j-1); end, end
x(:,ymax)=x(:,ymax-1); y(xmax,:)=y(xmax-1,:);


ss.x=x; ss.y=y; ss.dx=dx; ss.dy=dy;

end
```

```
% Boundary Condition Function

function [p1,p2,pc1,pc2,u1,u2,v1,v2,T1,T2] = bcic()

% Call global variables
global ly nx ny umax

% Pressure and velocity vectors

u1=zeros(nx+1,ny+2);
u2=zeros(nx+1,ny+2);
v1=zeros(nx+2,ny+1);
v2=zeros(nx+2,ny+1);
p1=zeros(nx+2,ny+2);
p2=zeros(nx+2,ny+2);
pc1=zeros(nx+2,ny+2);
pc2=zeros(nx+2,ny+2);
T1=zeros(nx+2,ny+2);
T2=zeros(nx+2,ny+2);

% BC for pressure and velocities
% West
u1(1,2:end) = umax;
u2(1,2:end) = umax;
v1(1,:)=0;
v2(1,:)=0;
T1(1,:) =298;
T2(1,:) =313;
% east
u1(nx+1,:)=0;
u2(nx+1,:)=0;
v1(nx+2,:)=0;
v2(nx+2,:)=0;
T1(nx+2,:)=0;
T2(nx+2,:)=0;
% north
u1(:,ny+2)=0;
u2(:,ny+2)=0;
v1(:,ny+1)=0;
v2(:,ny+1)=0;
T1(2:end-1,ny+2)=0;
T2(2:end-1,ny+2)=0;
% south
u1(:,1)=0;
u2(:,1)=0;
v1(:,1)=0;v(:,1)=0;
T1(2:end-1,1)=0;
T2(2:end-1,1)=0;
```

```
end
```

```
% Blockage condition in baffles

function [mu,k_vec,D_vec]=blockage(mu)

% Call global variables
global k lx ly nx ny EntL h Th Pit ExiL D

% Define block as high vicosity region
if isvector(mu)
    % initialize viscosity
    mu=mu*ones(nx+2,ny+2);
    k_vec=k*ones(nx+2,ny+2);
    D_vec=D*ones(nx+2,ny+2);


Mu1T=round(EntL/lx*nx)+round(Th/lx*nx+1); % Index of viscosity matrix
Mu0T=round(EntL/lx*nx+1);
Mu1B=round(EntL/lx*nx)+round(((Pit-(abs(Pit-Th)/2))+Th)/lx*nx+1);
Mu0B=round(EntL/lx*nx)+round((Pit-(abs(Pit-Th)/2))/lx*nx+1);
i=1;



while Mu1B<(nx+2-round(ExiL/lx*nx))

    mu(Mu0B+1:Mu1B,1:round(h/ly*ny+1))=10^45;
    mu(Mu0T+1:Mu1T,(ny+2-round(h/ly*ny):end))=10^45;
    k_vec(Mu0B+1:Mu1B,1:round(h/ly*ny+1))=0;
    k_vec(Mu0T+1:Mu1T,(ny+2-round(h/ly*ny):end))=0;
    D_vec(Mu0B+1:Mu1B,1:round(h/ly*ny+1))=0;
    D_vec(Mu0T+1:Mu1T,(ny+2-round(h/ly*ny):end))=0;

    Mu0T=round(EntL/lx*nx)+round((Th+Pit)*i/lx*nx+1);
    Mu1T=round(EntL/lx*nx)+round((Th+((Th+Pit)*i))/lx*nx+1);
    Mu1B=round(EntL/lx*nx)+round(((Pit-(abs(Pit-Th)/2))+Th+((Th+Pit)*i))/lx*nx+1);
    Mu0B=round(EntL/lx*nx)+round(((Pit-(abs(Pit-Th)/2))+((Th+Pit)*i))/lx*nx+1);


    i=i+1;
end

elseif ~isvector(mu)

Mu1T=round(EntL/lx*nx)+round(Th/lx*nx+1); % Index of viscosity matrix
Mu0T=round(EntL/lx*nx+1);
Mu1B=round(EntL/lx*nx)+round(((Pit-(abs(Pit-Th)/2))+Th)/lx*nx+1);
Mu0B=round(EntL/lx*nx)+round((Pit-(abs(Pit-Th)/2))/lx*nx+1);
i=1;



while Mu1B<(nx+2-round(ExiL/lx*nx))

    mu(Mu0B+1:Mu1B,1:round(h/ly*ny+1))=10^45;
    mu(Mu0T+1:Mu1T,(ny+2-round(h/ly*ny):end))=10^45;

    Mu0T=round(EntL/lx*nx)+round((Th+Pit)*i/lx*nx+1);
    Mu1T=round(EntL/lx*nx)+round((Th+((Th+Pit)*i))/lx*nx+1);
    Mu1B=round(EntL/lx*nx)+round(((Pit-(abs(Pit-Th)/2))+Th+((Th+Pit)*i))/lx*nx+1);
    Mu0B=round(EntL/lx*nx)+round(((Pit-(abs(Pit-Th)/2))+((Th+Pit)*i))/lx*nx+1);
```

```
    i=i+1;
end


end
end
```

```
% U velocity Coefficients
function [us]=u_coefficient1(us,u,v,mu)

% define maximum points
xmax=size(u,1); ymax=size(u,2);

% Initialize coefficients
aw=zeros(xmax,ymax);
ae=aw;
as=aw;
an=aw;
ap=aw;
Dw=aw;
De=aw;
Ds=aw;
Dn=aw;
Fw=aw;
Fe=aw;
Fs=aw;
Fn=aw;
Pw=aw;
Pe=aw;
Ps=aw;
Pn=aw;

% Call global variables
global rho lx ly nx ny

%Difussion
for j=2:ymax-1
    for i=2:xmax-1
        switch i
            case 2          % 3/2 control volumes
                Dw(i,j)=mu(i-1,j)*ly/ny/us.dx(i-1,j);
                De(i,j)=mu(i+1,j)*ly/ny/us.dx(i,j);
                Ds(i,j)=harmmean([mu(i,j-1) mu(i+1,j-1)])*...
                    lx/nx*1.5/us.dy(i,j-1);
                Dn(i,j)=harmmean([mu(i,j) mu(i+1,j)])*lx/nx*1.5/us.dy(i,j);
            case xmax-1     % 3/2 control volumes
                Dw(i,j)=mu(i,j)*ly/ny/us.dx(i-1,j);
                De(i,j)=mu(i+2,j)*ly/ny/us.dx(i,j);
                Ds(i,j)=harmmean([mu(i,j-1) mu(i+1,j-1)])*...
                    lx/nx*1.5/us.dy(i,j-1);
                Dn(i,j)=harmmean([mu(i,j) mu(i+1,j)])*lx/nx*1.5/us.dy(i,j);
            otherwise       % regular control volumes
                Dw(i,j)=mu(i,j)*ly/ny/us.dx(i-1,j);
                De(i,j)=mu(i+1,j)*ly/ny/us.dx(i,j);
                Ds(i,j)=harmmean([mu(i,j-1) mu(i+1,j-1)])*...
                    lx/nx/us.dy(i,j-1);
                Dn(i,j)=harmmean([mu(i,j) mu(i+1,j)])*lx/nx/us.dy(i,j);
        end
    end
end

% Convection
for j=2:ymax-1
    for i=2:xmax-1
        switch i
```

```
            case 2          % west 3/2 control volumes
                Fw(i,j)=rho*u(i-1,j)*ly/ny;
                Fe(i,j)=rho*(u(i,j)+u(i+1,j))/2*ly/ny;
                Fs(i,j)=rho*(v(i-1,j-1)+3*v(i,j-1)+2*v(i+1,j-1))/4*lx/nx;
                Fn(i,j)=rho*(v(i-1,j)+3*v(i,j)+2*v(i+1,j))/4*lx/nx;
            case xmax-1     % east 3/2 control volumes
                Fw(i,j)=rho*(u(i-1,j)+u(i,j))/2*ly/ny;
                Fe(i,j)=rho*u(i+1,j)*ly/ny;
                Fs(i,j)=rho*(2*v(i,j-1)+3*v(i+1,j-1)+v(i+2,j-1))/4*lx/nx;
                Fn(i,j)=rho*(2*v(i,j)+3*v(i+1,j)+v(i+2,j))/4*lx/nx;
            otherwise       % inside control volumes
                Fw(i,j)=rho*(u(i-1,j)+u(i,j))/2*ly/ny;
                Fe(i,j)=rho*(u(i,j)+u(i+1,j))/2*ly/ny;
                Fs(i,j)=rho*(v(i,j-1)+v(i+1,j-1))/2*lx/nx;
                Fn(i,j)=rho*(v(i,j)+v(i+1,j))/2*lx/nx;
        end
    end
end

% Peclet numbers
Pw(2:xmax-1,2:ymax-1)=Fw(2:xmax-1,2:ymax-1)./Dw(2:xmax-1,2:ymax-1);
Pe(2:xmax-1,2:ymax-1)=Fe(2:xmax-1,2:ymax-1)./De(2:xmax-1,2:ymax-1);
Ps(2:xmax-1,2:ymax-1)=Fs(2:xmax-1,2:ymax-1)./Ds(2:xmax-1,2:ymax-1);
Pn(2:xmax-1,2:ymax-1)=Fn(2:xmax-1,2:ymax-1)./Dn(2:xmax-1,2:ymax-1);

% Coefficients
for i=2:xmax-1
    for j=2:ymax-1
        aw(i,j)=Dw(i,j)*powerlaw(Pw(i,j))+max(Fw(i,j),0);
        ae(i,j)=De(i,j)*powerlaw(Pe(i,j))+max(-Fe(i,j),0);
        as(i,j)=Ds(i,j)*powerlaw(Ps(i,j))+max(Fs(i,j),0);
        an(i,j)=Dn(i,j)*powerlaw(Pn(i,j))+max(-Fn(i,j),0);
    end
end
ap=aw+ae+as+an;


us.aW=aw; us.aE=ae; us.aS=as; us.aN=an; us.aP=ap;

end
```

```
% U solve implicit x

function [u]=imp_x_u(us,u,p,w)

xmax=size(u,1);
ymax=size(u,2);


% call global variables
global ly ny


aw=us.aW;
ae=us.aE;
as=us.aS;
an=us.aN;
ap=us.aP;

for j=ymax-1:-1:2
    % TDMA variables
```

```
    a=zeros(1,xmax-2);
    b=zeros(1,xmax-2);
    c=zeros(1,xmax-3);
    d=zeros(1,xmax-2);
    % TDMA Boundary Conditions
    b(1)=ap(2,j)/w; c(1)=-ae(2,j);
    d(1)=as(2,j)*u(2,j-1)+an(2,j)*u(2,j+1)+ap(2,j)*u(2,j)*(1/w-1)+...
        aw(2,j)*u(1,j)+(p(2,j)-p(3,j))*ly/ny;
    a(xmax-2)=-aw(xmax-1,j); b(xmax-2)=ap(xmax-1,j)/w-ae(xmax-1,j);
    d(xmax-2)=as(xmax-1,j)*u(xmax-1,j-1)+an(xmax-1,j)*u(xmax-1,j+1)+...
        ap(xmax-1,j)*u(xmax-1,j)*(1/w-1)+(p(xmax-1,j)-p(xmax,j))*ly/ny;
    for i=2:xmax-3
        a(i)= -aw(i+1,j);
        b(i)=ap(i+1,j)/w;
        c(i)= -ae(i+1,j);
        d(i)=as(i+1,j)*u(i+1,j-1)+an(i+1,j)*u(i+1,j+1)+...
            ap(i+1,j)*u(i+1,j)*(1/w-1)+(p(i+1,j)-p(i+2,j))*ly/ny;
    end
    % Call TDMA function
    [u(2:xmax-1,j)] = TDMA(a,b,c,d);
    % Boundary condition: du/dx = 0
    u(xmax,j)=u(xmax-1,j);
end
return;
```

```
% U solve implicit y

function [u]=imp_y_u(us,u,p,w)

xmax=size(u,1);
ymax=size(u,2);

% call global variables
global ly ny


aw=us.aW;
ae=us.aE;
as=us.aS;
an=us.aN;
ap=us.aP;

for i=2:1:xmax-1
    % TDMA variables
    a=zeros(1,ymax-2); b=zeros(1,ymax-2);
    c=zeros(1,ymax-3); d=zeros(1,ymax-2);
     % TDMA Boundary Conditions
    b(1)=ap(i,2)/w; c(1)=-an(i,2);
    d(1)=aw(i,2)*u(i-1,2)+ae(i,2)*u(i+1,2)+ap(i,2)*u(i,2)*(1/w-1)+...
        as(i,2)*u(i,1)+(p(i,2)-p(i+1,2))*ly/ny;
    a(ymax-2)=-as(i,ymax-1); b(ymax-2)=ap(i,ymax-1)/w;
    d(ymax-2)=aw(i,ymax-1)*u(i-1,ymax-1)+ae(i,ymax-1)*u(i+1,ymax-1)+...
        ap(i,ymax-1)*u(i,ymax-1)*(1/w-1)+an(i,ymax-1)*u(i,ymax)+...
        (p(i,ymax-1)-p(i+1,ymax-1))*ly/ny;
    for j=2:ymax-3
        a(j)= -as(i,j+1);
        b(j)=ap(i,j+1)/w;
        c(j)= -an(i,j+1);
        d(j)=aw(i,j+1)*u(i-1,j+1)+ae(i,j+1)*u(i+1,j+1)+...
            ap(i,j+1)*u(i,j+1)*(1/w-1)+(p(i,j+1)-p(i+1,j+1))*ly/ny;
    end
    % Call TDMA function
    [u(i,2:ymax-1)] = TDMA(a,b,c,d);
end
% Apply boundary condition
```

```
u(xmax,:)=u(xmax-1,:);
return;
```

```
% V velocity Coefficients

function [vs]=v_coefficient1(vs,u,v,mu)

xmax=size(v,1);
ymax=size(v,2);

% Initialize coefficients
aw=zeros(xmax,ymax);
ae=aw;
as=aw;
aN=aw;
ap=aw;
Dw=aw;
De=aw;
Ds=aw;
Dn=aw;
Fw=aw;
Fe=aw;
Fs=aw;
Fn=aw;
Pw=aw;
Pe=aw;
Ps=aw;
Pn=aw;

% Call global variables
global rho lx ly nx ny

%Difussion
for j=2:ymax-1
    for i=2:xmax-1
        switch j
            case 2      % 3/2 control volumes
                Ds(i,j)=mu(i,j-1)*lx/nx/vs.dy(i,j-1);
                Dn(i,j)=mu(i,j+1)*lx/nx/vs.dy(i,j);
                Dw(i,j)=harmmean([mu(i-1,j) mu(i,j)])*...
                    ly/ny*1.5/vs.dx(i-1,j);
                De(i,j)=harmmean([mu(i,j) mu(i+1,j)])*ly/ny*1.5/vs.dx(i,j);
            case ymax-1    % 3/2 control volumes
                Ds(i,j)=mu(i,j)*lx/nx/vs.dy(i,j-1);
                Dn(i,j)=mu(i,j+2)*lx/nx/vs.dy(i,j);
                Dw(i,j)=harmmean([mu(i-1,j) mu(i,j)])*...
                    ly/ny*1.5/vs.dx(i-1,j);
                De(i,j)=harmmean([mu(i,j) mu(i+1,j)])*ly/ny*1.5/vs.dx(i,j);
            otherwise       % inside control volumes
                Ds(i,j)=mu(i,j)*lx/nx/vs.dy(i,j-1);
                Dn(i,j)=mu(i,j+1)*lx/nx/vs.dy(i,j);
                Dw(i,j)=harmmean([mu(i-1,j) mu(i,j)])*ly/ny/vs.dx(i-1,j);
                De(i,j)=harmmean([mu(i,j) mu(i+1,j)])*ly/ny/vs.dx(i,j);
        end
    end
end

% Convection
for j=2:ymax-1
    for i=2:xmax-1
        switch j
            case 2          % south 2/3 control volumes
                Fw(i,j)=rho*(u(i-1,j-1)+3*u(i-1,j)+2*u(i-1,j+1))/4*ly/ny;
                Fe(i,j)=rho*(u(i,j-1)+3*u(i,j)+2*u(i,j+1))/4*ly/ny;
                Fs(i,j)=rho*v(i,j-1)*lx/nx;
```

```
                Fn(i,j)=rho*(v(i,j)+v(i,j+1))/2*lx/nx;
            case ymax-1     % north 2/3 control volumes
                Fw(i,j)=rho*(2*u(i-1,j)+3*u(i-1,j+1)+u(i-1,j+2))/4*ly/ny;
                Fe(i,j)=rho*(2*u(i,j)+3*u(i,j+1)+u(i,j+2))/4*ly/ny;
                Fs(i,j)=rho*(v(i,j-1)+v(i,j))/2*lx/nx;
                Fn(i,j)=rho*v(i,j+1)*lx/nx;
            otherwise       % inside control volumes
                Fw(i,j)=rho*(u(i-1,j)+u(i-1,j+1))/2*ly/ny;
                Fe(i,j)=rho*(u(i,j)+u(i,j+1))/2*ly/ny;
                Fs(i,j)=rho*(v(i,j-1)+v(i,j))/2*lx/nx;
                Fn(i,j)=rho*(v(i,j)+v(i,j+1))/2*lx/nx;
        end
    end
end

% Peclet numbers
Pw(2:xmax-1,2:ymax-1)=Fw(2:xmax-1,2:ymax-1)./Dw(2:xmax-1,2:ymax-1);
Pe(2:xmax-1,2:ymax-1)=Fe(2:xmax-1,2:ymax-1)./De(2:xmax-1,2:ymax-1);
Ps(2:xmax-1,2:ymax-1)=Fs(2:xmax-1,2:ymax-1)./Ds(2:xmax-1,2:ymax-1);
Pn(2:xmax-1,2:ymax-1)=Fn(2:xmax-1,2:ymax-1)./Dn(2:xmax-1,2:ymax-1);

% Coefficients
for i=2:xmax-1
    for j=2:ymax-1
        aw(i,j)=Dw(i,j)*powerlaw(Pw(i,j))+max(Fw(i,j),0);
        ae(i,j)=De(i,j)*powerlaw(Pe(i,j))+max(-Fe(i,j),0);
        as(i,j)=Ds(i,j)*powerlaw(Ps(i,j))+max(Fs(i,j),0);
        aN(i,j)=Dn(i,j)*powerlaw(Pn(i,j))+max(-Fn(i,j),0);
    end
end
ap=aw+ae+as+aN;

vs.aW=aw; vs.aE=ae; vs.aS=as; vs.aN=aN; vs.aP=ap;

end
```

```
% V solve implicit x

function [v]=imp_x_v(vs,v,p,w)


xmax=size(v,1);
ymax=size(v,2);

% call global variables
global lx nx

% allocate structure array
aw=vs.aW;
ae=vs.aE;
as=vs.aS;
an=vs.aN;
ap=vs.aP;

for j=ymax-1:-1:2
    % TDMA variables
    a=zeros(1,xmax-2);
    b=zeros(1,xmax-2);
    c=zeros(1,xmax-3);
    d=zeros(1,xmax-2);
     % TDMA Boundary Conditions
    b(1)=ap(2,j)/w; c(1)=-ae(2,j);
    d(1)=as(2,j)*v(2,j-1)+an(2,j)*v(2,j+1)+ap(2,j)*v(2,j)*(1/w-1)+...
        aw(2,j)*v(1,j)+(p(2,j)-p(2,j+1))*lx/nx;
    a(xmax-2)=-aw(xmax-1,j); b(xmax-2)=ap(xmax-1,j)/w;
```

```
    d(xmax-2)=as(xmax-1,j)*v(xmax-1,j-1)+an(xmax-1,j)*v(xmax-1,j+1)+...
        ap(xmax-1,j)*v(xmax-1,j)*(1/w-1)+ae(xmax-1,j)*v(xmax,j)+...
        (p(xmax-1,j)-p(xmax-1,j+1))*lx/nx;
    for i=2:xmax-3
        a(i)= -aw(i+1,j);
        b(i)=ap(i+1,j)/w;
        c(i)= -ae(i+1,j);
        d(i)=as(i+1,j)*v(i+1,j-1)+an(i+1,j)*v(i+1,j+1)+...
            ap(i+1,j)*v(i+1,j)*(1/w-1)+(p(i+1,j)-p(i+1,j+1))*lx/nx;
    end
    % Call TDMA function
    [v(2:xmax-1,j)] = TDMA(a,b,c,d);

    % Apply boundary condition: fully developed flow
    v(xmax,j)=0;
end
return;
```

```
% V solve implicit y

function [v]=imp_y_v(vs,v,p,w)


xmax=size(v,1);
ymax=size(v,2);


% call global variables
global lx nx

aw=vs.aW;
ae=vs.aE;
as=vs.aS;
an=vs.aN;
ap=vs.aP;

for i=xmax-1:-1:2
    % TDMA variables
    a=zeros(1,ymax-2); b=zeros(1,ymax-2);
    c=zeros(1,ymax-3); d=zeros(1,ymax-2);
    % TDMA Boundary Conditions
    b(1)=ap(i,2)/w; c(1)=-an(i,2);
    d(1)=aw(i,2)*v(i-1,2)+ae(i,2)*v(i+1,2)+ap(i,2)*v(i,2)*(1/w-1)+...
        as(i,2)*v(i,1)+(p(i,2)-p(i,3))*lx/nx;
    a(ymax-2)=-as(i,ymax-1); b(ymax-2)=ap(i,ymax-1)/w;
    d(ymax-2)=aw(i,ymax-1)*v(i-1,ymax-1)+ae(i,ymax-1)*v(i+1,ymax-1)+...
        ap(i,ymax-1)*v(i,ymax-1)*(1/w-1)+an(i,ymax-1)*v(i,ymax)+...
        (p(i,ymax-1)-p(i,ymax))*lx/nx;
    for j=2:ymax-3
        a(j)= -as(i,j+1);
        b(j)=ap(i,j+1)/w;
        c(j)= -an(i,j+1);
        d(j)=aw(i,j+1)*v(i-1,j+1)+ae(i,j+1)*v(i+1,j+1)+...
            ap(i,j+1)*v(i,j+1)*(1/w-1)+(p(i,j+1)-p(i,j+2))*lx/nx;
    end
    % Call TDMA function
    [v(i,2:ymax-1)] = TDMA(a,b,c,d);
end
% Apply boundary condition: fully developed flow
v(xmax,:)=0;
return;
```

```
% P Pressure Coefficients

function [ps]=p_coefficient(ps,us,vs)

xmax=size(ps.x,1);
ymax=size(ps.y,2);


aw=zeros(xmax,ymax);
ae=aw;
as=aw;
an=aw;
ap=aw;

% Call global variables
global rho lx ly nx ny


for i=2:xmax-1
    for j=2:ymax-1
        aw(i,j)=rho*ly/ny/us.aP(i-1,j)*ly/ny;
        ae(i,j)=rho*ly/ny/us.aP(i,j)*ly/ny;
        as(i,j)=rho*lx/nx/vs.aP(i,j-1)*lx/nx;
        an(i,j)=rho*lx/nx/vs.aP(i,j)*lx/nx;
    end
end

% BC
aw(2,:)=0; ae(xmax-1,:)=0; as(:,2)=0; an(:,ymax-1)=0;

ap=aw+ae+as+an;

ps.aW=aw; ps.aE=ae; ps.aS=as; ps.aN=an; ps.aP=ap;

end
```

```
% P solve implicit x

function [pc]=imp_x_p(ps,u,v,pc)


xmax=size(pc,1);
ymax=size(pc,2);

% call global variables
global rho lx ly nx ny

aw=ps.aW;
ae=ps.aE;
as=ps.aS;
an=ps.aN;
ap=ps.aP;

for j=ymax-1:-1:2
     % TDMA variabless
    a=zeros(1,xmax-2);
    b=zeros(1,xmax-2);
    c=zeros(1,xmax-3);
    d=zeros(1,xmax-2);
   % TDMA Boundary Conditions
    b(1)=ap(2,j)-3/2*aw(2,j); c(1)=-ae(2,j)+0.5*aw(2,j);
    d(1)=as(2,j)*pc(2,j-1)+an(2,j)*pc(2,j+1)+rho*(u(1,j)-u(2,j))*ly/ny+...
        rho*(v(2,j-1)-v(2,j))*lx/nx;
    a(xmax-2)=-aw(xmax-1,j)+0.5*ae(xmax-1,j);
```

```
    b(xmax-2)=ap(xmax-1,j)-3/2*ae(xmax-1,j);
    d(xmax-2)=as(xmax-1,j)*pc(xmax-1,j-1)+an(xmax-1,j)*pc(xmax-1,j+1)+...
        rho*(u(xmax-2,j)-u(xmax-1,j))*ly/ny+...
        rho*(v(xmax-1,j-1)-v(xmax-1,j))*lx/nx;
    for i=2:xmax-3
        a(i)= -aw(i+1,j);
        b(i)=ap(i+1,j);
        c(i)= -ae(i+1,j);
        d(i)=as(i+1,j)*pc(i+1,j-1)+an(i+1,j)*pc(i+1,j+1)+...
            rho*(u(i,j)-u(i+1,j))*ly/ny+rho*(v(i+1,j-1)-v(i+1,j))*lx/nx;
    end
    % Call TDMA function
    [pc(2:xmax-1,j)] = TDMA(a,b,c,d);
    % dp/dx=constant
    pc(1,j)=3/2*pc(2,j)-0.5*pc(3,j);                % west
    pc(xmax,j)=3/2*pc(xmax-1,j)-0.5*pc(xmax-2,j);   % east
end
return;
```

```
% P solve implicit y

function [pc]=imp_y_p(ps,u,v,pc)

xmax=size(pc,1);
ymax=size(pc,2);

% call global variables
global rho lx ly nx ny

% allocate structure array

aw=ps.aW;
ae=ps.aE;
as=ps.aS;
an=ps.aN;
ap=ps.aP;

for i=xmax-1:-1:2
    % TDMA variables
    a=zeros(1,ymax-2); b=zeros(1,ymax-2);
    c=zeros(1,ymax-3); d=zeros(1,ymax-2);
   % TDMA Boundary Conditions
    b(1)=ap(i,2); c(1)=-an(i,2);
    d(1)=aw(i,2)*pc(i-1,2)+ae(i,2)*pc(i+1,2)+as(i,2)*pc(i,1)+...
        rho*(u(i-1,2)-u(i,2))*ly/ny+rho*(v(i,1)-v(i,2))*lx/nx;
    a(ymax-2)=-as(i,ymax-1); b(ymax-2)=ap(i,ymax-1);
    d(ymax-2)=aw(i,ymax-1)*pc(i-1,ymax-1)+ae(i,ymax-1)*pc(i+1,ymax-1)+...
        +an(i,ymax-1)*pc(i,ymax)+rho*(u(i-1,ymax-1)-u(i,ymax-1))*ly/ny+...
        rho*(v(i,ymax-2)-v(i,ymax-1))*lx/nx;
    for j=2:ymax-3
        a(j)= -as(i,j+1);
        b(j)=ap(i,j+1);
        c(j)= -an(i,j+1);
        d(j)=aw(i,j+1)*pc(i-1,j+1)+ae(i,j+1)*pc(i+1,j+1)+...
            rho*(u(i-1,j+1)-u(i,j+1))*ly/ny+rho*(v(i,j)-v(i,j+1))*lx/nx;
    end
    % Call TDMA function
    [pc(i,2:ymax-1)] = TDMA(a,b,c,d);
end
% dp/dx=constant
pc(1,:)=3/2*pc(2,:)-0.5*pc(3,:);                % west
pc(xmax,:)=3/2*pc(xmax-1,:)-0.5*pc(xmax-2,:);   % east
return;
```

```
% Velocity Correction

function [u,v]=vel_correction(us,vs,u,v,pc)

% Call global variables
global lx ly nx ny

% calculate velocity correction

% u velocity

for i=2:size(u,1)-1
    for j=2:size(u,2)-1
        u(i,j)=u(i,j)+ly/ny/us.aP(i,j)*(pc(i,j)-pc(i+1,j));
    end
end
% v velocity
for i=2:size(v,1)-1
    for j=2:size(v,2)-1
        v(i,j)=v(i,j)+lx/nx/vs.aP(i,j)*(pc(i,j)-pc(i,j+1));
    end
end

end
```

```
% T1 Temperature Coefficients Duct 1

function [ts]=t_coefficient1(ts,u,v,T,k_vec)


xmax=size(T,1);
ymax=size(T,2);

% Initialize coefficients

aw=zeros(xmax,ymax);
ae=aw;
as=aw;
an=aw;
ap=aw;
Dw=aw;
De=aw;
Ds=aw;
Dn=aw;
Fw=aw;
Fe=aw;
Fs=aw;
Fn=aw;
Pw=aw;
Pe=aw;
Ps=aw;
Pn=aw;

% Call global variables
global rho cp lx ly nx ny

%Difussion
for j=2:ymax-1
    for i=2:xmax-1
        Dw(i,j)=harmmean([k_vec(i-1,j) k_vec(i,j)])/cp*ly/ny/ts.dx(i-1,j);
        De(i,j)=harmmean([k_vec(i,j) k_vec(i+1,j)])/cp*ly/ny/ts.dx(i,j);
        Ds(i,j)=harmmean([k_vec(i,j-1) k_vec(i,j)])/cp*lx/nx/ts.dy(i,j-1);
        Dn(i,j)=harmmean([k_vec(i,j) k_vec(i,j+1)])/cp*lx/nx/ts.dy(i,j);
    end
end
```

```
% Convection
for j=2:ymax-1
    for i=2:xmax-1
        Fw(i,j)=rho*u(i-1,j)*ly/ny; Fe(i,j)=rho*u(i,j)*ly/ny;
        Fs(i,j)=rho*v(i,j-1)*lx/nx; Fn(i,j)=rho*v(i,j)*lx/nx;
    end
end

% Peclet numbers
Pw(2:xmax-1,2:ymax-1)=Fw(2:xmax-1,2:ymax-1)./Dw(2:xmax-1,2:ymax-1);
Pe(2:xmax-1,2:ymax-1)=Fe(2:xmax-1,2:ymax-1)./De(2:xmax-1,2:ymax-1);
Ps(2:xmax-1,2:ymax-1)=Fs(2:xmax-1,2:ymax-1)./Ds(2:xmax-1,2:ymax-1);
Pn(2:xmax-1,2:ymax-1)=Fn(2:xmax-1,2:ymax-1)./Dn(2:xmax-1,2:ymax-1);

% Coefficients
for i=2:xmax-1
    for j=2:ymax-1
        aw(i,j)=Dw(i,j)*powerlaw(Pw(i,j))+max(Fw(i,j),0);
        ae(i,j)=De(i,j)*powerlaw(Pe(i,j))+max(-Fe(i,j),0);
        as(i,j)=Ds(i,j)*powerlaw(Ps(i,j))+max(Fs(i,j),0);
        an(i,j)=Dn(i,j)*powerlaw(Pn(i,j))+max(-Fn(i,j),0);
    end
end

ap=aw+ae+as+an;

%Coefficients at the baffles

[ap]=blockage(ap);


ts.aW=aw; ts.aE=ae; ts.aS=as; ts.aN=an; ts.aP=ap;

end
```

```
% T2 Temperature Coefficients Duct 2

function [ts]=t_coefficient1(ts,u,v,T,k_vec)


xmax=size(T,1);
ymax=size(T,2);



aw=zeros(xmax,ymax);
ae=aw;
as=aw;
an=aw;
ap=aw;
ap2=aw;
Dw=aw;
De=aw;
Ds=aw;
Dn=aw;
Fw=aw;
Fe=aw;
Fs=aw;
Fn=aw;
Pw=aw;
Pe=aw;
Ps=aw;
Pn=aw;
```

```
% Call global variables
global rho cp lx ly nx ny

%Difussion
for j=2:ymax-1
    for i=2:xmax-1
        Dw(i,j)=harmmean([k_vec(i-1,j) k_vec(i,j)])/cp*ly/ny/ts.dx(i-1,j);
        De(i,j)=harmmean([k_vec(i,j) k_vec(i+1,j)])/cp*ly/ny/ts.dx(i,j);
        Ds(i,j)=harmmean([k_vec(i,j-1) k_vec(i,j)])/cp*lx/nx/ts.dy(i,j-1);
        Dn(i,j)=harmmean([k_vec(i,j) k_vec(i,j+1)])/cp*lx/nx/ts.dy(i,j);
    end
end

% Convection
for j=2:ymax-1
    for i=2:xmax-1
        Fw(i,j)=rho*u(i-1,j)*ly/ny; Fe(i,j)=rho*u(i,j)*ly/ny;
        Fs(i,j)=rho*v(i,j-1)*lx/nx; Fn(i,j)=rho*v(i,j)*lx/nx;
    end
end

% Peclet numbers
Pw(2:xmax-1,2:ymax-1)=Fw(2:xmax-1,2:ymax-1)./Dw(2:xmax-1,2:ymax-1);
Pe(2:xmax-1,2:ymax-1)=Fe(2:xmax-1,2:ymax-1)./De(2:xmax-1,2:ymax-1);
Ps(2:xmax-1,2:ymax-1)=Fs(2:xmax-1,2:ymax-1)./Ds(2:xmax-1,2:ymax-1);
Pn(2:xmax-1,2:ymax-1)=Fn(2:xmax-1,2:ymax-1)./Dn(2:xmax-1,2:ymax-1);

% Coefficients
for i=2:xmax-1
    for j=2:ymax-1
        aw(i,j)=Dw(i,j)*powerlaw(Pw(i,j))+max(Fw(i,j),0);
        ae(i,j)=De(i,j)*powerlaw(Pe(i,j))+max(-Fe(i,j),0);
        as(i,j)=Ds(i,j)*powerlaw(Ps(i,j))+max(Fs(i,j),0);
        an(i,j)=Dn(i,j)*powerlaw(Pn(i,j))+max(-Fn(i,j),0);
    end
end


ap=aw+ae+as+an;

%Coefficients at the baffles

[ap2]=blockage(ap2);
ap2=fliplr(ap2);
ap2=flipud(ap2);
ap=ap2+ap;



ts.aW=aw; ts.aE=ae; ts.aS=as; ts.aN=an; ts.aP=ap;

end
```

```
% T1 solve implicit x (Duct1)

function [T1]=imp_x_t1(ts,T1,T2,w)


xmax=size(T1,1);
ymax=size(T1,2);

% Load global variable
global k q ly ny
```

```
aw=ts.aW;
ae=ts.aE;
as=ts.aS;
an=ts.aN;
ap=ts.aP;



for j=2:ymax-1
    % TDMA variables
    a=zeros(1,xmax-2);
    b=zeros(1,xmax-2);
    c=zeros(1,xmax-3);
    d=zeros(1,xmax-2);
   % TDMA Boundary Conditions
    b(1)=ap(2,j)/w; c(1)=-ae(2,j);
    d(1)=as(2,j)*T1(2,j-1)+an(2,j)*T1(2,j+1)+ap(2,j)*T1(2,j)*(1/w-1)+...
        aw(2,j)*T1(1,j);
    a(xmax-2)=-aw(xmax-1,j)+0.5*ae(xmax-1,j);
    b(xmax-2)=ap(xmax-1,j)/w-3/2*ae(xmax-1,j);
    d(xmax-2)=as(xmax-1,j)*T1(xmax-1,j-1)+an(xmax-1,j)*T1(xmax-1,j+1)+...
        ap(xmax-1,j)*T1(xmax-1,j)*(1/w-1);
    for i=2:xmax-3
        a(i)= -aw(i+1,j);
        b(i)=ap(i+1,j)/w;
        c(i)= -ae(i+1,j);
        d(i)=as(i+1,j)*T1(i+1,j-1)+an(i+1,j)*T1(i+1,j+1)+...
            ap(i+1,j)*T1(i+1,j)*(1/w-1);
    end
    % Call TDMA function
    [T1(2:xmax-1,j)] = TDMA(a,b,c,d);

 for i=2:xmax
T1(i,1)=(T2(xmax+1-i,ymax-1)+T1(i,2))/2;
T1(i,ymax)=(T2(xmax+1-i,2)+T1(i,ymax-1))/2;
end

    % dT/dx=constant
    T1(xmax,j)=3/2*T1(xmax-1,j)-0.5*T1(xmax-2,j);
end

return;
```

```
% T2 solve implicit x (Duct2)

function [T2]=imp_x_t2(ts,T2,T1,w)

xmax=size(T2,1);
ymax=size(T2,2);

% Load global variable
global k q ly ny

aw=ts.aW;
ae=ts.aE;
as=ts.aS;
an=ts.aN;
ap=ts.aP;



for j=2:ymax-1
     % TDMA variables
    a=zeros(1,xmax-2); b=zeros(1,xmax-2);
```

```
    c=zeros(1,xmax-3); d=zeros(1,xmax-2);
    % TDMA Boundary Conditions
    b(1)=ap(2,j)/w; c(1)=-ae(2,j);
    d(1)=as(2,j)*T2(2,j-1)+an(2,j)*T2(2,j+1)+ap(2,j)*T2(2,j)*(1/w-1)+...
        aw(2,j)*T2(1,j);
    a(xmax-2)=-aw(xmax-1,j)+0.5*ae(xmax-1,j);
    b(xmax-2)=ap(xmax-1,j)/w-3/2*ae(xmax-1,j);
    d(xmax-2)=as(xmax-1,j)*T2(xmax-1,j-1)+an(xmax-1,j)*T2(xmax-1,j+1)+...
        ap(xmax-1,j)*T2(xmax-1,j)*(1/w-1);
    for i=2:xmax-3
        a(i)= -aw(i+1,j);
        b(i)=ap(i+1,j)/w;
        c(i)= -ae(i+1,j);
        d(i)=as(i+1,j)*T2(i+1,j-1)+an(i+1,j)*T2(i+1,j+1)+...
            ap(i+1,j)*T2(i+1,j)*(1/w-1);
    end
    % Call TDMA function
    [T2(2:xmax-1,j)] = TDMA(a,b,c,d);

    for i=2:xmax
T2(i,1)=(T1(xmax+1-i,ymax-1)+T2(i,2))/2;
T2(i,ymax)=(T1(xmax+1-i,2)+T2(i,ymax-1))/2;
end



    % dT/dx=constant
    T2(xmax,j)=3/2*T2(xmax-1,j)-0.5*T2(xmax-2,j);

end

return;
```

```
% T1 solve implicit y (Duct1)

function [T1]=imp_y_t1(ts,T1,T2,w)

xmax=size(T1,1);
ymax=size(T1,2);

% Load global variable
global k q ly ny

% allocate structure array
aw=ts.aW;
ae=ts.aE;
as=ts.aS;
an=ts.aN;
ap=ts.aP;

for i=2:1:xmax-1
    % TDMA variables
    a=zeros(1,ymax-2); b=zeros(1,ymax-2);
    c=zeros(1,ymax-3); d=zeros(1,ymax-2);
     % TDMA Boundary Conditions
    b(1)=ap(i,2)/w; c(1)=-an(i,2);
    d(1)=aw(i,2)*T1(i-1,2)+ae(i,2)*T1(i+1,2)+as(i,2)*T1(i,1)...
        +ap(i,2)*T1(i,2)*(1/w-1);
    a(ymax-2)=-as(i,ymax-1);
    b(ymax-2)=ap(i,ymax-1)/w;
    d(ymax-2)=aw(i,ymax-1)*T1(i-1,ymax-1)+ae(i,ymax-1)*T1(i+1,ymax-1)+...
        ap(i,ymax-1)*T1(i,ymax-1)*(1/w-1)+...
        an(i,ymax-1)*T1(i,ymax);
    for j=2:ymax-3
        a(j)= -as(i,j+1);
```

```
        b(j)=ap(i,j+1)/w;
        c(j)= -an(i,j+1);
        d(j)=aw(i,j+1)*T1(i-1,j+1)+ae(i,j+1)*T1(i+1,j+1)+...
            ap(i,j+1)*T1(i,j+1)*(1/w-1);
    end
    % Call TDMA function
    [T1(i,2:ymax-1)] = TDMA(a,b,c,d);


T1(i,1)=(T2(xmax+1-i,ymax-1)+T1(i,2))/2;
T1(i,ymax)=(T2(xmax+1-i,2)+T1(i,ymax-1))/2;
end
%  dT/dx=constant
T1(xmax,:)=3/2*T1(xmax-1,:)-0.5*T1(xmax-2,:);
return;
```

```
% T2 solve implicit y (Duct2)

function [T2]=imp_y_t2(ts,T2,T1,w)

xmax=size(T2,1);
ymax=size(T2,2);

% Load global variable
global k q ly ny


aw=ts.aW;
ae=ts.aE;
as=ts.aS;
an=ts.aN;
ap=ts.aP;

for i=2:2:xmax-1
   % TDMA variables
    a=zeros(1,ymax-2); b=zeros(1,ymax-2);
    c=zeros(1,ymax-3); d=zeros(1,ymax-2);
     % TDMA Boundary Conditions
    b(1)=ap(i,2)/w; c(1)=-an(i,2);
    d(1)=aw(i,2)*T2(i-1,2)+ae(i,2)*T2(i+1,2)+as(i,2)*T2(i,1)...
        +ap(i,2)*T2(i,2)*(1/w-1);
    a(ymax-2)=-as(i,ymax-1);
    b(ymax-2)=ap(i,ymax-1)/w;
    d(ymax-2)=aw(i,ymax-1)*T2(i-1,ymax-1)+ae(i,ymax-1)*T2(i+1,ymax-1)+...
        ap(i,ymax-1)*T2(i,ymax-1)*(1/w-1)+...
        an(i,ymax-1)*T2(i,ymax);
    for j=2:ymax-3
        a(j)= -as(i,j+1);
        b(j)=ap(i,j+1)/w;
        c(j)= -an(i,j+1);
        d(j)=aw(i,j+1)*T2(i-1,j+1)+ae(i,j+1)*T2(i+1,j+1)+...
            ap(i,j+1)*T2(i,j+1)*(1/w-1);
    end
    % Call TDMA function
    [T2(i,2:ymax-1)] = TDMA(a,b,c,d);

    T2(i,1)=(T1(xmax+1-i,ymax-1)+T2(i,2))/2;
    T2(i,ymax)=(T1(xmax+1-i,2)+T2(i,ymax-1))/2;
end

% dT/dx=constant
T2(xmax,:)=3/2*T2(xmax-1,:)-0.5*T2(xmax-2,:);
return;
```

```
% Residual

function [Ru,Rv,Rp,Rt,R]=residuals(us,vs,ts,u,v,p,T,IT,R)

% Call global variables
global rho lx ly nx ny umax

% u velocity residual

xmax=size(u,1); ymax=size(u,2);
tempora1=zeros(xmax,ymax); tempora2=zeros(xmax,ymax);

for j=2:ymax-1
    for i=2:xmax-1
        tempora1(i,j)=abs(us.aP(i,j)*u(i,j)-us.aW(i,j)*u(i-1,j)-...
            us.aE(i,j)*u(i+1,j)-us.aS(i,j)*u(i,j-1)-...
            us.aN(i,j)*u(i,j+1)-ly/ny*(p(i,j)-p(i+1,j)));
        tempora2(i,j)=abs(us.aP(i,j)*u(i,j));
    end
end
Ru=sum(tempora1(:))/sum(tempora2(:));

% v velocity residual

xmax=size(v,1); ymax=size(v,2);
tempora1=zeros(xmax,ymax); tempora2=zeros(xmax,ymax);

for j=2:ymax-1
    for i=2:xmax-1
        tempora1(i,j)=abs(vs.aP(i,j)*v(i,j)-vs.aW(i,j)*v(i-1,j)-...
            vs.aE(i,j)*v(i+1,j)-vs.aS(i,j)*v(i,j-1)-...
            vs.aN(i,j)*v(i,j+1)-lx/nx*(p(i,j)-p(i,j+1)));
        tempora2(i,j)=abs(vs.aP(i,j)*v(i,j));
    end
end
Rv=sum(tempora1(:))/sum(tempora2(:));

% P pressure residual
xmax=size(p,1); ymax=size(p,2);
temp=zeros(xmax,ymax);

for j=2:ymax-1
    for i=2:xmax-1
        temp(i,j)=abs(rho*((u(i-1,j)-u(i,j))*ly/ny+...
            (v(i,j-1)-v(i,j))*lx/nx));
    end
end
Rp=sum(temp(:))/(rho*umax*ly);

% T temperature residual
xmax=size(T,1); ymax=size(T,2);
tempora1=zeros(xmax,ymax); tempora2=zeros(xmax,ymax);

for j=2:ymax-1
    for i=2:xmax-1
        tempora1(i,j)=abs(ts.aP(i,j)*T(i,j)-ts.aW(i,j)*T(i-1,j)-...
            ts.aE(i,j)*T(i+1,j)-ts.aS(i,j)*T(i,j-1)-ts.aN(i,j)*T(i,j+1));
        tempora2(i,j)=abs(ts.aP(i,j)*T(i,j));
    end
end
Rt=sum(tempora1(:))/sum(tempora2(:));
```

```
R(1,IT)=Ru; R(2,IT)=Rv; R(3,IT)=Rp; R(4,IT)=Rt;


end
```

```
% Velocity at nodal points

function [u,v] = postprocess(us,vs,u,v)


xmax=size(u,1);
ymax=size(u,2);

% calculate x velocity on nodal points
u(xmax+1,:)=u(xmax,:);
u(2:xmax,1:ymax)=(u(1:xmax-1,1:ymax)+u(2:xmax,1:ymax))/2;


xmax=size(v,1);
ymax=size(v,2);

% calculate y velocity on nodal points
v(:,ymax+1)=v(:,ymax);
v(1:xmax,2:ymax)=(v(1:xmax,1:ymax-1)+v(1:xmax,2:ymax))/2;
```

```
% Bulk Temperature (Duct1)

function [T_bulk1,rho_bulk1,Nu]=Tbulk1(T1,rho1,ts1,rhos1,uc1)

% Call global variables
global k lx ly nx ny EntL h Th Pit q ExiL



for i=1:size(T1,1)
u_ave1(i)=trapezoid(ts1.y(i,2:end-1),uc1(i,2:end-1))/ly;
T_bulk1(i)=trapezoid(ts1.y(i,2:end-1),uc1(i,2:end-1).*T1(i,2:end-1))/u_ave1(i)/ly;
rho_bulk1(i)=trapezoid(rhos1.y(i,2:end-1),uc1(i,2:end-1).*rho1(i,2:end-1))/u_ave1(i)/ly;
end


Mu1T=round(EntL/lx*nx)+round(Th/lx*nx+1); % Index of viscosity matrix
Mu0T=round(EntL/lx*nx+1);
Mu1B=round(EntL/lx*nx)+round(((Pit-(abs(Pit-Th)/2))+Th)/lx*nx+1);
Mu0B=round(EntL/lx*nx)+round((Pit-(abs(Pit-Th)/2))/lx*nx+1);
j=1;

while Mu1B<(nx+2-round(ExiL/lx*nx))


 for i=Mu0B+1:1:Mu1B
u_ave1(i)=trapezoid(ts1.y(i,round(h/ly*ny+1):end-1),uc1(i,round(h/ly*ny+1):end-1))/ly;
T_bulk1(i)=trapezoid(ts1.y(i,round(h/ly*ny+1):end-1),uc1(i,round(h/ly*ny+1):end-
1).*T1(i,round(h/ly*ny+1):end-1))/u_ave1(i)/ly;
rho_bulk1(i)=trapezoid(rhos1.y(i,round(h/ly*ny+1):end-1),uc1(i,round(h/ly*ny+1):end-
1).*rho1(i,round(h/ly*ny+1):end-1))/u_ave1(i)/ly;
 end

 for i=Mu0T+1:1:Mu1T
u_ave1(i)=trapezoid(ts1.y(i,2:(ny+2-round(h/ly*ny))),uc1(i,2:(ny+2-round(h/ly*ny))))/ly;
T_bulk1(i)=trapezoid(ts1.y(i,2:(ny+2-round(h/ly*ny))),uc1(i,2:(ny+2-
round(h/ly*ny))).*T1(i,2:(ny+2-round(h/ly*ny))))/u_ave1(i)/ly;
```

```
rho_bulk1(i)=trapezoid(rhos1.y(i,2:(ny+2-round(h/ly*ny))),uc1(i,2:(ny+2-
round(h/ly*ny))).*rho1(i,2:(ny+2-round(h/ly*ny))))/u_ave1(i)/ly;

 end

    Mu0T=round(EntL/lx*nx)+round((Th+Pit)*j/lx*nx+1);
    Mu1T=round(EntL/lx*nx)+round((Th+((Th+Pit)*j))/lx*nx+1);
    Mu1B=round(EntL/lx*nx)+round(((Pit-(abs(Pit-Th)/2))+Th+((Th+Pit)*j))/lx*nx+1);
    Mu0B=round(EntL/lx*nx)+round(((Pit-(abs(Pit-Th)/2))+((Th+Pit)*j))/lx*nx+1);


    j=j+1;
end

hconv=2*k/(ly/ny)*((T1(:,1)-T1(:,2))./(T1(:,1)-T_bulk1(:)));
Nu=hconv*2*ly/k;

figure(1)
plot(ts1.x(:,1),Nu,'LineWidth',2)
xlabel('Channel length, L [m]','FontSize',10)
ylabel('Nu','FontSize',10)

end




end
```

```
% Bulk Temperature (Duct1)

function [T_bulk2,Nu]=Tbulk2(T2,ts2,uc2)

% Call global variables
global k lx ly nx ny EntL h Th Pit q ExiL



for i=1:size(T2,1)
u_ave2(i)=trapezoid(ts2.y(i,2:end-1),uc2(i,2:end-1))/ly;
T_bulk2(i)=trapezoid(ts2.y(i,2:end-1),uc2(i,2:end-1).*T2(i,2:end-1))/u_ave2(i)/ly;
end



Mu1T=round(EntL/lx*nx)+round(Th/lx*nx+1); % Index of viscosity matrix
Mu0T=round(EntL/lx*nx+1);
Mu1B=round(EntL/lx*nx)+round(((Pit-(abs(Pit-Th)/2))+Th)/lx*nx+1);
Mu0B=round(EntL/lx*nx)+round((Pit-(abs(Pit-Th)/2))/lx*nx+1);
j=1;

while Mu1B<(nx+2-round(ExiL/lx*nx))


 for i=nx+2-Mu0T:-1:nx+2+1-Mu1T
u_ave2(i)=trapezoid(ts2.y(i,round(h/ly*ny+2):end-1),uc2(i,round(h/ly*ny+2):end-1))/ly;
T_bulk2(i)=trapezoid(ts2.y(i,round(h/ly*ny+2):end-1),uc2(i,round(h/ly*ny+2):end-
1).*T2(i,round(h/ly*ny+2):end-1))/u_ave2(i)/ly;
 end

 for i=nx+2-Mu0B:-1:nx+2+1-Mu1B
u_ave2(i)=trapezoid(ts2.y(i,2:(ny+2-round(h/ly*ny+1))),uc2(i,2:(ny+2-
round(h/ly*ny+1))))/ly;
T_bulk2(i)=trapezoid(ts2.y(i,2:(ny+2-round(h/ly*ny+1))),uc2(i,2:(ny+2-
round(h/ly*ny+1))).*T2(i,2:(ny+2-round(h/ly*ny+1))))/u_ave2(i)/ly;
```

```
 end


    Mu0T=round(EntL/lx*nx)+round((Th+Pit)*j/lx*nx+1);
    Mu1T=round(EntL/lx*nx)+round((Th+((Th+Pit)*j))/lx*nx+1);
    Mu1B=round(EntL/lx*nx)+round(((Pit-(abs(Pit-Th)/2))+Th+((Th+Pit)*j))/lx*nx+1);
    Mu0B=round(EntL/lx*nx)+round(((Pit-(abs(Pit-Th)/2))+((Th+Pit)*j))/lx*nx+1);

    j=j+1;
end

hconv=2*k/(ly/ny)*((T2(:,1)-T2(:,2))./(T2(:,1)-T_bulk2(:)));
Nu=hconv*2*ly/k;

    % 3. Plot T_bulk and Tw as function of x
% figure(5)
% plot(ts2.x,T_bulk2,'LineWidth',2)
% xlabel('Channel length, L [m]','FontSize',18)
% ylabel('Temperature [K]','FontSize',18)

figure(2)
plot(ts2.x(:,1),Nu,'LineWidth',2)
xlabel('Channel length, L [m]','FontSize',10)
ylabel('Nu','FontSize',10)

% % 4. Plot loca Nusselt number as a function of x
% figure(4)
% h=2*k/(ly/ny)*((T(:,1)-T(:,2))./(T(:,1)-T_bulk1(:)));
% Nu=h*2*ly/k;
% plot(ts.x(:,1),Nu,'LineWidth',2)
% xlabel('Channel length, L [m]','FontSize',24)
% ylabel('Local Nusselt number, Nu','FontSize',24)
end
```