ABSTRACT

Two-dimensional laminar natural convection heat transfer in air around an isothermal vertical plate, an isothermal horizontal flat plate subjected to heat from both sides, and also isothermal horizontal ducts with rectangular and square cross sections are examined numerically.

The computational procedures are based on the finite difference technique. Finite difference scheme based on the integration of the governing equations over finite cells is applied to the two dimensional, steady state problem. Different aspect ratios are used for a wide range of Rayleigh number. Results are presented in the form of streamlines, velocity vectors, and isotheral plots around the circumference of the ducts. Software for this purpose was developed using Java language for the numerical solution.

Circulation and separation are observed at high aspect ratios. The value of y-component of velocities near the vertical wall of the duct is found to be higher than the values of y-component of velocities near a vertical plate having the same height. Heat transfer data is generated and presented in terms of Nusselt numbers versus Rayleigh numbers for different aspect ratios. The correlation covering the aspect ratios used is obtained in dimensionless form of Nusselt number, Rayleigh number, and aspect ratio. The results of the correlation are in close agreement with the data from the numerical analysis with a maximum deviation of 10.34%.

Keywords: Finite difference scheme, Natural convection, Rectangular duct, Aspect ratio, Horizontal plate, Vertical plate.

ÖZET

Her iki yüzeyinden soğutulan sabit sıcaklıktaki düşey bir levha, yatay bir levha ve ayrıca dikdörtgen kesitli yatay bir kanal üzerindeki iki boyutlu laminer doğal taşınım ısı transferi sayısal olarak incelenmiştir.

Sayısal çözümlemede sonlu farklar tekniği kullanımıştır. Korunum denklemlerini sonlu hücrelerde entegre etme esasına dayanan sonlu farklar yönetimi iki boyutlu kararlı probleme uygulanmıştır. Sayısal çözümleme için Java programlama dili kullanılarak bir yazılım geliştirilmiştir. Hesaplamalarda geniş bir Rayleigh sayısı aralığında farklı aspekt oranı kullanılmıştır ve elde edilen sonuçlar akım çizgileri, hız vektörleri ve sabit sıcaklık konturları olarak verilmiştir.

Dikdörtgen kesitli yatay kanallarda yüksek aspekt oranında sirkülasyon ve seperasyonun yaşandığı gözlenmiştir. Bununla birlikte düşey yüzey boyunca düşey hızın düşey levhaya ait düşey hızlardan daha yüksek olduğu belirlenmiştir. Farklı aspekt oranları için taşınım ısı tansferi katsayıları elde edilmiş ve Rayleigh sayısının fonksiyonu olarak Nusselt sayısı şekilinde grafikler halinde verilmiştir. Bu çalışmada kullanılan aspekt oranı ve Rayleigh sayısı aralığı için, Nusselt sayısını Rayleigh sayısı ve aspekt oranının bir fonksiyonu olarak veren bir korelasyon elde edilmiştir. Bu korelasyon ve sayısal çözümleme ile elde edilen değerler karşılatırıldığında bunların, aralarında en fazla %10.34'lük bir farkla, birbirlerine yakın değerler oldukları görünmektedir.

Anahtar kelimeler: Sonlu farklar, Doğal taşınım, Dikdörtgen kanal, Aspekt oranı, Düşey levha, Yatay levha.

ACKNOWLEDGMENTS

Foremost, I would like to express my sincere gratitude to my advisor Dr. Cemal GÖVSA for the continuous support of my Master's study, for his patience, motivation, enthusiasm, and immense knowledge. His guidance helped me in all the time of research and writing of this thesis. (I could not have imagined having a better advisor and mentor for my Master's study).

In addition, I would like to thank Assist. Prof. Dr. Ali EVCIL and Assist. Prof. Dr Hüseyin ÇAMUR, who offered me the opportunity to work with them in the Department of Mechanical Engineering at NEU and gave me untiring help during my difficult moments.

I am indebted to my many of my colleagues to support me, especially Sharif Al-agha for his unlimited help in writing the code, Mohammed Rabaia and his wife Ferihan, who gave me their laptop in a critical time when my laptop stopped working, I used their laptop until I finished my thesis.

Finally, I owe special thanks to my dearest family; my parents; since without their encouragement it would have been impossible for me to finish my work, they helped me a lot to pass many tides during this thesis.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

APPREVIATIONS

Latin Symbols	**Definition**
a, b, c, d	Coefficients in Eq. (3.18)
A_E, A_W, A_N,A_S	Coefficients in the convection terms
B_E, B_W, B_N,B_S	Coefficients in the diffusion term
c_p	Specific heat at constant pressure, J/ kg.K
C_E, C_W, C_N,C_S	Coefficient in the substitution formula, Eq.(3.45)
G	Gravitational acceleration, m/s^2
Gr	Grashof number
H	Heat transfer coefficient, W/m^2K
Δh	Step size
H	Height of the object, m
I	Index of the nods in the horizontal distance
J	Index of the nodes in the vertical distance
K	Thermal conductivity, W/m.K
L	Length
L_1, L_2, L_3	Metric coefficients of the coordinate system
Nu	Nusselt number
P	Pressure, Pa
Pr	Prandtl number
Q	Heat flux
R_∞	Radius of the pseudo boundary, m
Ra	Rayleigh number
T	Temperature, K
U	Velocity component in x-direction, m/s
V	Velocity component in y-direction, m/s
W	Width of the object, m
X	Cartesian coordinate denoting horizontal distance
Y	Cartesian coordinate denoting vertical distance

Greek Symbols	
A	Angel
B	Volume expansion coefficient, 1/K
Γ	Aspect ratio
ξ_1, ξ_2	Orthogonal curvilinear coordinates
M	Dynamic viscosity, kg/m.s

P	Density
Φ	A dependent variable
Ψ	Stream function
Ω	Vorticity
Λ	The error

Subscripts

c	Convection
diff	Diffusion
F	Film
S	At the surface
∞	Ambient
Sor	Source
LS	Lower surface
US	Upper surface
RS	Right surface
e, w, n, s, ne, nw, se, sw	Compass point location of the integration cell
E, W, N, S, NE, NW, SE, SW	Compass point location of nodes with respect to p

CHAPTER 1

INTRODUCTION

Convective heat transfer is the study of heat transport process affected by the flow of fluids. In convective heat transfer problems, the flow usually is classified as forced convection flow in which the flow is caused by external forces, such as pumps or fans, or free convection flow which observed as a result of the motion of the fluid due to density changes arising from heating process. A hot radiator used for heating a room is one example of a practical device which transfers heat by free convection. The movement of the fluid in free convection, whether it is a gas or a liquid, results from the buoyancy forces imposed on the fluid when its density in the proximity of heat transfer surface is decreased as a result of the heating process. The buoyancy forces would not be present if the fluid were not acted upon by some external force field such as the gravity.

Natural convection heat transfer has increasingly attracting the interest of researchers. This is due to its importance in many engineering applications. Among various problems of natural convection, laminar convection from heated rectangular or square ducts is an important problem in heat transfer. It is used to simulate a wide range of engineering applications as well as provide a better insight into more new systems of heat transfer. Accurate knowledge of the overall natural convection heat transfer around ducts is important; while it appears in variety of natural circumstances such as thermal plums and meteorological phenomena, including heat exchangers, design of solar collectors, and cooling of electronic circuit boards. Because of its industrial importance, this class of heat transfer has been extensively studied experimentally, but the numerical investigations are not so many as expected, due to the larger theoretical complexities involved, where the mass flow rate is unknown , the momentum and energy equations are coupled, and the fluid flow cannot be assumed to be fully developed.

In this study, natural convection over three different geometries; isothermal horizontal duct, vertical plate and an isothermal horizontal flat plate subjected to heat transfer from both sides, surrounded by a fluid reservoir is assumed to be at constant temperature T_∞, will be investigated numerically using finite difference method and applying the current numerical techniques, successive over relaxation iteration method were chosen, which performed by developing a software using Java language. The buoyancy which induces the flow is considered to be steady and laminar, the geometries are assumed to be much longer than their heights and widths, which implies that the end effects can reasonably be neglected and temperature and velocity profiles can be considered to be two dimensional. The fluid is air and the physical properties were assumed as constants at the film temperature, and negligible viscous dissipation and pressure work. The buoyancy effects on the momentum transfer are taken into account through the Boussinesq approximation. These considerations are employed in the conservation equations of mass, momentum and energy.

CHAPTER 2

LITERATURE SURVEY

2.1 Over view

Prior to World War II, convective heat and mass transfers were largely empirical sciences, and engineering design was accomplished almost by the use of experimental data generalized to some degree by dimensional analysis. During the past two decades great studies have been made in developing analytical methods of convection analysis. While today experiment is assumed to be more classical role of testing the validity of the theoretical models [1]. This is not to say that direct experimental data are not still of vital importance in engineering design, but there is no question that the area of complete dependence on direct experimental data has been greatly diminished. Most studies concern the nature of the portions of such flows attached to the surface of a body. These may be described by simplified mathematical models, such as the boundary layer theory, the subsequent regions where the flow may be separate from the surface and rise perhaps as a buoyant plume have not been considered in details. The laminar boundary layer equations are physically only meaningful for large Grashof numbers 10^9 for which they represent asymptotic solutions to the continuity, momentum and energy equations. For air in free convection on a vertical flat plate, the critical Grashof number has been observed by Eckert and Soehngen [2] to be approximately 4×10^8. Values ranging between 10^8 and 10^9 may be observed for different fluids and environment "turbulence levels".

Due to the nonlinearities arise from the fluid acceleration terms and the coupling of the momentum energy equations; the exact solution of the boundary layer equations for natural convection is complicated by the nonlinear nature of the problem. Because of the analytical

difficulties, there has been a heavy reliance on empirical information to solve many problems of engineering importance. The empirical approach, however, is not completely satisfactory since, when complicated geometries are involved, it is not always clear which dimensionless parameters should be used to correlate experimental results. Another difficulties arise in the empirical approach because of the velocities are usually so small that they are very difficult to measure in the free convection problems. Despite of that, velocity measurements have been performed using hydrogen-bubble techniques [3], hot-wire anemometry [4]. Temperature field measurements have been obtained through the use of the Zehender-Mach interferometer. The laser anemometer [5] is particularly useful for free convection measurements because it does not disturb the flow field.

Several techniques; for flow measurements have been discussed by Jensen [6]. Basic theory and implementation of those techniques is provided. The intent of this study is to provide enough detail to enable a novice user to make an informed decision in selecting the proper equipment to solve a particular flow measurement problem.

Additionally, many flows of practical interest involve geometries and boundary conditions which do not permit similarity solutions where the complete Navier Stokes and energy equations have to be considered.

The problem laminar natural convection received continuous attention since the early work of Churchill and Chu; a theoretical solution using boundary layer theory obtained for laminar and turbulent free convection from a vertical plate [7] and from horizontal cylinder [8]. The expression is applicable to uniform heating as well as to uniform wall temperature and for mass transfer and simultaneous heat and mass transfer. The correlation provides a basis for estimating transfer rates for non-Newtonian fluids and for inclined plates and cylinders.

The first work used to test the numerical methods for computation of laminar natural convection in enclosures is referred to Churchill and Chu[9], the model represents two dimensional, laminar, and natural convection in a rectangular cross section of a long channel is developed in terms of the stream function and the vorticity. The Grashof number, the Prandtl number, the aspect ratio and the boundary conditions are adjustable parameters. The effects of

initial conditions, grid size and the order of the various finite difference representations were investigated. Computed values of the steady-state heat flux over various planes were compared.

A survey of the literature shows that correlations for external natural convection from different geometries have been reported for different thermal boundary conditions. Furthermore, numerical studies on heat transfer inside horizontal ducts, cavities, and enclosures have been reported by many authors for various boundary conditions. For the current study, in the literature survey; for results comparison three geometries were considered vertical flat plate, horizontal flat plate, and rectangular ducts, and for the numerical techniques followed to solve the problems horizontal circular cylinder has been studied.

2.2 Vertical Plate

Natural convection heat transfer from flat surfaces continues to be an area of study that requires further research. Many scientific and engineering applications, such as meteorological thermal rises or plumes emanating from the Earth's surface, are driven by natural convection heat transfer. Investigations of this mode of heat transfer from flat surfaces have been conducted for a variety of plan forms and orientations, and using a variety of techniques.

Extensive published studies on the literature, deal with isothermal vertical plate; Dachun and Jianjun [10] experimentally studied the instability of the isothermal natural convection boundary layer around a vertical heated flat plate, T. Zitzmann, Malcolm C., and P. Peter [11], investigated the air flow and heat transfer over a heated vertical plate and in a differentially heated cavity using Computational Fluid Dynamics (CFD).

Recently in 2005 a numerical and standard experimental study of the interaction natural convection in an isothermal vertical plate finite-sized, with a neighborhood vertical back side and horizontal down placed surface has been carried out by Comunelo and Güths [12]. The problem was solved using the finite volume method on a two-dimensional structured grid built using ANSYS BUILD grid and mesh generator software. The full governing elliptical equations was solved using ANSYS CFX 5.7.1 solver, results obtained show the

increasing caused by back side influence is around 23 % while the wall down is 30 % indicating that wall down approximation is slightly more efficient.

2.3 Horizontal Plate

Goldstein et. al. [13] investigated the natural convection mass transfer adjacent to a horizontal downward facing plate. Using a naphthalene sublimation technique, a common correlation for horizontal plates with specific characteristic lengths in the Sherwood and Rayleigh numbers was obtained. The results were compared to a number of other studies and some differences were noted due to deviations in geometry and experimental technique. Lloyd and Moran [14] used an electrochemical technique to measure mass transfer. They developed correlations for turbulent and laminar mass transfer, the latter agreeing with Goldstein et. al.

Wei, Yu and Tao [15] studied numerically the simultaneous natural convection heat transfer from both sides of uniformly heated thin plate with arbitrary inclination, correlations surrounded by air. Finite volume method was adopted to descretize the governing equations. The SIMPLE algorithm was used for treating the coupling between the velocity and pressure, and QUICK scheme and first order upwind difference scheme were used to descretize the convective terms in momentum and energy equations, respectively. Average Nusselt number on the upper and lower surfaces is expressed in terms of Rayleigh number and the inclination angle as:

$$Nu_{u,avg} = \left[0.317 + 0.645\,(\sin\theta)^{1.18}\right] Ra^{0.2} \tag{2.1}$$

and

$$Nu_{l,avg} = \left[0.675(Ra)^{-0.04} + \left[0.00293(Ra)^{0.256} + 0.158\right]\sin\theta\right] Ra^{0.2} \tag{2.2}$$

for $4.8\times10^{6} \leq Ra \leq 1.87\times10^{8}$.

The results show that for $Ra \leq 7.5\times10^{7}$, average Nusselt number on the lower surface is larger than the corresponding Nusselt number on the upper surface.

In the field of meteorology, natural convection in a large body of fluid leading to freely rising plumes is of great concern. This led Pera and Gebhart [16] to conduct analysis and experiment for horizontal and slightly inclined surfaces. Using interfere grams, a comparison was made between analytical and experimental results which were found to be in fair agreement depending on Prantl and Grashof numbers. Pera and Gebhart [17] then repeated their investigation to include small disturbances. Their goal was to compare an analysis of the stability of flow subjected to small disturbances using inter ferograms,, to experimental results in order to provide greater insight into instability and flow transition.

Zakerullah and Ackroyd [18] investigated natural convection over circular disk numerically, using the Runge-Kutta-Merson integration procedure, taking into the account the properties variation of the fluid. They found that the behavior is close to the flow over flat plate. Kobus and Wedekind [19] experimentally presented heat transfer data and dimensionless correlations were proposed for natural convection from heated, stationary, isothermal, horizontal disks over a wide range of Rayleigh number.

Al-Arabi and El-Reidy [20] studied natural convection over heated isothermal plates. Investigating a number of plan forms, they developed new local and average heat transfer correlations for both laminar and turbulent boundary layers.

2.4 Horizontal Circular Cylinder

One of the recent studies done by; Molla, Hossain, and Pual [21] analysis of natural convection laminar boundary layer flow from a horizontal circular cylinder is done with a uniform surface temperature at presence of heat generation. The governing boundary layer equations are transformed into a non-dimensional form and the resulting nonlinear systems of partial differential equations are solved numerically applying two distinct methods; implicit finite difference method together with the Keller box scheme and series solution technique. The results of the surface shear stress in terms of the local skin friction and the surface rate of heat transfer in terms of the local Nusselt number for a selection of the heat generation parameter c = 0.0, 0.2, 0.5, 0.8 and 1.0 are obtained and presented in both tabular and graphical formats. Without effect of the internal heat generation inside the fluid domain the

parameter c has been taken to be 0, as a result of this study the effect of heat generation both the velocity and temperature distributions, increased significantly and the thickness of the thermal boundary layer enhanced.

Corcione [22], numerically studied the free convection from flat vertical arrays of equally spaced horizontal isothermal cylinders to derive heat transfer dimensionless correlating equations for any individual cylinder in the array and for the whole tube array, spanning across a range of the Rayleigh number definitely wider than those of other empirical equations. The study is performed under the assumption of steady laminar flow, for arrays of 2–6 circular cylinders, for center-to-center separation distances from 2 up to more than 50 cylinder diameters, and for values of the Rayleigh number based on the cylinder diameter in the range between 5×10^2 and 5×10^5. At any Rayleigh number, and for any tube array, the heat transfer rate at the bottom cylinder is substantially identical to that for a single cylinder is obtained as result of this work.

Then, in 2006, Corcione [23], improved his work by an investigation on both the nature and the effects of the free convective interactions which occur between a pair of flat vertical arrays of horizontal isothermal cylinders set parallel to each other in free space. The study was performed numerically under the assumption of steady laminar flow, for pairs of tube-arrays consisting of 1–4 circular cylinders, for center-to-center horizontal and vertical spacing from 1.4 to 24 cylinder-diameters, and from 2 to 12 cylinder-diameters, respectively, and for values of the Rayleigh number based on the cylinder diameter in the range between 10^2 and 10^4.

2.5 Horizontal Ducts

Experiments on convective heat transfer from side walls of a vertical square cylinder in air were done by Popiel and Wojtkowiak [24]. It was found that the experimental data obtained; below some Rayleigh numbers depending on the ratio between the height and the width of the cylinder, due to the curvature effects, the average natural convective heat transfer from side walls of the vertical square cylinder is higher in comparison with the data for a vertical flat plate. The boundary layer curvature effects on the average heat transfer coefficient for a square cylinder (prism) are represented in the correlating equation by the factor $\left(\frac{H}{w}\right)^{0.123}$.

Two dimensional laminar natural convection heat transfers in air around horizontal ducts with rectangular and square cross sections was studied numerically by Zeitoun and Ali [25]. Different aspect ratios were used for wide ranges of Rayleigh numbers. Results were presented in the form of streamlines and isothermal plots around the circumference of the ducts. The computational procedure was based on the finite-element technique. Temperature and velocity profiles were obtained near each surface of the ducts. Reverse flow and circulations are observed at high aspect ratios. Heat transfer data were generated and presented in terms of Nusselt number versus Rayleigh number for different aspect ratios. Correlation covering the aspect ratios used was obtained in dimensionless form of Nusselt number, Rayleigh number, and aspect ratio. They showed that as Rayleigh number increases, the velocity boundary layer thickness decreases and the maximum velocity shifts toward the surface of the body. The correlation for Nusselt number they got in terms of Ra number was:

$$Nu = \left(0.9\Gamma^{-0.061} + 0.371\Gamma^{0.114}Ra^{0.1445}\right)^2 \quad ,700 \leq Ra \leq 1.3 \times 10^8 \tag{2.3}$$

where Γ represents the aspect ratio between height and width.

CHAPTER 3

METHODOLOGY

3.1 Introduction

Most of the earlier studies related with the solution of convective flow problems have been concerned with two dimensional forced boundary layer flows. These have one predominant direction of flow. The shear stress and heat flux are significant only in the directions perpendicular to the main flow. Such simplification enables the partial differential equations to be reduced to the parabolic form which are simpler to solve than the elliptic ones. Patankar and Spalding [26] surveyed these investigations and pointed out the categories of the methods used.

Hellums and Churchil [7, 8] were the first to solve natural convection problems by using finite difference method. They employed the boundary layer type assumptions and use an explicit method to obtain the numerical solutions.

The effect of recirculation flows and interaction of flow fields make the equations elliptic and so their reduction to the parabolic form is not applicable. In spite of the large number of papers which have been produced on parabolic flows, relatively little has been done in the field of elliptic flows. Moreover, related studies were generally on forced convection flow.

An improved numerical solution technique of predicting flows was developed by Gosman et. al. [27]. The method of solution avoids the instabilities arising from central difference scheme at high Reynolds numbers by using an upwind difference technique. This technique regards the direction of the fluid flow when formulating the finite difference representation of conservation equations. Lounder and Massey [28] have applied this method to the prediction of forced convective flow through tube banks. After that, Farouk [29] has

successfully adapted this method to laminar and turbulent natural convection heat transfer from horizontal cylinders. Bezzazoglu [30] also used the same method to solve the laminar natural convective heat transfer over two vertically spaced horizontal cylinders .Govsa [31] adapted the above method for laminar natural convective heat transfer in a cavity .

After reviewing available numerical techniques, the above finite difference technique was selected to solve the strongly coupled equations of natural convection. The method is capable of predicting two dimensional, laminar, and turbulent re-circulating natural convection; for a wide range of flow situations; this feature makes it attractive in the presence study.

3.2 The Conservation Equations

The fundamental conservation laws of mass, momentum and energy provide the differential equation. Then auxiliary relations for the thermodynamics and transport properties and for the boundary conditions, make the mathematical problem complete. All that is needed to obtain solution to the equations and to deduce information. The present method of solution is finite difference one. It confines attention to finite number of points, distributed regularly through the flow field as the nodes of a grid. The differential equations should be replaced by algebraic ones in order to calculate the values of variables on these nodes. The simultaneous algebraic equations thus obtained are non-linear. Therefore, an iterative procedure of solution can be devised.

Considering the particular flow problem, the natural convective flow induced by a heated vertical flat plate, heated horizontal plate and horizontal isothermal rectangular duct in free space, the grid to be used is schematically illustrated in Figure3.1.

The equations required are in the Cartesian coordinate system. The conservation equations for two-dimensional steady-state laminar flow, employing Boussinesq approximation i.e. the density variations are negligible in the continuity, momentum and energy equations except in the buoyancy terms as follows:

1) Conservation of Mass

$$\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}=0 \tag{3.1}$$

2) Conservation of Momentum

- x-direction

$$\rho\left[u\frac{\partial u}{\partial x}+v\frac{\partial u}{\partial y}\right]=-\frac{\partial P}{\partial x}+\mu\left[\frac{\partial^2 u}{\partial x^2}+\frac{\partial^2 u}{\partial y^2}\right] \tag{3.2}$$

- y-direction

$$\rho\left[u\frac{\partial v}{\partial x}+v\frac{\partial v}{\partial y}\right]=-\frac{\partial P}{\partial y}+\mu\left[\frac{\partial^2 v}{\partial x^2}+\frac{\partial^2 v}{\partial y^2}\right]-\grave{\rho}g \tag{3.3}$$

Since $\rho\grave{}=\grave{\rho}(T)$, the volume expansion coefficient of the fluid

$$\beta=-\frac{1}{\rho}\left(\frac{\partial\grave{\rho}}{\partial T}\right)_p=-\frac{1}{\rho}\frac{\partial\grave{\rho}}{\partial T}=-\frac{1}{\rho}\frac{(\grave{\rho}-\rho)}{(T-T_\infty)} \tag{3.4}$$

then,

$$\grave{\rho}=\rho[1-\beta(T-T_\infty)] \tag{3.5}$$

substituting the value of $\grave{\rho}$ into the body force term, the conservation of momentum equation in y-direction becomes

$$\rho\left[u\frac{\partial v}{\partial x}+v\frac{\partial v}{\partial y}\right]=-\frac{\partial P}{\partial y}+\mu\left[\frac{\partial^2 v}{\partial x^2}+\frac{\partial^2 v}{\partial y^2}\right]-\rho g+\rho g\beta(T-T_\infty) \tag{3.6}$$

3) Conservation of Energy

$$\rho c_p\left[u\frac{\partial T}{\partial x}+v\frac{\partial T}{\partial y}\right]=k\left[\frac{\partial^2 T}{\partial x^2}+\frac{\partial^2 T}{\partial y^2}\right] \tag{3.7}$$

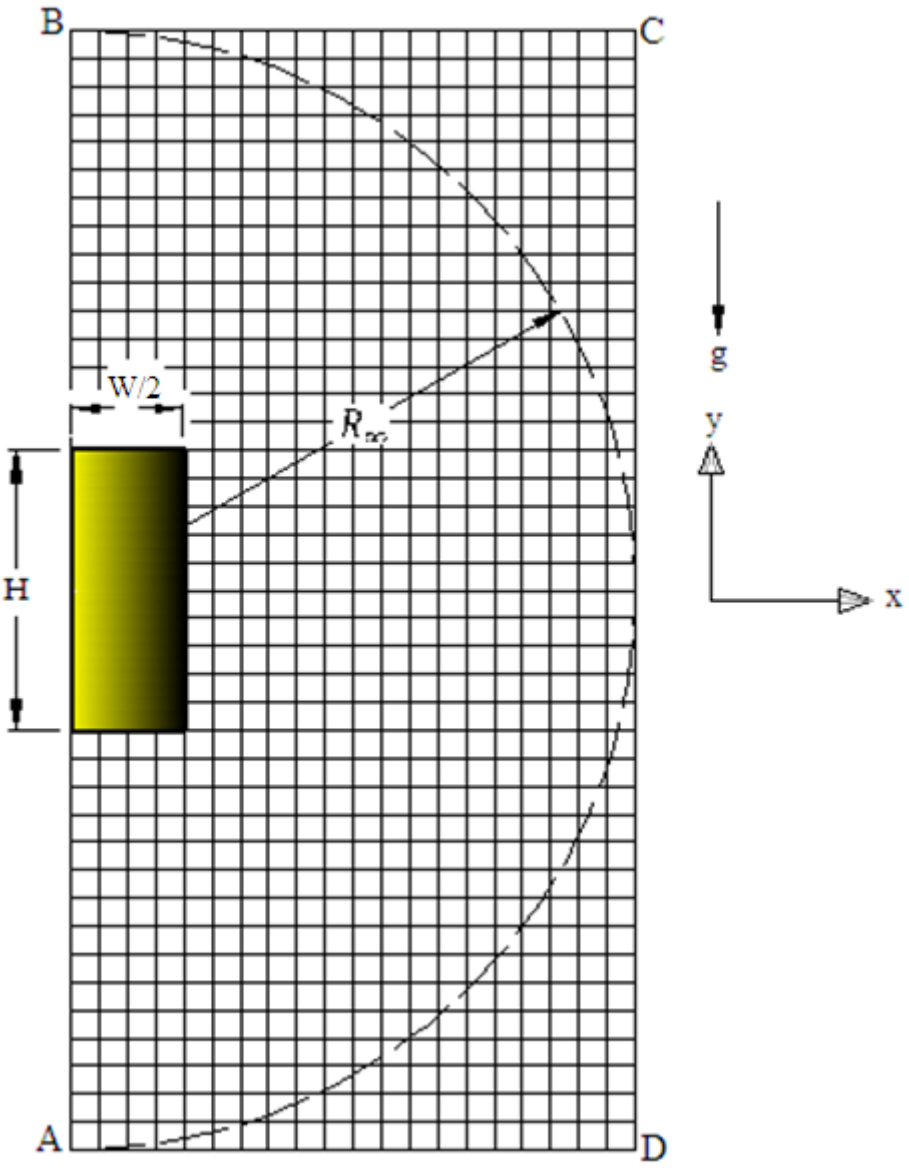

Figure 3.1 Right symmetry Cartesian grid used for flow calculations (not to scale).

In this study, it is assumed that there are no heat sources due to chemical reaction, kinetic heating or any other means of heat loss, or gain by radiation is also ignored. The boundary conditions of the problem are given in Figure 3.2.

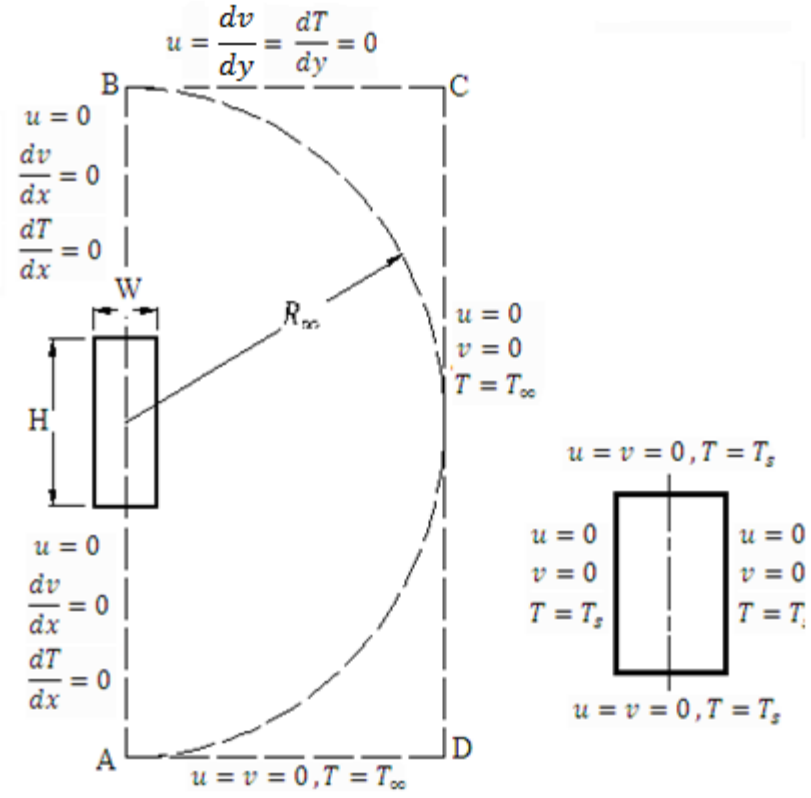

Figure 3.2 Boundary conditions, field boundaries (left), object (right)

These three conservation equations; conservation of mass, momentum and energy equation with the appropriate boundary conditions will be enough to obtain a solution for four unknowns, namely, two components of velocity, temperature and pressure at any point in the flow domain. However, the solution of the three flow equations will not be so simple since there is no equation for the pressure and the construction of finite difference equation will be very complicated.

Fortunately, before setting up the finite difference equations, the two velocity components and the pressure can be replaced by another two unknowns namely, stream function (ψ), and vorticity (ω), then the pressure term will be eliminated since the fluid flow can be considered as two dimensional flow.

Introducing the stream function and vorticity

$$u = \frac{1}{\rho}\frac{\partial \psi}{\partial y} \qquad v = -\frac{1}{\rho}\frac{\partial \psi}{\partial x} \qquad \omega = \left[\frac{\partial v}{\partial x} - \frac{\partial u}{\partial y}\right] \tag{3.8}$$

According to the above arrangements, the governing equations take the following forms:

$$\left[\frac{\partial}{\partial x}\left(\omega\frac{\partial \psi}{\partial y}\right) - \frac{\partial}{\partial y}\left(\omega\frac{\partial \psi}{\partial x}\right)\right] - \left[\frac{\partial^2(\mu\omega)}{\partial x^2} + \frac{\partial^2(\mu\omega)}{\partial y^2}\right] = g\rho\beta\frac{\partial T}{\partial x} \tag{3.9}$$

$$\frac{1}{\rho}\frac{\partial^2 \psi}{\partial x^2} + \frac{1}{\rho}\frac{\partial^2 \psi}{\partial y^2} = -\omega \tag{3.10}$$

$$\left[\frac{\partial}{\partial x}\left(T\frac{\partial \psi}{\partial y}\right) - \frac{\partial}{\partial y}\left(T\frac{\partial \psi}{\partial x}\right)\right] - \left[\frac{\partial}{\partial x}\left(\frac{k}{c_p}\frac{\partial T}{\partial x}\right) + \frac{\partial}{\partial y}\left(\frac{k}{c_p}\frac{\partial T}{\partial \theta}\right)\right] = 0 \tag{3.11}$$

Now, the number of equations and unknowns are reduced to three, after rearrangements, namely vorticity, stream function and temperature.

The simultaneous solution of the flow and energy equations will yield distributions of vorticity, stream function and temperature. The velocity field can obtained from the stream function distribution. Since the driving force for the flows comes from the buoyancy term, the pressure distribution is generally not sought for the natural convection problems. Once the velocity and temperature fields have been solved the pressure distribution can be obtained.

In the derivation of these flow equations certain assumptions were made concerning the behavior of the shear stresses (τ). In laminar flows, the assumption that the fluid is Newtonian means that the shear stresses can be related to the rate of strain as follows:

$$\tau_{xy} = \mu\left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\right) \tag{3.12}$$

The heat flux q can be related to the temperature gradient in a like manner for laminar flows. In this case, the thermal conductivity k rather than the laminar viscosity is the proportionality constant.

$$q_x = -k\frac{\partial T}{\partial x} \qquad q_y = -k\frac{\partial T}{\partial y} \tag{3.13}$$

3.3 Boundary Conditions

The full equations governing the fluid motion are of elliptic type; therefore they require that the boundary conditions for each of the dependent variable be specified along the entire boundary enclosing the physical solution domain. However, for the problem considered there is no physical boundary enclosing the flow domain completely.

Since the objects are in open air; the location of the boundaries far from the objects is not clear. Therefore, a pseudo boundary has to be employed to limit the solution domain and boundary conditions have to be defined also in this pseudo boundary. In general there are three types of boundaries in this problem, these are the symmetry plane, solid walls and open surfaces which allow the flow to go through.

3.3.1 Boundary Conditions on the Symmetry Plane

Since the boundary conditions as well as the physical geometry of the flow field are symmetrical and the gravity vector is along one of the independent coordinates, symmetric solution, therefore a symmetry plane can be employed for this problem. Furthermore, since these conditions are satisfied, it can easily be seen that there will be no flow across the symmetry plane AB, thus the stream function along the coordinates which coincides with the symmetry plane can be zero. It also follows from the reasoning that the vorticity along the symmetry plane can be zero. The same rule applies to the temperature making its gradient across the symmetry plane equal to zero. These boundary conditions can mathematically be stated as

$$\omega|_{AB} = \psi|_{AB} = \frac{dT}{dy}\Big|_{AB} = 0 \tag{3.14}$$

3.3.2 Boundary Conditions on the Surface of the Duct

By definition of this type boundary, the objects surfaces are not-permeable; there is no flow across it. Therefore, the stream function value is constant along the solid boundary. Its value is same as along the adjacent symmetry plane. The thermal boundary conditions applied in the present study is constant surface temperature.

The specification of the boundary condition for vorticity is generally difficult and follows mathematical treatment rather than physical reasoning since it requires information on the gradients of velocities which are not known a priori. A common practice is to use Taylor series expansion type expression to evaluate the vorticity at wall node in terms of the flow variables at the nodes inside the flow field and making use of the continuity and the no-slip condition. This is generally done by considering the first node in the flow field next to the solid boundary. One such expression is given as [32]

$$\omega_p = -\frac{2\Delta\psi}{\rho(\Delta h)^2} \tag{3.15}$$

Here Δh represents the distance between the wall and the node at which stream function, ψ, is evaluated. If the wall is vertical then Δh represents Δx, if its horizontal Δh represents Δy.

3.3.3 Boundary Conditions at the Far Field

The necessity to limit the size of the solution domain requires that a pseudo boundary be defined in the far field. To establish these conditions a common way that the flow can be assumed to be perpendicular to the sections in which case the gradient condition namely that the derivative of the stream function with respect to the coordinate perpendicular to the pseudo boundary is zero. If the domain is large circle, the stream lines becomes normal to the boundary hence the boundary conditions of the stream function at the right pseudo boundary (CD) in Figure 3.2 become:

$$\frac{\partial\psi}{\partial x}\Big|_{R_\infty} = 0 \tag{3.16}$$

and at the upper and lower pseudo boundaries BC and DA in Figure 3.2 are :

$$\frac{\partial \psi}{\partial y}|_{R\infty} = 0 \qquad (3.17)$$

The boundary conditions of the vorticity also should be determined at the pseudo boundaries; which can be given in two regions. The first one is where the fluid is coming to the flow bounadry CD and DA. At this region effect of the heated object is negligible, and it is reasonable to assume that the fluid there is stationary, thus the vorticity along the coordinates which coincides with this region can be zero, $\omega|_{CD,DA} = 0$. The second region where the fluid is leaving the flow boundary BC, However, the y-component of the velocity of the fluid is not known in a priori, but it is known that the x-component of the velocity at this region is zero, thus the applicable boundary condition for the vorticity is the gradient of the vorticity is zero, $\frac{d\omega}{dy}|_{BC} = 0$

In the present study, however, the selected Cartesian grid does not allow the use of such a large circle because of the rectangular shape of the flow domain. Since the radius of this large circle has no fixed value, instead of trying to obtain such a circular pseudo boundary, the rectangular shape of the pseudo boundary was not changed and the above form of the boundary conditions was applied for each grid node on rectangular pseudo boundary. Center of this circle of the pseudo boundary was thought to be in the middle of the left symmetry plane. The radius of the pseudo boundary was determined as follows. First, a large radius was selected and the problem was solved using this radius, then this radius was reduced gradually up to a length for which the local Nusselt numbers was not significantly affected.

The temperature of the fluid coming to the flow field is the same as the ambient temperature. However, the temperature of the fluid leaving the pseudo boundary is not known a priori. It is assumed that, as in the case of stream function and vorticity boundary conditions, the temperature gradient normal to the pseudo boundary is zero, implying that, heat transfer on this boundary is purely by convection rather than by conduction.

The boundary conditions of this study for the flow field and the heated walls in terms of ψ and ω are illustrated in Figure 3.3.

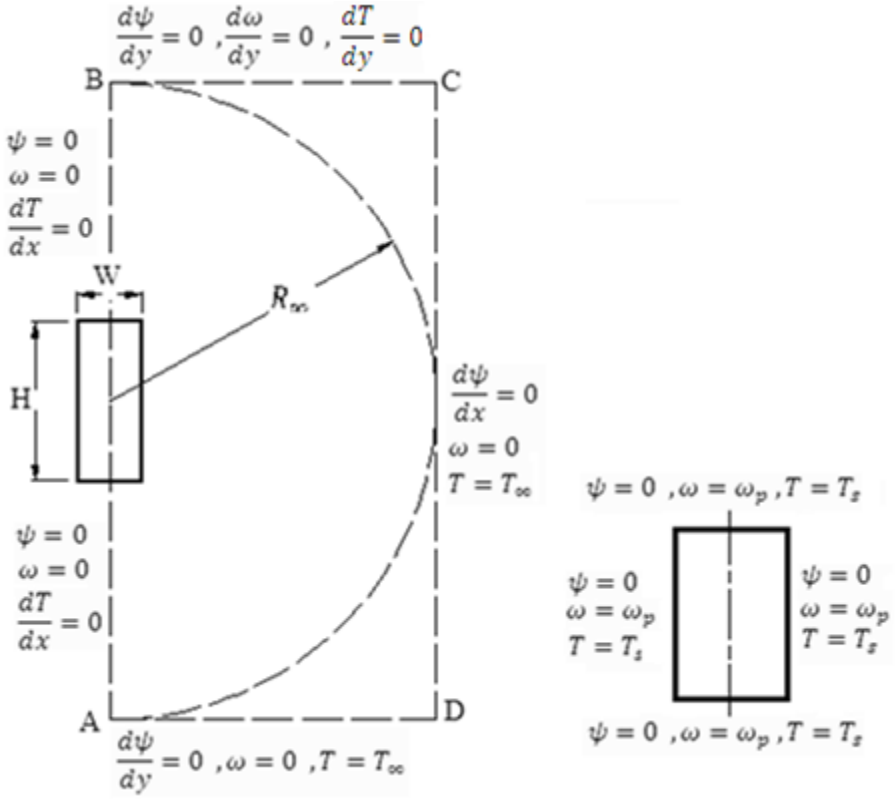

Figure 3.3 Boundary conditions in terms of stream function and vorticity, pseudo boundaries (left), object (right), ω_p is given by Eq. (3.15)

3.4 Finite Difference Formulation

It is mathematically possible to find a solution for three dependent variables for any required conditions since the conservation laws governing the flow and heat transfer for laminar natural convection have been reduced to three partial differential equations for stream function, vorticity and temperature. Thus, the problem became three equations with three unknowns.

It is impossible to solve three strongly coupled equations analytically, if the flow geometry and Grashof numbers are considered, since it is necessary to make some assumptions and approximations with which the solution will be destroyed.

Therefore, these three elliptical partial differential equations have to be solved by the means of numerical methods. For this purpose, the numerical solution technique developed by

Gosman et. al. [27]. For dealing primarily with forced convection flows and modified for solving natural convection problems by Farouk [29] will be used in this study.

For both polar and Cartesian coordinate systems the independent variables, the radii of curvature, metric coefficients and the angle α (used only for the body force term) are as shown in the table below:

Table 3.1 Table of Equivalences for Cartesian and Polar Coordinates

System	ξ_1	ξ_2	ξ_3	l_1	l_2	l_3	α
Cartesian	z	x	y	1	1	1	$+90^o$
Polar	z	r	Θ	1	1	r	Θ

The three governing equations can be expressed in the form of a single general elliptic equation proposed by Gosman et. al. [27]

$$a^{\emptyset}\left[\frac{\partial}{\partial\xi_2}\left(\phi\frac{\partial\psi}{\partial\xi_3}\right)-\frac{\partial}{\partial\xi_3}\left(\phi\frac{\partial\psi}{\partial\xi_2}\right)\right]-\left[\frac{\partial}{\partial\xi_2}\left(b^{\emptyset}\frac{l_1l_3}{l_2}\frac{\partial(c^{\emptyset}\emptyset)}{\partial\xi_2}\right)+\frac{\partial}{\partial\xi_3}\left(b^{\emptyset}\frac{l_1l_2}{l_3}\frac{\partial(c^{\emptyset}\emptyset)}{\partial\xi_3}\right)\right] = -l_1l_2l_3d^{\emptyset} \tag{3.18}$$

Where Ø represents the dependent variables ψ, ω and T, the multipliers $a^{\emptyset}, b^{\emptyset}, c^{\emptyset}$and $d^{\emptyset}$ take the forms as shown in the following table.

Table 3.2 The Functions and Associated with Equation 3.18

Ø	$a^{\emptyset}$	$b^{\emptyset}$	$c^{\emptyset}$	$d^{\emptyset}$
ω	1	1	μ	$-g\rho\beta\left(\frac{\partial T}{\partial\xi_2}sin\alpha+\frac{1}{l_3}\frac{\partial T}{\partial\xi_3}cos\alpha\right)$
ψ	0	l_1^2/ρ	1	$-\omega/l_1$
T	1	k/c_p	1	0

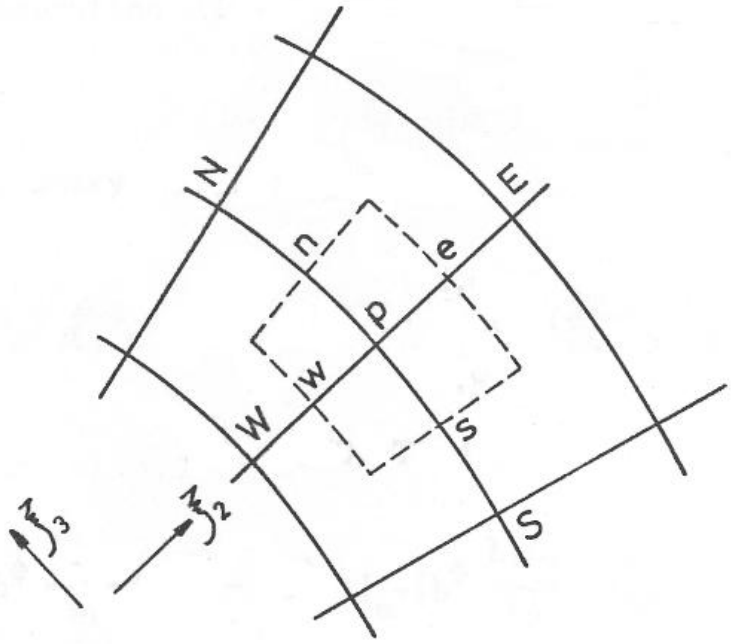

Figure 3.4 Portion of the finite difference grid, dotted lines enclose the area of integration

For the purpose of the derivation of the finite difference equations, the field of interest has been covered by an orthogonal grid network and the nodes of finite difference grid correspond with the intersections of the grid lines. The figure above displays a part of such a grid, a typical node p, and the four surrounding nodes N, S, E and W are shown. The finite difference equation will be eventually expressed primarily in terms of the values of the variables at these nodes, and to lesser extent in terms of the values on the nodes labeled NE, NW, SE, and SW.

Instead of using standard Taylor series expansions, the finite difference equations are obtained by integration over a micro-cell shown by the dotted lines which enclose the point p.

The general equation is then integrated over the cell area to give:

$$\int_{\xi_{3,s}}^{\xi_{3,n}}\int_{\xi_{2,w}}^{\xi_{2,e}} a^{\emptyset}\left[\frac{\partial}{\partial\xi_2}\left(\phi\frac{\partial\psi}{\partial\xi_3}\right)-\frac{\partial}{\partial\xi_3}\left(\phi\frac{\partial\psi}{\partial\xi_2}\right)\right]d\xi_2 d\xi_3$$
$$-\int_{\xi_{3,s}}^{\xi_{3,n}}\int_{\xi_{2,w}}^{\xi_{2,e}}\left[\frac{\partial}{\partial\xi_2}\left(b^{\emptyset}\frac{l_1 l_3}{l_2}\frac{\partial(c^{\emptyset}\emptyset)}{\partial\xi_2}\right)+\frac{\partial}{\partial\xi_3}\left(b^{\emptyset}\frac{l_1 l_2}{l_3}\frac{\partial(c^{\emptyset}\emptyset)}{\partial\xi_3}\right)\right]d\xi_2 d\xi_3 \qquad (3.19)$$
$$=-\int_{\xi_{3,s}}^{\xi_{3,n}}\int_{\xi_{2,w}}^{\xi_{2,e}} l_1 l_2 l_3 d^{\emptyset}\, d\xi_2 d\xi_3$$

Where the integration limits are the coordinates of the sides of the rectangle.

By formal integration, one obtains:

$$\underbrace{a_P^{\phi}\left[\int_{\xi_{3,s}}^{\xi_{3,n}}\left\{\phi_e\left(\frac{\partial\psi}{\partial\xi_3}\right)_e-\phi_w\left(\frac{\partial\psi}{\partial\xi_3}\right)_w\right\}d\xi_3-\int_{\xi_{2,w}}^{\xi_{2,e}}\left\{\phi_n\left(\frac{\partial\psi}{\partial\xi_2}\right)_n-\phi_s\left(\frac{\partial\psi}{\partial\xi_2}\right)_s\right\}d\xi_2\right]}_{CONVECTION\ TERM}$$

$$\underbrace{-\int_{\xi_{3,s}}^{\xi_{3,n}}\left\{\left[b^{\phi}\frac{l_1 l_3}{l_2}\right]_e\left[\frac{\partial}{\partial\xi_2}(c^{\emptyset}\emptyset)\right]_e-\left[b^{\phi}\frac{l_1 l_3}{l_2}\right]_w\left[\frac{\partial}{\partial\xi_2}(c^{\emptyset}\emptyset)\right]_w\right\}d\xi_3}_{DIFFUSION\ TERM} \qquad (3.20)$$

$$\underbrace{\begin{aligned}&-\int_{\xi_{2,w}}^{\xi_{2,e}}\left\{\left[b^{\phi}\frac{l_1 l_2}{l_3}\right]_n\left[\frac{\partial}{\partial\xi_3}(c^{\emptyset}\emptyset)\right]_n-\left[b^{\phi}\frac{l_1 l_2}{l_3}\right]_s\left[\frac{\partial}{\partial\xi_3}(c^{\emptyset}\emptyset)\right]_s\right\}d\xi_2=\\&\qquad-\int_{\xi_{3,s}}^{\xi_{3,n}}\int_{\xi_{2,w}}^{\xi_{2,e}} l_1 l_2 l_3 d^{\emptyset}\, d\xi_2 d\xi_3\end{aligned}}_{SOURCE\ TERM}$$

The names convection, diffusion and source are assigned to various groups of terms. In order to proceed further, we must introduce additional assumptions which will differ according to the nature of the terms; therefore, each group will be considered separately.

3.4.1 Convection Term

There are four integrals to be evaluated in the convection terms; however it will be sufficient to outline the procedure for only one of these in detail. The first of the integrals which denoted by the symbol I_c, is

$$I_c = a_P^{\phi} \int_{\xi_{3,s}}^{\xi_{3,n}} \phi_e \left(\frac{\partial \psi}{\partial \xi_3}\right)_e d\xi_3 \tag{3.21}$$

Since ϕ and ψ are well behaved functions, there exists an average value of ϕ_e which is donated by $\overline{\phi_e}$, such that:

$$\overline{\phi_e} = \frac{\int_{\xi_{3,s}}^{\xi_{3,n}} \phi_e \left(\frac{\partial \psi}{\partial \xi_3}\right)_e d\xi_3}{\int_{\xi_{3,s}}^{\xi_{3,n}} \left(\frac{\partial \psi}{\partial \xi_3}\right)_e d\xi_3} \cong \frac{I_c}{a_P^{\phi}(\varphi_{ne} - \varphi_{se})} \tag{3.22}$$

Then I_c can be expressed as:

$$I_c = a_P^{\phi} \overline{\phi_e} (\psi_{ne} - \psi_{se}) \tag{3.23}$$

To express the values of $\overline{\phi_e}$, ψ_{ne} and ψ_{se} in terms of the values of the variables at the nodes of the grid, we make three assumptions as follow:

1- Ø is uniform within each rectangle and has the value which prevails at particular node which the rectangle encloses.
2- $\overline{\phi_e}$ takes on the ϕ value possessed by the fluid upstream of e-face of the rectangle.
3- The value of stream function at a particular corner of the rectangle is equal to the average of the values on the four neighboring nodes.

For example:

$$\psi_{se} \cong \frac{\psi_{SE} + \psi_E + \psi_P + \psi_S}{4} \tag{3.24}$$

For later convenience, Eq. (3.25) is expressed in a new way in which the upwind difference scheme has been introduced.

$$I_c = a_P^{\phi}\left[\emptyset_E\left\{\frac{(\psi_{ne} - \psi_{se}) - |\psi_{ne} - \psi_{se}|}{2}\right\} + \emptyset_P\left\{\frac{(\psi_{ne} - \psi_{se}) - |\psi_{ne} - \psi_{se}|}{2}\right\}\right] \tag{3.25}$$

In this equation the presence of the difference, first within a bracket and then within a modulus sign, ensures that one of the terms in the curly brackets in the equation will be zero and the term which remains will be that which represents the contribution from the node upstream of e-face of the rectangle. In this way upwind differences are introduced into finite difference scheme.

When the other terms are also treated in similar fashion, after minor rearrangements the sum of the integrals of all the convection terms I_{con} can be expressed as:

$$I_{con} = A_E(\emptyset_P - \emptyset_E) + A_W(\emptyset_P - \emptyset_W) + A_N(\emptyset_P - \emptyset_N) + A_S(\emptyset_P - \emptyset_S) \tag{3.26}$$

where,

$$\begin{aligned}
A_E &= a_P^{\phi}[(\psi_{SE} + \psi_S - \psi_{NE} - \psi_N) + |\psi_{SE} + \psi_S - \psi_{NE} - \psi_N|]/8 \\
A_W &= a_P^{\phi}[(\psi_{NW} + \psi_N - \psi_{SW} - \psi_S) + |\psi_{NW} + \psi_N - \psi_{SW} - \psi_S|]/8 \\
A_N &= a_P^{\phi}[(\psi_{NE} + \psi_E - \psi_{NW} - \psi_W) + |\psi_{NE} + \psi_E - \psi_{NW} - \psi_W|]/8 \\
A_S &= a_P^{\phi}[(\psi_{SW} + \psi_W - \psi_{SE} - \psi_E) + |\psi_{SW} + \psi_W - \psi_{SE} - \psi_E|]/8
\end{aligned} \tag{3.27}$$

3.4.2 Diffusion Term

Again because of similarity of the terms, only one term is considered

$$I_d = \int_{\xi_{3,s}}^{\xi_{3,n}} \left[b^{\phi} \frac{l_1 l_3}{l_2} \right]_e \left[\frac{\partial}{\partial \xi_2} (c^{\emptyset} \emptyset) \right]_e d\xi_3 \tag{3.28}$$

The following assumptions can be made to evaluate this integral

$$l_{3,e} \cong \frac{l_{3,E} + l_{3,P}}{2} \tag{3.29}$$

$$b_e^{\emptyset} \cong \frac{b_E^{\emptyset} + b_P^{\emptyset}}{2} \tag{3.30}$$

$$\left[\frac{\partial}{\partial \xi_2} (c^{\emptyset} \emptyset) \right]_e \cong \frac{c_E^{\emptyset} \emptyset_E - c_P^{\emptyset} \emptyset_P}{\xi_{2,E} - \xi_{2,P}} \tag{3.31}$$

These assumptions are, of course correct if the quantities: l_3 , b^{ϕ} and $(c^{\emptyset}\emptyset)$ vary linearly with ξ_2. The integration then yields:

$$I_d = \frac{l_{3,E} + l_{3,P}}{2} \cdot \frac{b_E^{\emptyset} + b_P^{\emptyset}}{2} \cdot \frac{c_E^{\emptyset} \emptyset_E - c_P^{\emptyset} \emptyset_P}{\xi_{2,E} - \xi_{2,P}} \cdot \frac{\xi_{3,n} - \xi_{3,s}}{2} \tag{3.32}$$

The other diffusion terms are evaluated in similar arguments and the following expression for I_{diff}, the sum of the integrals of all four terms, is obtained.

$$I_{diff} = B_E(c_E^{\emptyset} \emptyset_E - c_P^{\emptyset} \emptyset_P) + B_W(c_W^{\emptyset} \emptyset_W - c_P^{\emptyset} \emptyset_P) + B_N(c_N^{\emptyset} \emptyset_N - c_P^{\emptyset} \emptyset_P) + B_S(c_S^{\emptyset} \emptyset_S - c_P^{\emptyset} \emptyset_P) \tag{3.33}$$

where,

$$B_E = \frac{b_E^{\emptyset} + b_P^{\emptyset}}{2} \cdot \frac{\xi_{3,n} - \xi_{3,s}}{\xi_{2,E} - \xi_{2,P}} (l_{3,E} + l_{3,P}) \tag{3.34}$$

$$B_W = \frac{b_W^{\emptyset} + b_P^{\emptyset}}{2} \cdot \frac{\xi_{3,n} - \xi_{3,s}}{\xi_{2,P} - \xi_{2,W}} \left(l_{3,W} + l_{3,P}\right)$$

$$B_N = \frac{b_N^{\emptyset} + b_P^{\emptyset}}{2} \cdot \frac{\xi_{2,E} - \xi_{2,W}}{\xi_{3,N} - \xi_{3,P}} \frac{1}{\left(l_{3,N} + l_{3,P}\right)}$$

$$B_S = \frac{b_S^{\emptyset} + b_P^{\emptyset}}{2} \cdot \frac{\xi_{2,E} - \xi_{2,W}}{\xi_{3,P} - \xi_{3,S}} \frac{1}{\left(l_{3,S} + l_{3,P}\right)}$$

3.4.3 Source Term

The final integral which must be evaluated is:

$$I_{sor} = \int_{\xi_{3,s}}^{\xi_{3,n}} \int_{\xi_{2,w}}^{\xi_{2,e}} l_1 l_2 l_3 d^{\emptyset} \, d\xi_2 d\xi_3 \tag{3.35}$$

Here, integration of the source terms depends on the form of function ∅. Therefore, it has to be done separately for each of the two variables ψ and ω, it is assumed that $d^{\emptyset}$ is uniform over the area of integration and takes on the value at p.

For the stream function's equation

$$I_{sor}^{\psi} = \int_{\xi_{3,s}}^{\xi_{3,n}} \int_{\xi_{2,w}}^{\xi_{2,e}} l_1 l_2 l_3 \left(-\frac{\omega}{l_1}\right) d\xi_2 d\xi_3 \tag{3.36}$$

$$I_{sor}^{\psi} = -\frac{\omega_P}{4} \left(\xi_{2,E} - \xi_{2,W}\right) l_{3,P} \left(\xi_{3,N} - \xi_{3,S}\right) \tag{3.37}$$

For the vorticity equation,

$$I_{sor}^{\psi} = -g\rho\beta \int_{\xi_{3,s}}^{\xi_{3,n}} \int_{\xi_{2,w}}^{\xi_{2,e}} \left(l_1 l_2 l_3 \frac{\partial T}{\partial \xi_2} sin\alpha + l_1 l_2 \frac{\partial T}{\partial \xi_3} cos\alpha\right) d\xi_2 d\xi_3 \tag{3.38}$$

The above equation can be integrated by parts. After integrating once, when the temperature T appears as a multiplier within the integrand, it is assigned the value T_p

For Cartesian coordinates $\alpha = 90^{\circ}$

$$I_{sor}^{\psi} = -\frac{g\rho\beta}{4}(y_N - y_S)(T_E - T_W) \tag{3.39}$$

The integrated expressions for the convective, diffusive and source terms can now be written in such a way that the function $\emptyset_P$ can be expressed as a successive substitution formula.

$$\begin{aligned} A_E(\emptyset_P - \emptyset_E) + A_W(\emptyset_P - \emptyset_W) + A_N(\emptyset_P - \emptyset_N) + A_S(\emptyset_P - \emptyset_S) \\ - \left[B_E\left(c_E^{\emptyset}\emptyset_E - c_P^{\emptyset}\emptyset_P\right) + B_W\left(c_W^{\emptyset}\emptyset_W - c_P^{\emptyset}\emptyset_P\right) + B_N\left(c_N^{\emptyset}\emptyset_N - c_P^{\emptyset}\emptyset_P\right)\right. \\ \left. + B_S\left(c_S^{\emptyset}\emptyset_S - c_P^{\emptyset}\emptyset_P\right)\right] = -I_{sor} \end{aligned} \tag{3.40}$$

By taking $\emptyset_P$ to the left-hand side, the following general form obtained:

$$\emptyset_P = \frac{c_E\emptyset_E + c_W\emptyset_W + c_N\emptyset_N + c_S\emptyset_S - I_{sor}}{\sum AB} \tag{3.41}$$

Where,

$$\begin{aligned} c_E &= \frac{\left(A_E + B_E c_E^{\emptyset}\right)}{\sum AB} \\ c_W &= \frac{\left(A_W + B_W c_W^{\emptyset}\right)}{\sum AB} \\ c_N &= \frac{\left(A_N + B_N c_N^{\emptyset}\right)}{\sum AB} \\ c_S &= \frac{\left(A_S + B_S c_S^{\emptyset}\right)}{\sum AB} \end{aligned} \tag{3.42}$$

and

$$\sum AB \quad = A_E + A_W + A_N + A_S + c_P^{\emptyset}(B_E + B_W + B_N + B_S) \tag{3.43}$$

Before proceeding, it will be useful to put the general substitution formula (3.43) into a form which is more suitable for programming.

$$\emptyset_P = \frac{\sum_{j=E,W,N,S}\left\{\left(A_j + c_j^{\emptyset} B_j\right)\emptyset_j\right\} - I_{sor}}{\sum_{j=E,W,N,S}\left(A_j + c_j^{\emptyset} B_j\right)\emptyset_j} \tag{3.44}$$

Here the symbol $\sum_{j=E,W,N,S}$ denotes the summation over the nodes E, W, N and S, and $I_{sor} = d_P^{\emptyset} V_p$. Now dividing both the numerator and denominator byV_p, and rearranging, the following equation is obtained:

$$\emptyset_P = \frac{\sum_{j=E,W,N,S}\left\{\left[A_{j'} + c_j^{\emptyset}\left(b_j^{\emptyset} + b_j^{\emptyset}\right)B_{j'}\right]\emptyset_j\right\} - d_P^{\emptyset}}{\sum_{j=E,W,N,S}\left[A_{j'} + c_P^{\emptyset}\left(b_j^{\emptyset} + b_P^{\emptyset}\right)B_{j'}\right]} \tag{3.45}$$

Where $A_{j'}\, and\, B_{j'}$ are related to $A_j and\, B_j$ by:

$$A_{j'} = \frac{A_j}{V_p} \tag{3.46}$$

$$B_{j'} = \frac{B_j}{V_P\left(b_j^{\emptyset} + b_P^{\emptyset}\right)} \tag{3.47}$$

$$V_P = \frac{1}{4}\left(\xi_{2,E} - \xi_{2,W}\right)\left(\xi_{3,N} - \xi_{3,S}\right) \tag{3.48}$$

Equation (3.45) shows the substitution formula in the form which is needed for programming purpose. It must of course be remembered that this formula can only be applied at the interior nodes of the grid. In order to obtain a complete set of algebraic equations, additional substitution formula for those nodes which lie on the boundaries must be derived.

3.5 Solution of the Set of Algebraic Equations

The differential equations have already been converted into algebraic ones. These are simpler in form but their simplicity is balanced by their number since they have to be solved for each grid node. When the equations are linear and few in number, standard matrix inversion techniques can be used; but this case is rare. Usually, either because the equations are very numerous or because of non-linearity, successive substitution technique must be employed.

There is a choice of several alternatives; thus there is the Jacobi method, where in each cycle of iterations use is only made of values of the variables from the previous one; and there is the Gauss-Seidel method, in which the new values are used as soon as they become available. The second method has been chosen, for it is known to yield more rapid convergence than the Jacobi method and it places less demands on computer storage. Initial guesses for the values of the variables are substituted into the successive substitution formulae which have been derived from algebraic equations; and new values are computed. Then these values are used as new guesses to the solution and so on.

There is also a third method which is closely related to the Gauss-Seidel method. It is called “Successive over Relaxation, SOR”; and it is known in certain circumstances to be more rapid than the Gauss-Seidel method. However, there are certain difficulties associated with its applications. The under or over relaxation parameters have to be specified by trial and error while the value of the relaxation parameter is between 0 and 2; if the values are less than 1, it is under relaxation if more then it is, over relaxation . For this case study under relaxation parameter is used only for the vorticity to obtain convergence.

The computation procedures can be summarized as:

a. Each cycle of the iterative procedure is made up of three sub-cycles, each for one of the variables, namely vorticity, stream function and temperature.
b. In each sub-cycle the solution domain is scanned row by row and a single variable is updated during the scanning process.

c. A new sub-cycle is then performed for a new dependent variable. The procedure allows the order, in which the variables are to be updated, to be arbitrarily chosen. Usually, however, the vorticity and stream functions are updated first, in the order of mention.

d. When all of the sub-cycles have been completed, a new iteration cycle is commenced and new values are obtained.

e. The procedures are repeated until the changes in the values of the variables between successive iterations are less than a small specified quantity, λ.

$$\frac{\emptyset^N - \emptyset^{N-1}}{\emptyset^N} \leq \lambda \tag{3.49}$$

$\emptyset^N$ represents the value from the last iteration cycle and $\emptyset^{N-1}$ is the value from the previous one.

f. After the satisfaction of the convergence criterion, vertical and horizontal velocities are computed. Central difference approximation is used for the derivations of the stream function.

g. Finally local heat transfer coefficients along the heated walls and also the average heat transfer coefficient are calculated.

3.6 Algorithm of the Software

The computational procedures were implemented to a software using Java as a programming language. Figure 3.5 shows the flow chart followed to write the computer program.

The interface of program is represented in Figure 3.6, a sample mesh also appears to the right of the interface. The text fields; Object Height, Object Width, Del-x, Del-y, Domain Diameter, i-Min, T-Surface and T-Infinity are changeable variables according to the problem to be solved. In the progress monitor; the button Number of Iterations shows the number of iterations performed to achieve the convergence if it is occurred and the maximum number of iterations if the convergence not occurred. The button Maximum Residual represents the maximum error between two successive iterations.

The mesh in Figure 3.6 represents a sample of a right symmetry of a rectangular duct with a height of 50 mm width of 50 mm, step size of 5 mm, and domain diameter of 150 mm.

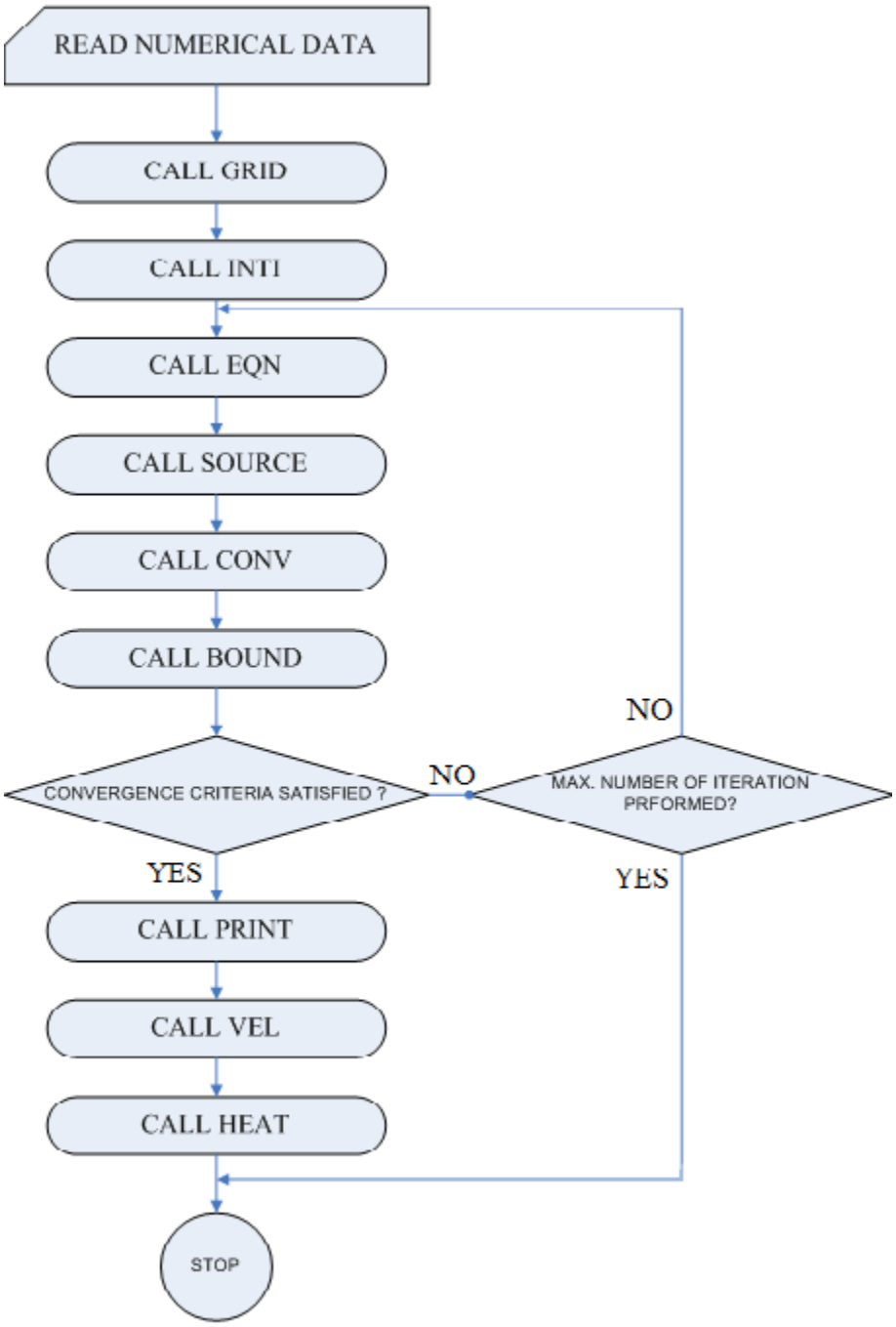

Figure 3.5 Flow chart of computer program for numerical solution

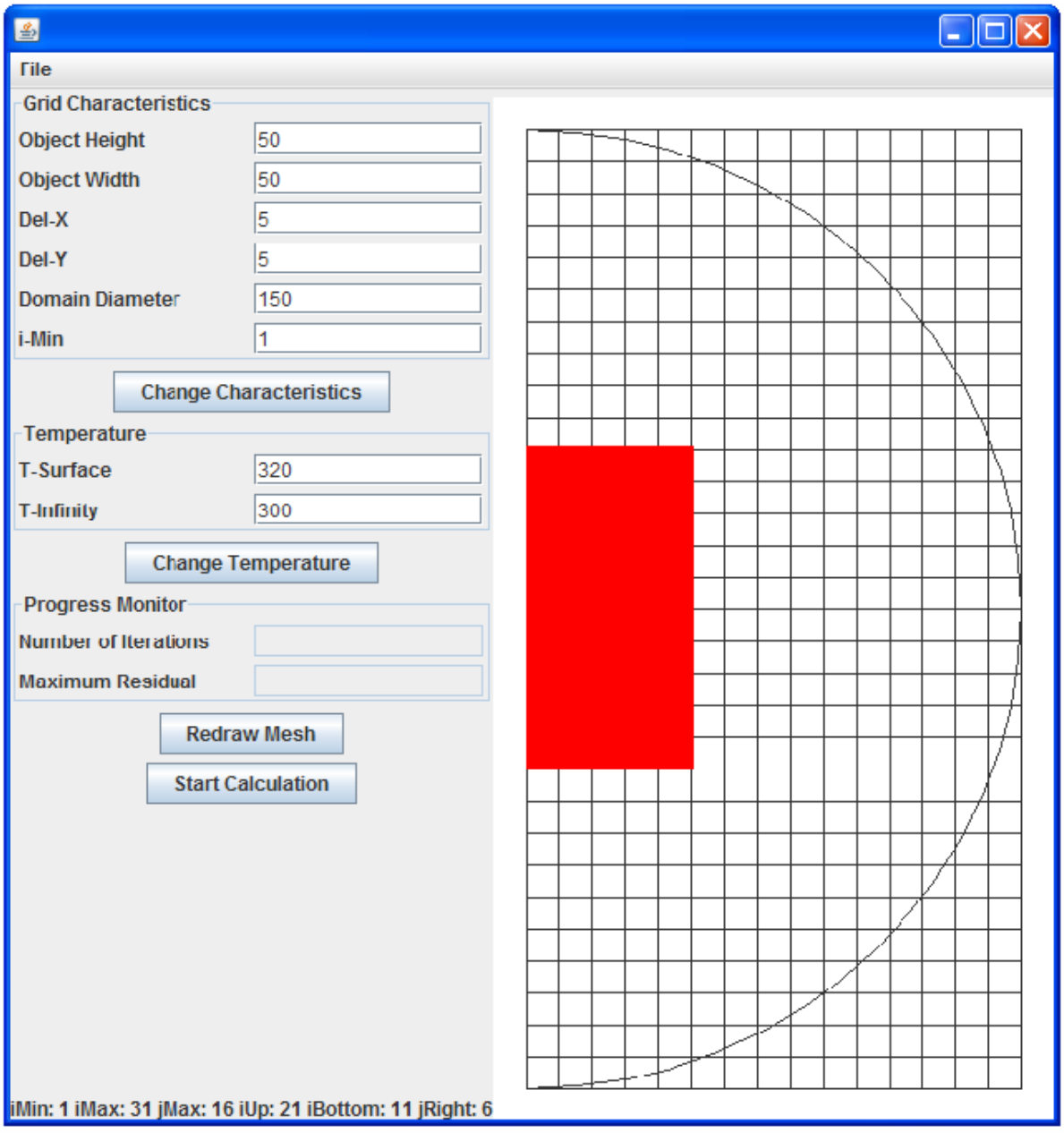

Figure 3.6 Interfaces and sample mish

CHAPTER 4

RESULTS AND DISCUSSION

4.1 Over view

In this chapter the results of the numerical computations for laminar convective heat transfer are presented for an isothermal vertical plate, isothermal horizontal plate subjected to heat transfer from both sides, and isothermal horizontal duct. The error λ between successive iterations is chosen to be less than 3×10^{-4}. The parameters of problems are:

- For the vertical plate the height of the plate (H) is the characteristic length.
- For the horizontal plate the length of the plate (W) is the characteristic length.
- For the rectangular duct aspect ratio (Γ) is defined as $\Gamma = \frac{H}{W}$ and the characteristic length is used as Lc = H+L.

4.2 Data Information

The average Nusselt number is defined as:

$$\overline{Nu} = \frac{\bar{h}.Lc}{k} \tag{4.1}$$

where $\bar{h}$ is the average heat transfer coefficient of the heated wall, Lc represents the characteristic length of the geometry; for vertical plate Lc = H, horizontal plate Lc = W, and

for the rectangular duct Lc = H+W, and k is the thermal conductivity of surrounding air. The average heat transfer coefficient is defined as:

$$\overline{h} = \frac{1}{Lc}\int_0^{Lc} h(\eta).d\eta \tag{4.2}$$

For the vertical plate η represents y-direction, for horizontal plate η is x-direction, but in the case of rectangular duct; h values are function of y for the vertical walls, and are function of x for horizontal walls, so for this case average heat transfer coefficient can be defined as:

$$\overline{h}_{Duct} = \frac{\sum h_{Ls,Us,Rs}}{N} \tag{4.4}$$

Where the subscripts Ls, Us and Rs represent lower surface, upper surface, and right surface respectively, and N represents the number of nodes at these surfaces.

The local heat transfer coefficient h(η) is evaluated from Newton's cooling law.

$$h(\eta) = \frac{q}{(T_w - T_\infty)} \tag{4.4}$$

where q, is the local convective heat flux from the heated surface to the fluid. T_w is the uniform surface temperature of the heated wall and T_∞ is the ambient temperature. q is defined as:

$$q = -k\frac{dT}{d\eta}|_{\eta=0} \tag{4.5}$$

The temperature gradient $(dT/d\eta)_{\eta=0}$ is calculated by using the temperature distribution results of the numerical program and thermal conductivity of the air is evaluated at the wall temperature.

The Grashof and Rayleigh numbers are calculated by using the following definitions:

$$Gr = \frac{\rho^2 g\beta(T_w - T_\infty)Lc^3}{\mu^2} \tag{4.6}$$

$$Ra = Gr.Pr \tag{4.7}$$

The thermo-physical properties of air were evaluated at the film temperature

$T_f = (T_w + T_\infty)/2$, except for the thermal expansion coefficient β which was evaluated as $(1/T_\infty)$ according to the perfect gas law.

Temperature profiles near the heated walls, at various vertical and horizontal levels, stream lines, velocity vectors, and the relation between Ra number and Nu number for the three geometries will be introduced.

4.3 Numerical Results for Isothermal Vertical Flat Plate

The isothermal vertical plate problem has been solved for air, for surface temperature variation Ts = 320, 340 and 360 K and ambient temperature of 300 K, several solutions are obtained for $1.45\times10^4 \leq Ra \leq 2.6\times10^6$ and for plate heights H = 50, 75 , and 100 mm. To limit the size of the solution domain R_∞ which determine the domain size was taken as 3×H as a first guess. While it is known priori that the temperature values at the nodes near right pseudo and bottom pseudo should approach T_∞, by checking the temperature values at those nodes; the guessed domain size can be determined whether is it enough or not. Step size Δx and Δy were chosen to be equal to each other; 5mm was the first guess, then a smaller step size were chosen, average heat transfer coefficient $\bar{h}$ has been calculated for each step size, then the results were compared, these procedures were repeated until the changes in average heat transfer coefficients between two successive step sizes are negligible. Table 4.1 shows the average heat transfer coefficients for step sizes = 5, 2.5 and 2 mm for an isothermal vertical plate at Ts = 320 K, and T_∞ = 300 K. The step size 2.5 was chosen, while the effect of decreasing it has a negligible effect.

Table 4.1 The Effect of Step Size on Average Heat Transfer Coefficient

Step Size (mm)	$\bar{h}$	Error %
5	9.451961	-
2.5	10.73049	13.5
2	10.50569	2.1

Figure 4.1 shows the lines of constant temperature around an isothermal vertical plate, H= 50 mm, Ts = 320 K in an ambient temperature of 300 K. The lines are closest together near the flat surface, indicating a higher temperature gradient in the region.

Figure 4.2 represents the stream lines near a vertical heated plate under the conditions specified above, it shows that the flow is perpendicular to the boundaries, closer lines indicates higher velocities at that region.

Figure 4.3 shows the profile of y-component of velocity near isothermal vertical plate H = 50 mm, in air at Ts = 320 K. While, in Figure 4.4 velocity vectors near an isothermal vertical plate are represented, it shows the velocity is higher at the region closest to the plate, the fluid particles are moving up because of the buoyancy forces effect, the velocities at the pseudo boundaries at the right and bottom are approximately zeros.

Figure 4.5 shows the variation of local heat transfer coefficients h(y) along an isothermal vertical plate with H = 50 mm, surface temperature Ts = 320 K, and the ambient temperature T_∞ = 300 K, it shows that while y is increasing the value of h is decreasing and at the node at the leading edge of the plate it is approaching to infinity which agrees the conclusions in references [1, 34].

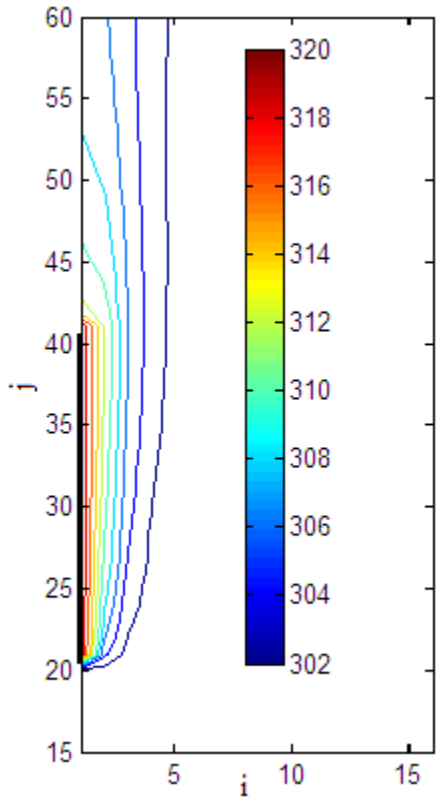

Figure 4.1 Temperature profile near a heated vertical plate at Ts = 320 K, T_∞ = 300 K, Pr = 0.726, and H =50 mm.

In the figure above and the figures below; the axis are representing the index, i is the position of the node in the x-direction and j is the position of the node in the y-direction, while the distance between the nods in both directions is 2.5 mm. For example the black thin line in Figures 4.1, 4.2 and 4.4 is representing the position of the isothermal plate, which located between j = 21and j = 41, so H = 2.5× (41-21) = 50mm.

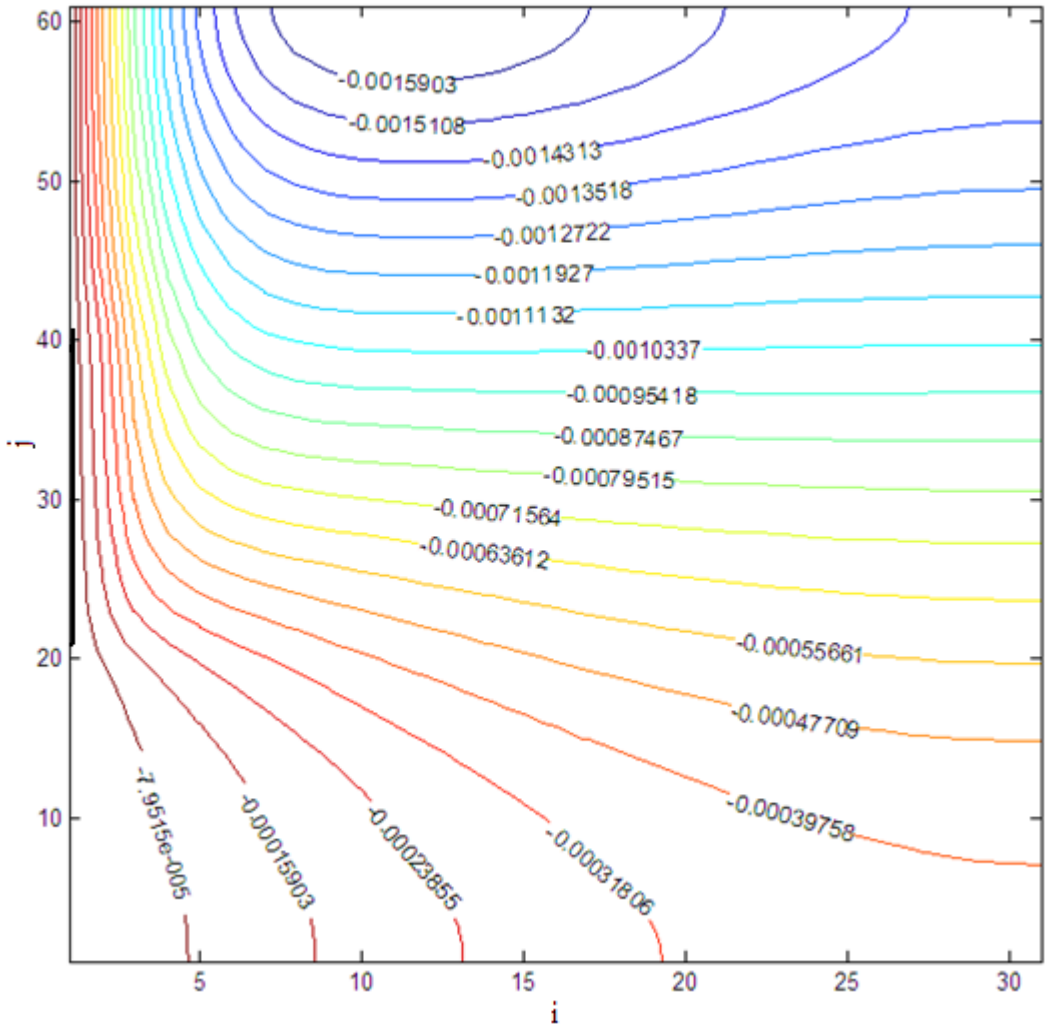

Figure 4.2 Streamlines near a heated vertical plate at Ts = 320K, T_∞ = 300 K, Pr = 0.726, and H =50mm.

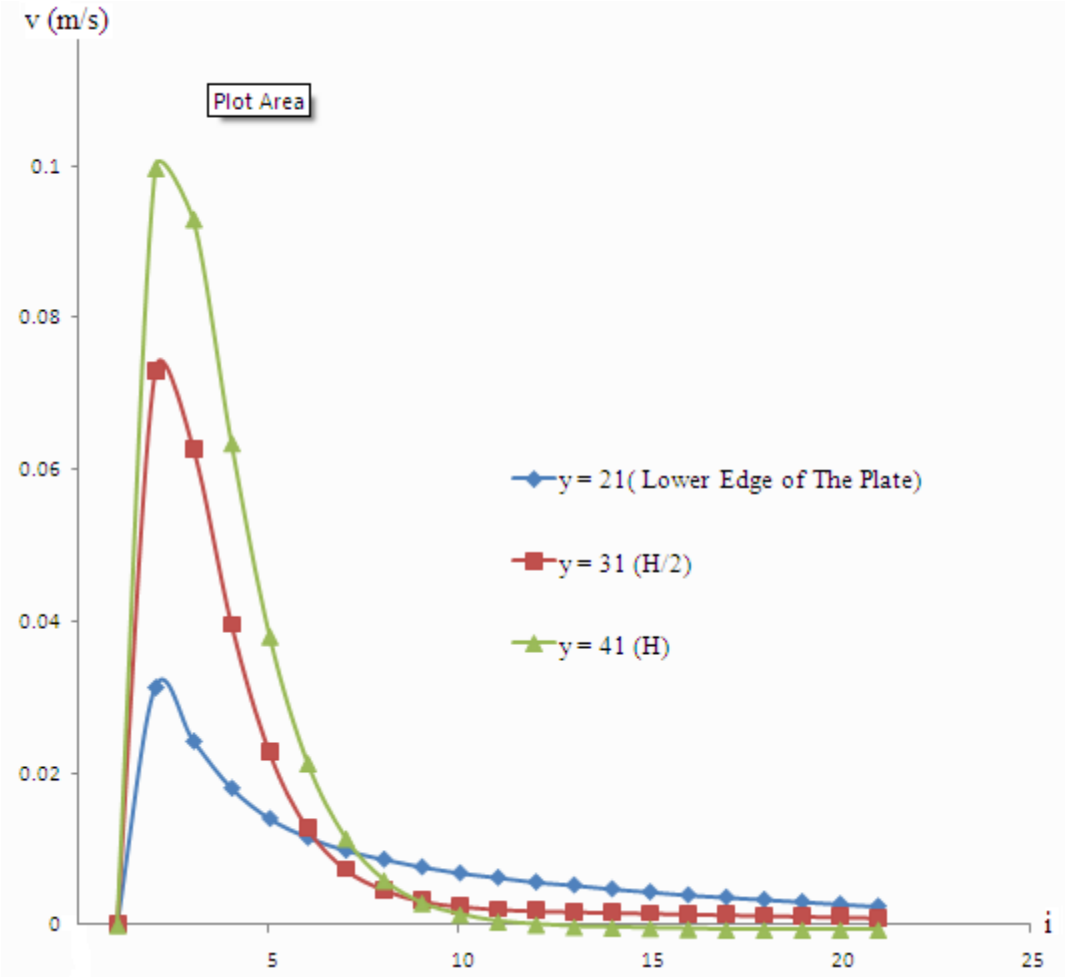

Figure 4.3 Velocity profile for flat plate with H = 50 mm at Ts = 320 K in air at T_∞ = 300 K along the y-axis.

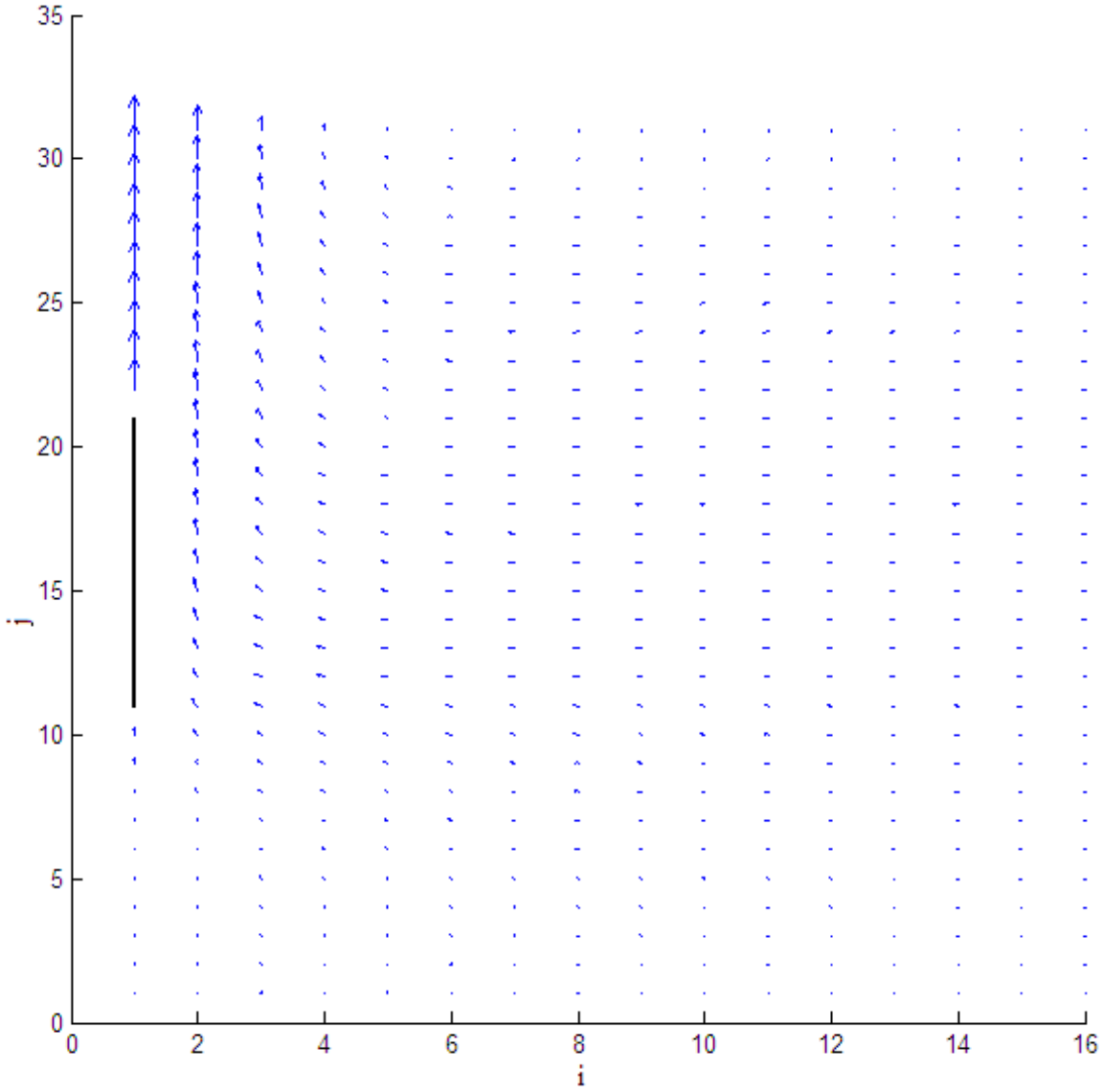

Figure 4.4 Velocity vectors near a heated vertical plate at Ts = 320K, T∞ = 300 K, Pr = 0.726, and H =50mm.

In the figure above, steps size Δx = Δy = 5mm, the plate is located between j = 11, and j = 21, which represents a vertical plate with height H = 50 mm.

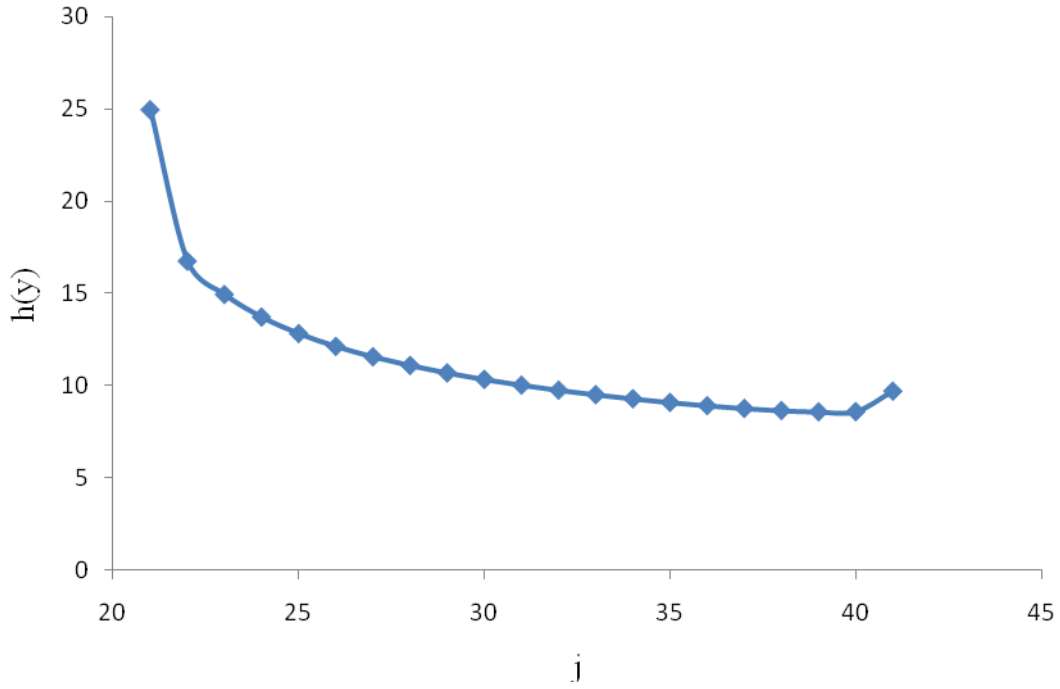

Figure 4.5 Variation of local heat transfer coefficients h(y) along a heated vertical plate at Ts = 320 K, in air at T_∞ = 300 K, an H = 50mm.

The simple empirical correlations for the average Nusselt number Nu in natural convection, especially for the simple geometry as the vertical flat plate, are in the form

$$\overline{\mathrm{Nu}} = CRa^{n} \tag{4.8}$$

The values of the constants C and n depend on the geometry of the surface and the flow regime.

Nusselt number of the numerical results are plotted versus Rayleigh numbers for various heights of the vertical plate H = 50, 75 and 100 mm in Figure. 4.5 on a normal scale, the data presented in Figure 4.5 are correlated using Nu, Ra and H, using CurveExpert 1.3 program the data was fitted, and the best fitting curve through these data is obtained as:

$$\overline{Nu} = 0.5762Ra^{0.2541} \qquad 1.45 \times 10^{4} \leq \mathrm{Ra} \leq 2.6 \times 10^{6} \tag{4.9}$$

A comparison between the results from the numerical solution, results using Eq. 4.9, and the results using Eq. 4.10 from references [1, 34] has been done as shown in Fig. 4.7.

$$\overline{Nu} = 0.59Ra^{0.25} \tag{4.10}$$

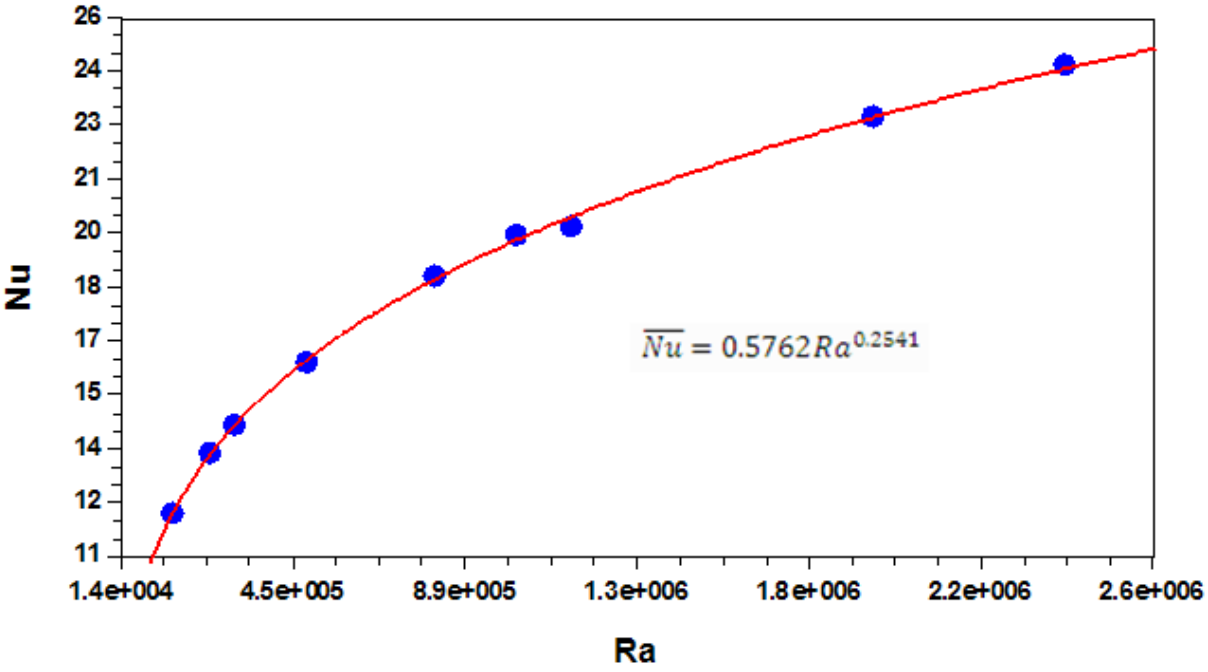

Figure 4.6 Free convection heat transfer correlations for heat transfer from isothermal vertical plate

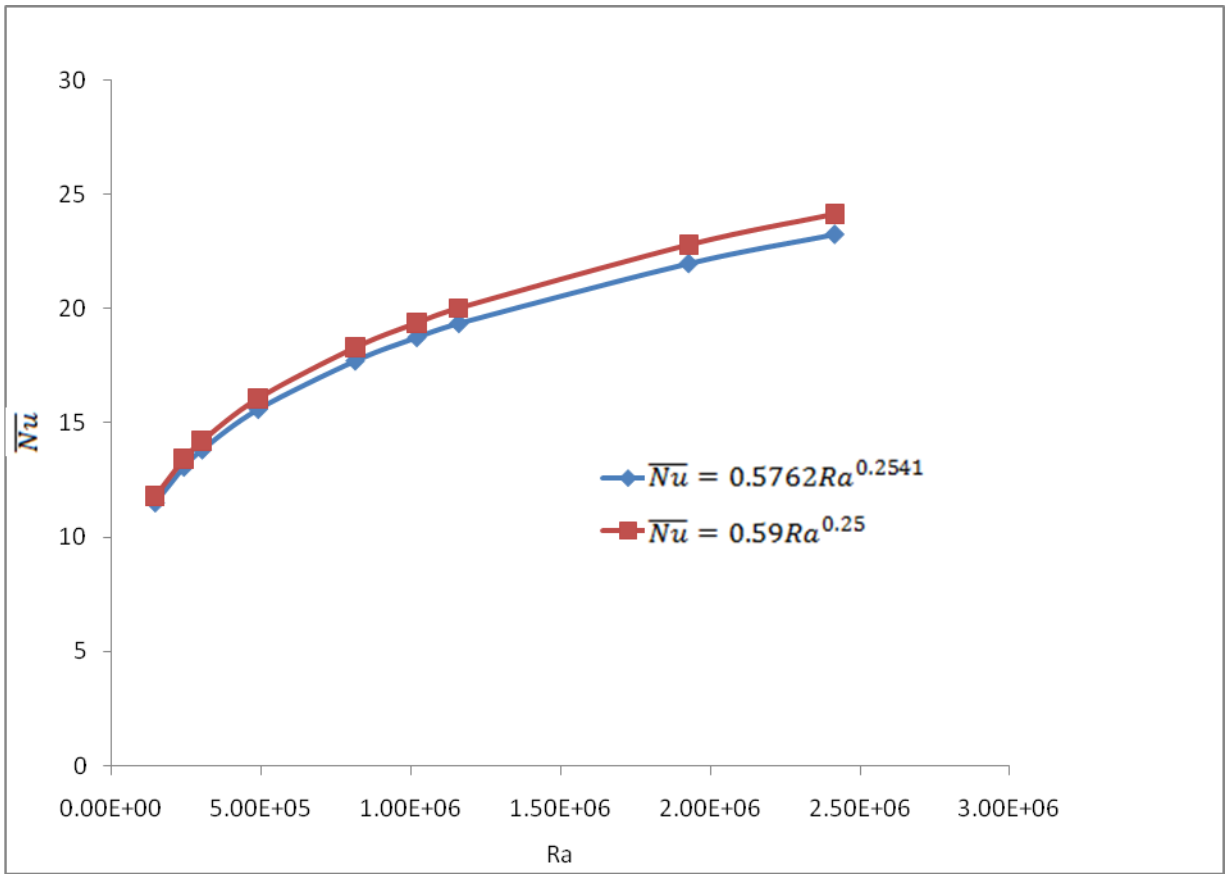

Figure 4.7 Results from Eq. (4.9) vs. result from Eq. (4.10)

4.4 Numerical Results for Horizontal Flat Plate Heated from Both Sides

The isothermal horizontal plate subjected to heat transfer from both sides problem has been solved for air, for surface temperature variation Ts = 320, 340 and 360 K and ambient temperature of 300 K, several solutions are obtained for $1.45\times10^4 \leq Ra \leq 2.6\times10^6$ and for plate widths W = 50, 75 and 100 mm. To limit the size of the solution domain same procedures for vertical plate were followed, also for the step size.

Figure 4.8 shows the isothermal lines around an isothermal horizontal thin flat plate subjected to heat transfer from both sides, Ts = 320 K, ambient temperature T_∞ = 300 K and W = 50 mm. The air streams move inward along the plate from the two opposite edges while trading to rise due to buoyancy. Obviously it can be seen that the farthest inward penetration of the streams is at the symmetric line at x = W/2, where they collide and form an air plume

upward. While the isothermal lines are closest together at the bottom, it can be concluded that; the temperature gradient at the bottom is greater than of the top of the plate.

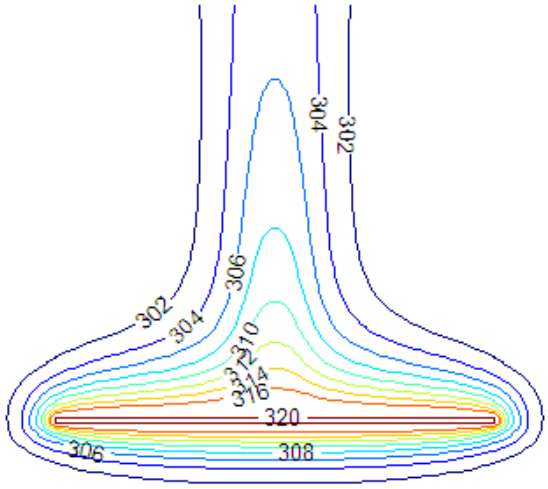

Figure 4.8 Temperature Profile near an Isothermal Horizontal Flat Plate, Both Sides at Ts = 320K, T_∞ = 300 K, Pr = 0.726, and W =50 mm.

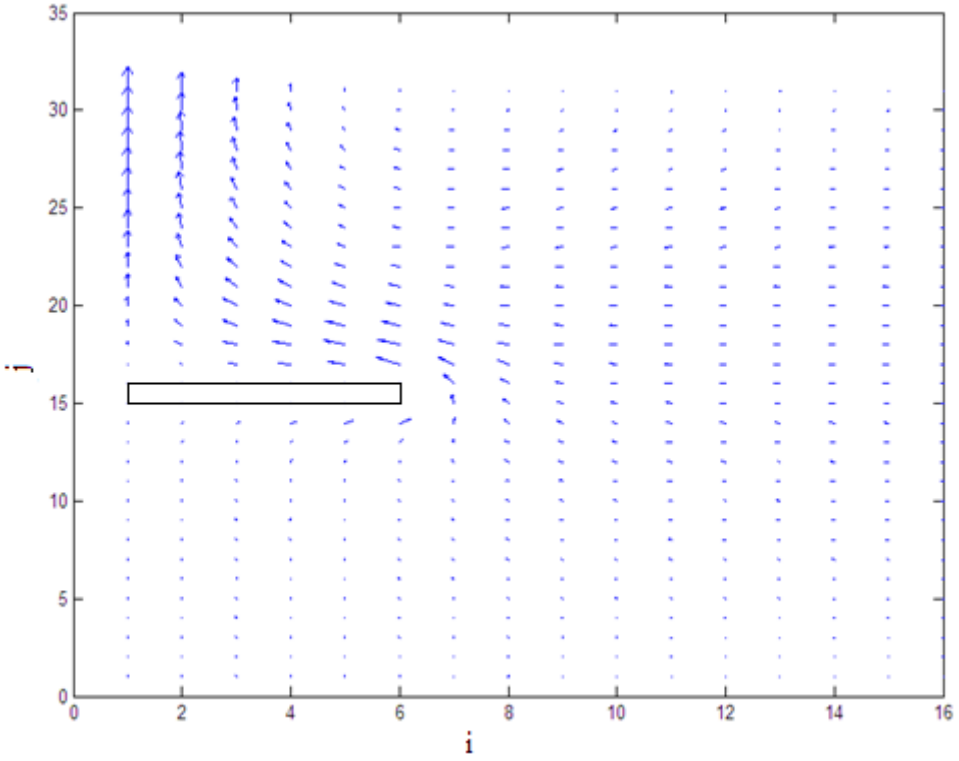

Figure 4.9 Velocity vectors near an isothermal horizontal plate at Ts = 320K, T_∞ = 300 K, Pr = 0.726, and W =50mm.

Figure 4.9 shows the velocity vectors near an isothermal horizontal plate subjected to heat transfer from both sides are represented. It shows that the velocity is higher at the region closest to the upper surface of the plate, since the fluid particles are moving up driven by the buoyancy forces. The velocities at the pseudo boundaries at the right and bottom are approximately zeros.

Figure 4.10 represents the stream lines near a horizontal heated plate under the conditions specified above. It shows closer lines near to the upper surface of the plate, that indicates higher velocities at that region.

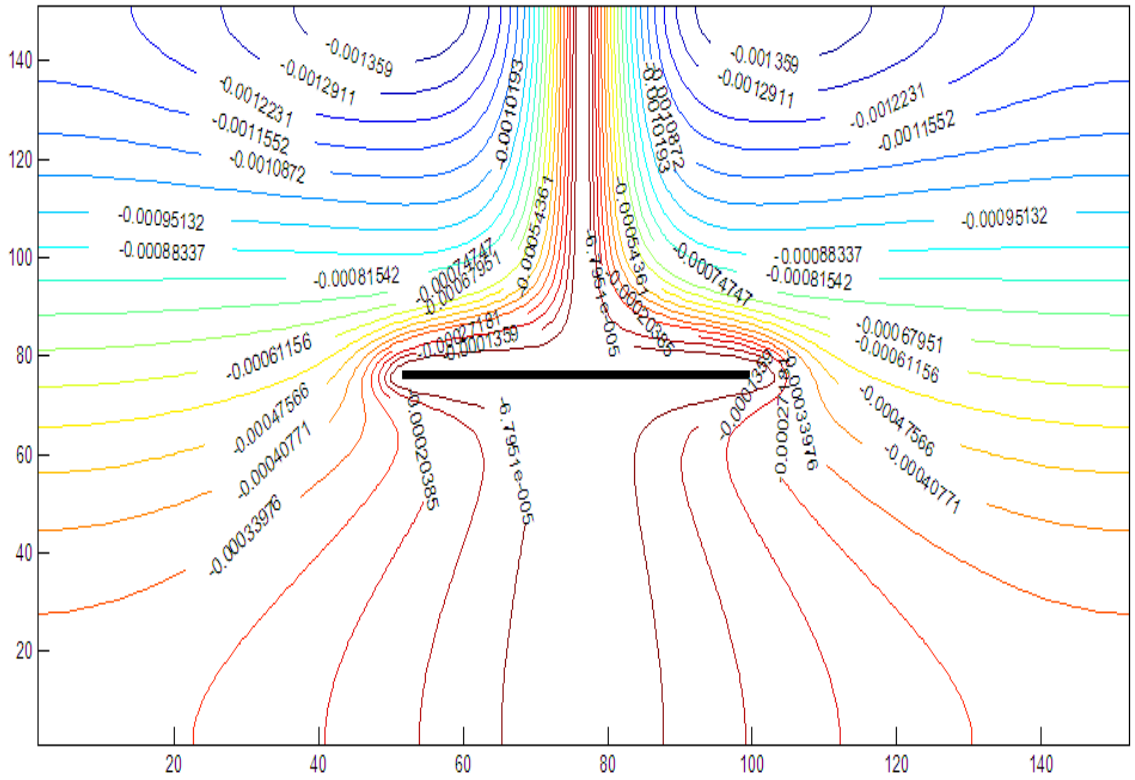

Figure 4.10 Stream lines near a heated horizontal flat plate, both sides at Ts = 320K, T_∞ = 300 K, Pr = 0.726, and W = 50 mm.

Nusselt number of the numerical results versus Rayleigh numbers for various lengths of the horizontal plate W = 50, 75 and 100 mm are plotted in Figure. 4.11 on a normal scale, the data presented in Figure 4.12 are correlated using Nu, Ra, and w, for the lower surface data, and the best fitting curve through these data is obtained as:

$$\overline{Nu}_{lower\ surface} = 0.969\mathrm{Ra}^{0.183} \tag{4.11}$$

For the same conditions, the data presented in Figure 4.13 are correlated using Nu, Ra, and w, for the upper surface data, and the best fitting curve through these data is obtained as:

$$\overline{Nu}_{upper\ surface} = 0.37\mathrm{Ra}^{0.21} \tag{4.12}$$

In Figure 4.11, it can clearly be seen that average Nu number on the lower surface is greater than the corresponding Nu number on the upper surface. The diffrence between lower

and upper Nu numbers decreases with the increase of Ra number. This is dute to the thickening of the upper surface boundary layer generated from the fluid being preheated by the lower surface [15].

The total average Nusselt number versus Rayleigh number depending on the length of the plate (W) including both the upper and lower surfaces is presented in Figure 4.14. The best curve fitting through these data is obtained as:

$$\overline{Nu} = 0.6569Ra^{0.1943} \tag{4.13}$$

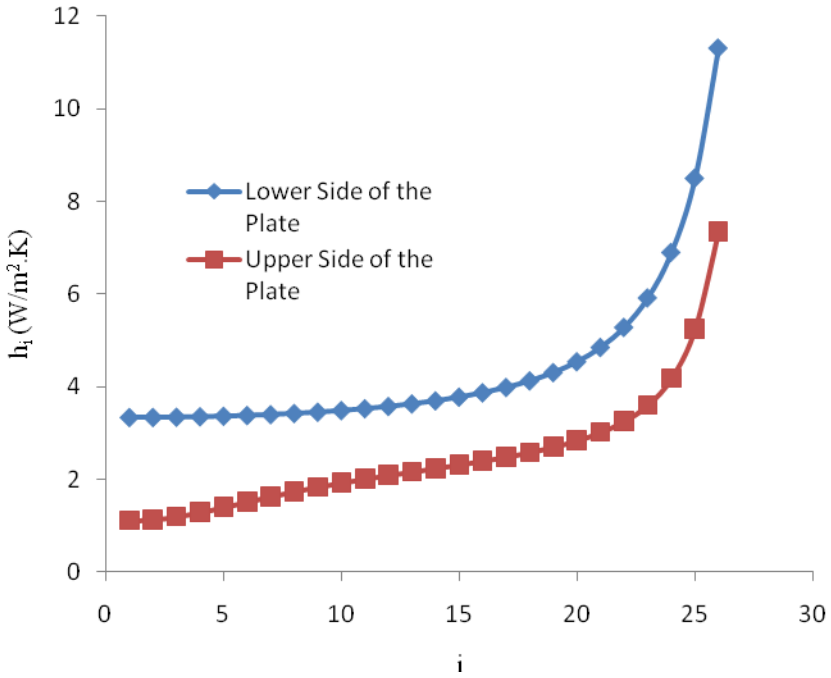

Figure 4.11 Variation of local heat transfer coefficient h(x) at lower and upper surface of a horizontal flat plate at Ts = 320K, T_∞ = 300K, and W = 50mm.

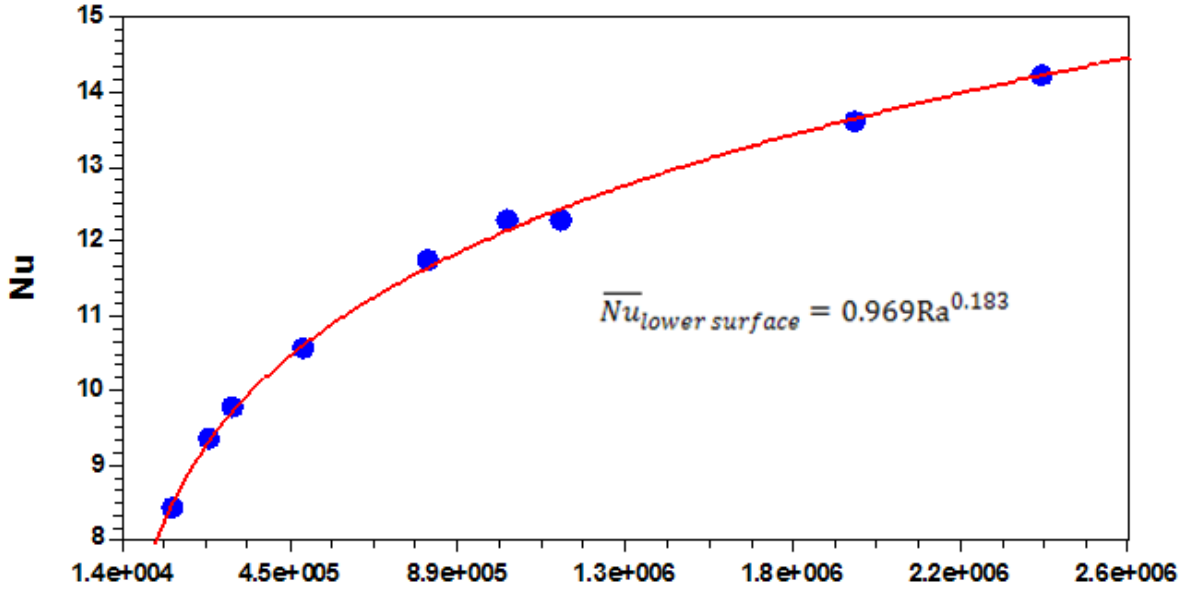

Figure 4.12 Free convection heat transfer correlations for lower surface of heated horizontal plate from both sides

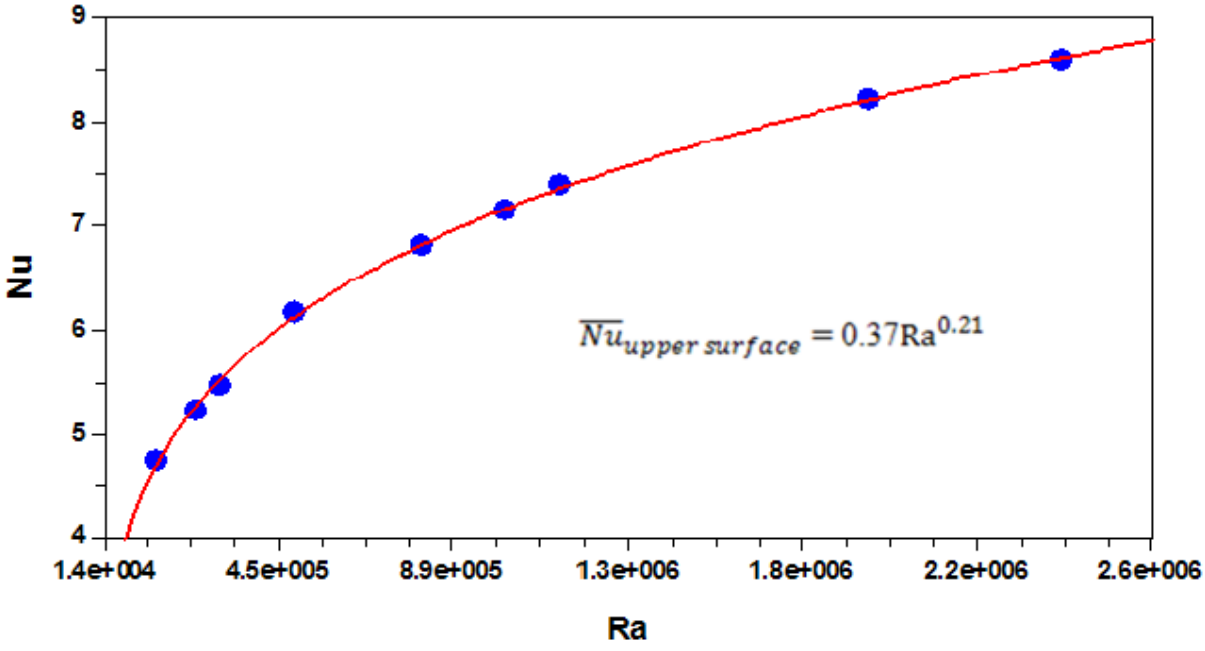

Figure 4.13 Free convection heat transfer correlations for upper surface of heated horizontal plate from both sides.

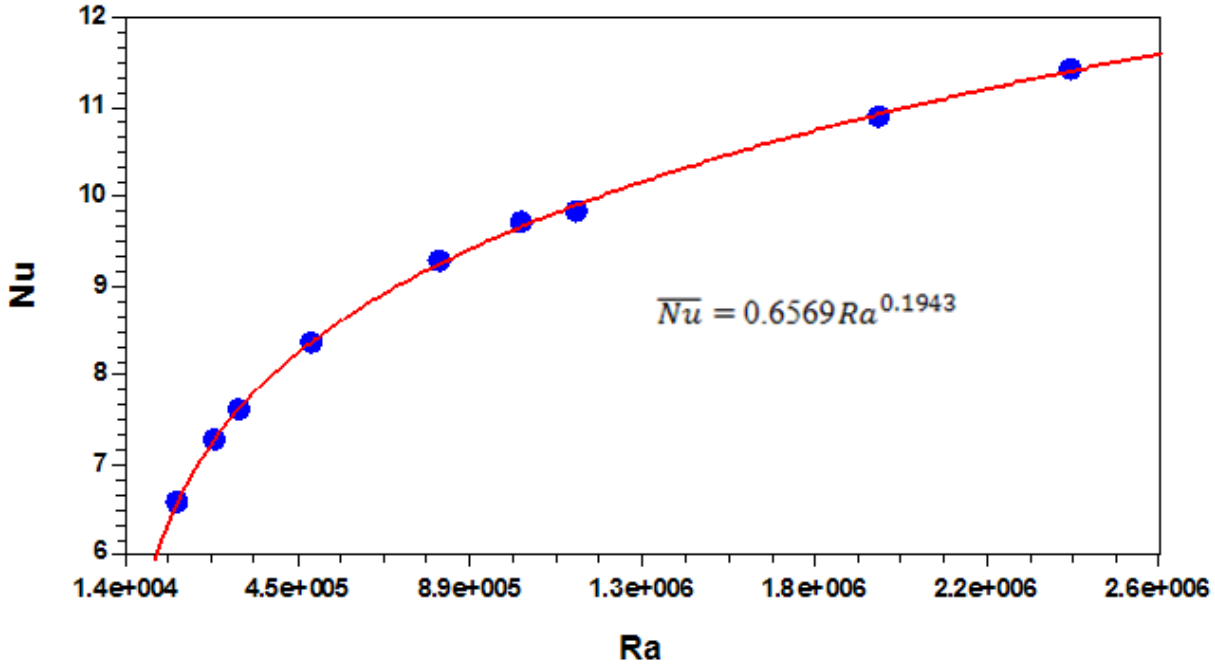

Figure 4.14 Total average free convection heat transfer correlation for a heated horizontal plate from both sides.

4.5 Numerical Results for Horizontal Isothermal Rectangular Duct

The problem of laminar convective heat transfer over a rectangular duct has been solved for air, and the run conditions are summarized in Table 4.2. Several solutions are obtained for $555 \leq Ra_{Lc} \leq 9.6\times10^{6}$, where the calculations of Ra number done using the characteristic length $L_c = W+L$, and for aspect ratios $\Gamma = 0.2, 0.5, 1, 2, 3, 4,$ and 5.

To limit the flow domain R_∞ were chosen as a first guess according to the width of the duct in such a way that $R_\infty = 3\times W$, the step size for all cases were chosen to be $\Delta x = \Delta y = 1$mm, except in Figure 4.22 the data were based on step size 5mm.

Table 4.2 Aspect Ratios Used as Run Conditions

Γ=H/W	W(mm)	H(mm)	T_w (K)	T_∞(K)
0.2	100	20	301 310 320 340 360 400	300K
	75	15		
	50	10		
0.5	20	10		
	80	40		
	100	50		
1	10	10		
	50	50		
	60	60		
	75	75		
2	10	20		
	40	80		
	50	100		
3	90	30		
4	10	40		
	20	80		
	25	100		
5	100	20		

Samples of isotherms and streamlines for different aspect ratios are represented, Figure 4.15, Figure 4.17, and Figure 4.19 show the isotherms for Ra = 1.2×10^5 (ΔT = 20 K) and for aspect ratio 0.2, 1, and 5 respectively. It can be seen in these figures that the isotherms hug the surface boundary at the bottom and vertical sides of the surface, where the thermal boundary layer thickness is thinner than it is at the upper surface of the body. These figures also show the effect of the aspect ratio on the isotherm especially at the upper surface of the duct, it can be observed that ; while the aspect ratio is increasing the isotherms near the middle of the upper surface start to concave down , and this is because of the circulation due to separation appears by increasing the aspect ratio near the top surface of the body ,which obviously can be seen in Figure 4.18 which represents the stream lines around a rectangular duct with aspect ratio of 5. This agrees with the results in Zeitoun and Ali [25].

Figures 4.16, 4.18, and 4.20 show the stream lines under the same conditions for the isotherms respectively. Those figures; show that the separation starts to appear at the upper surface of the duct causing the circulation near this surface.

The effect of the hot edges (upper and lower horizontal edges) on the y-component of the velocity is examined in Figure 4.21, the step size used to get the results in Figure 4.21 is 2.5 mm. The figure shows the y-component of the velocity near the vertical side of a rectangular duct with an aspect ratio of 1 and height H = 50 mm heated to Ts=320 K. Comparing the results with those for the flat plate, higher velocities are obtained in the case of the rectangular duct. This is due to the hot air arising from the lower plate which enhances the motion of air near the vertical wall, which, can be noticed from the behavior of the hot air around an isothermal rectangular duct given in Figure 4.22.

In Figure 4.22 the velocity vectors around a rectangular duct with aspect ratio of 1 in air, and the surface temperature is 320 K. It can be seen that the hot air is moving up driven by the buoyancy forces.

Figure 4.23 shows the variation of the local heat transfer coefficient for a square duct with length 50 mm. For the vertical wall the maximum value of the local heat transfer coefficient is at the leading edge of the wall. Also it can be seen that higher values of the local heat transfer coefficients at the lower surface of the duct.

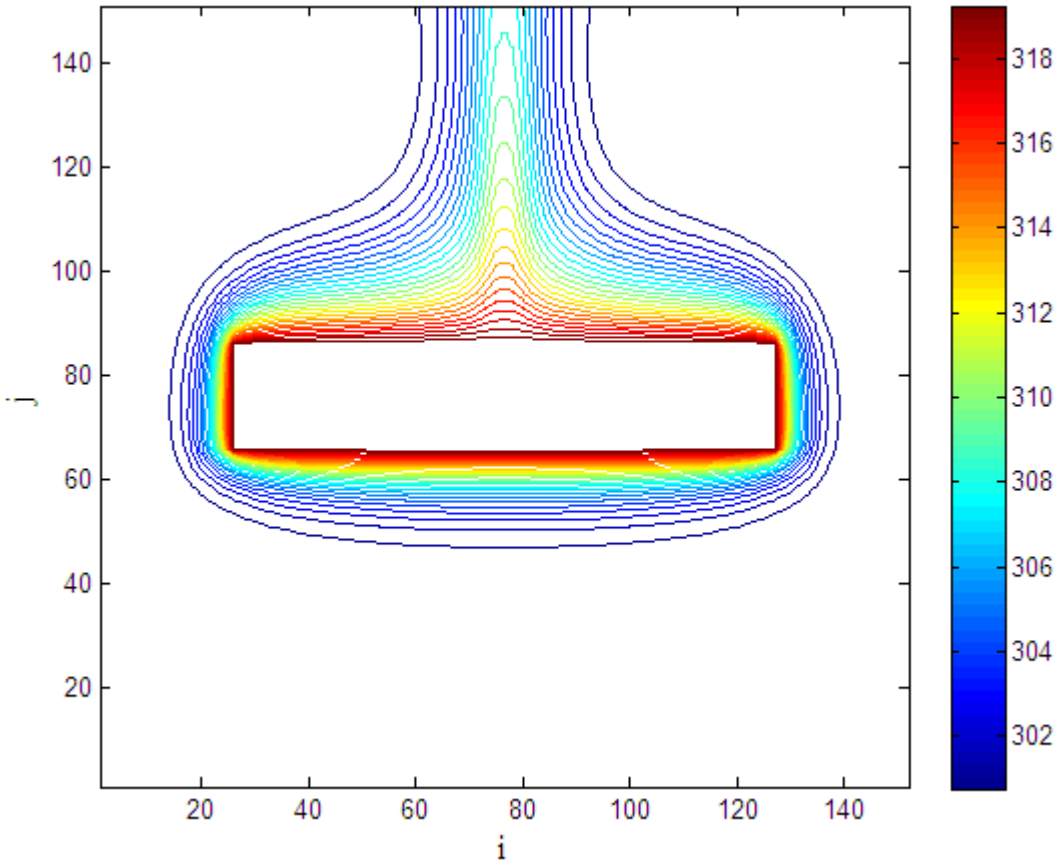

Figure 4.15 Isothermal Lines Over a Rectangular Duct with = 0.2, H = 20 mm, Ts =320 K, in Air at T_∞ = 300 K.

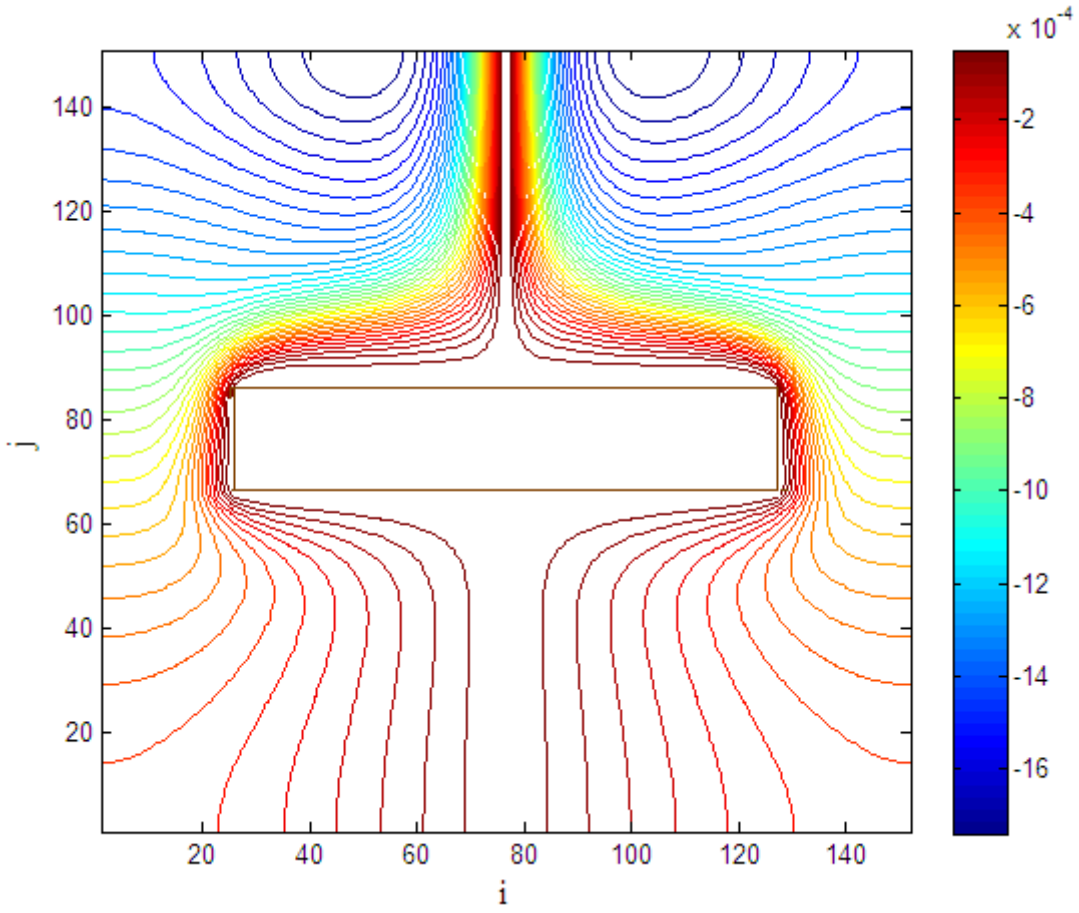

Figure 4.16 Streamlines over a rectangular duct with = 0.2, H = 20 mm, Ts=320 K, in air at T_∞ = 300 K.

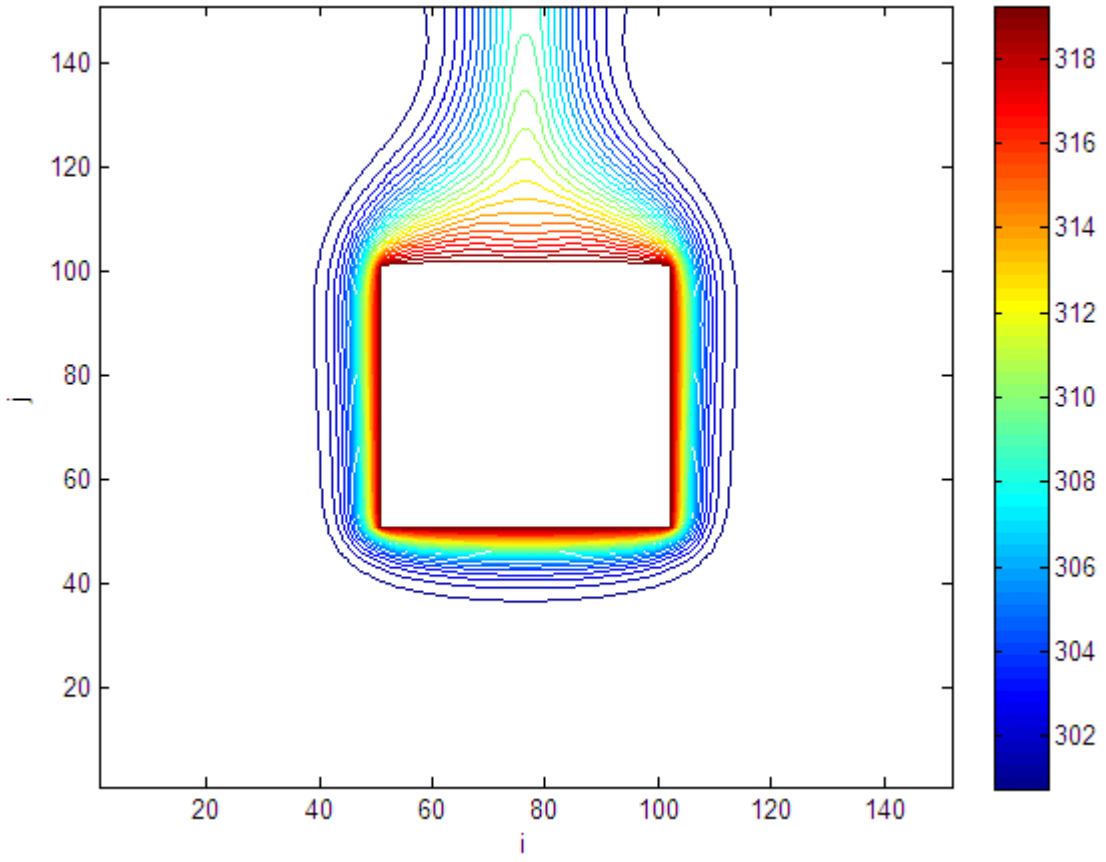

Figure 4.17 Isothermal lines over a rectangular duct with = 1, H = 50 mm, Ts=320 K, in air at T_∞ = 300 K.

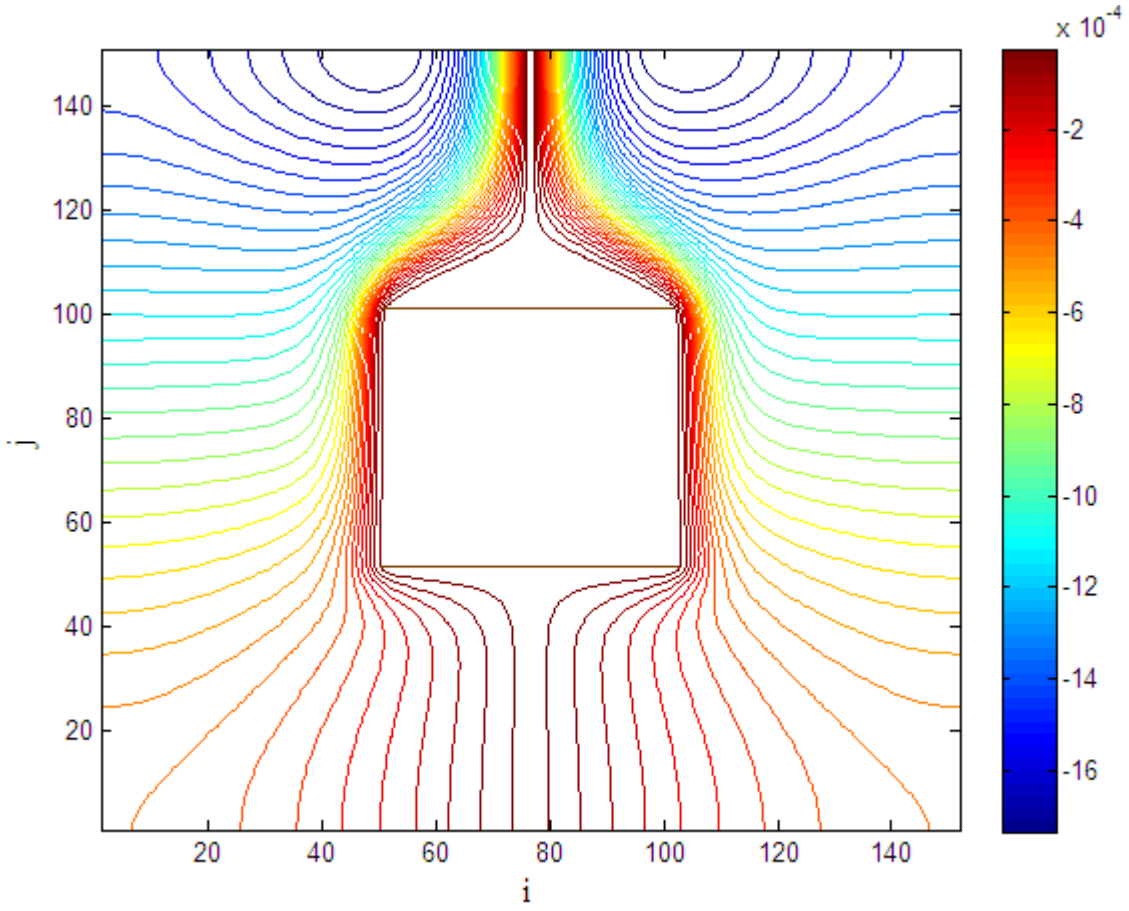

Figure 4.18 Streamlines over a rectangular duct with = 1, H = 50 mm, Ts=320 K, in air at T_∞ = 300 K.

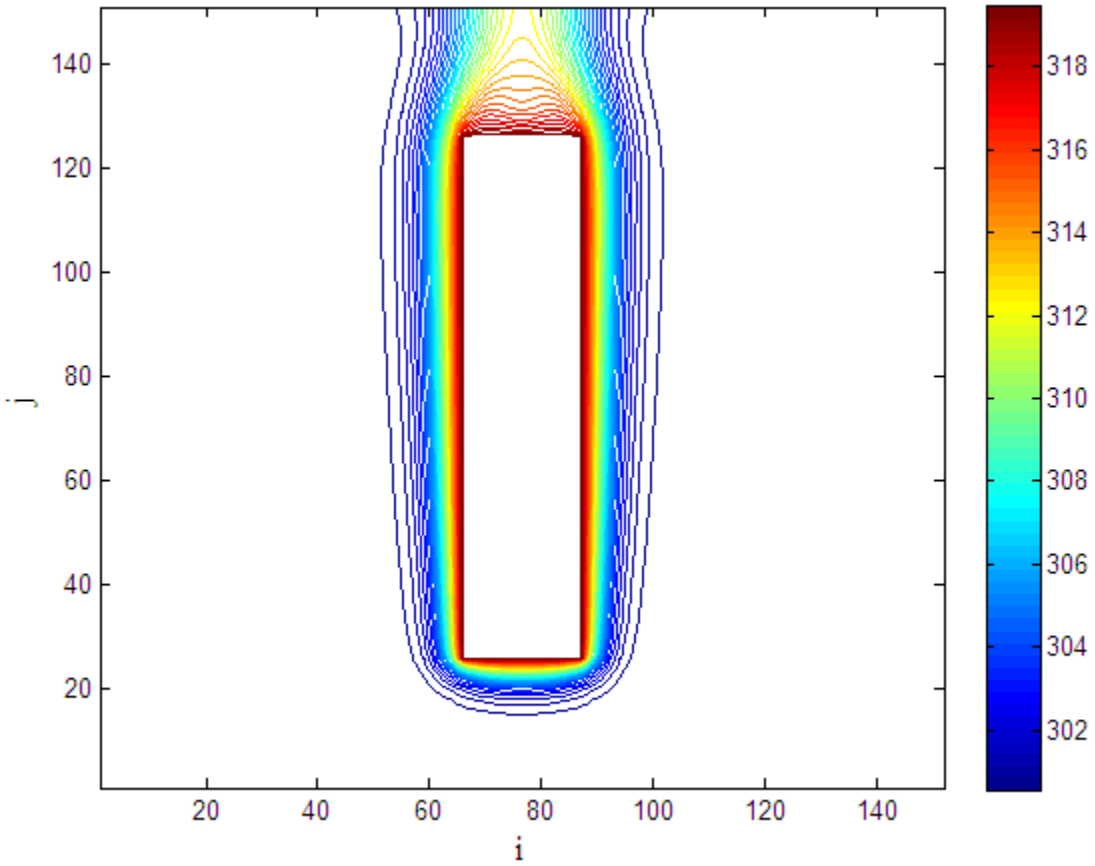

Figure 4.19 Isothermal lines over a rectangular duct with = 5, H = 100 mm, Ts=320K, in air at T_∞ = 300 K.

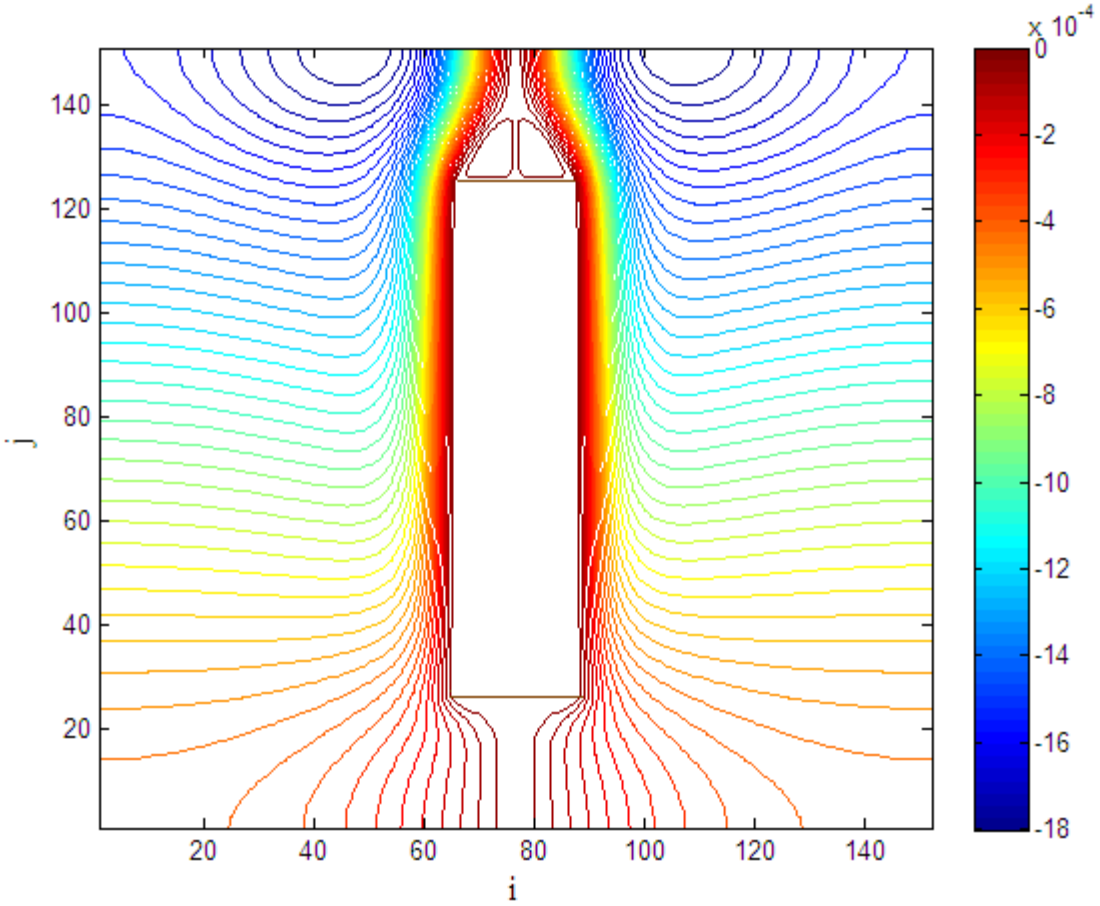

Figure 4.20 Streamlines over a rectangular duct with = 5, H = 100 mm, Ts=320 K, in air at T_∞ = 300 K.

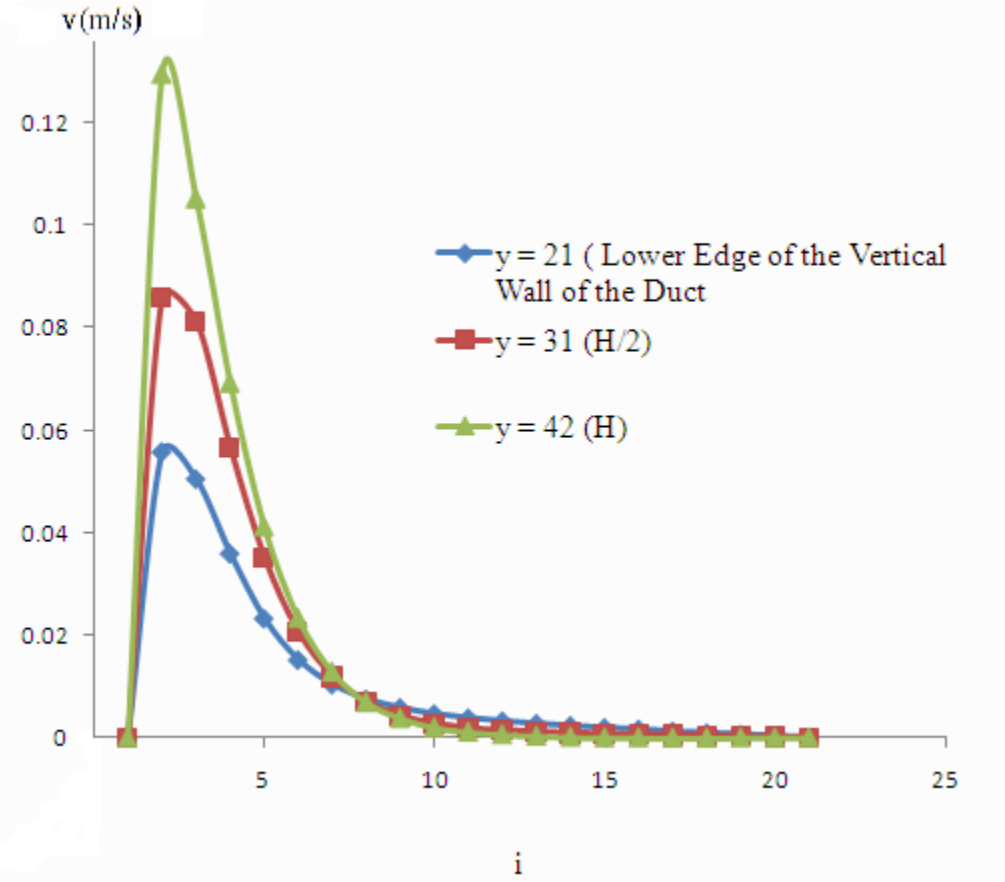

Figure 4.21 Velocity profile for rectangular duct with H = 50 mm, and L = 50 mm, at Ts= 320 K along the y-axis.

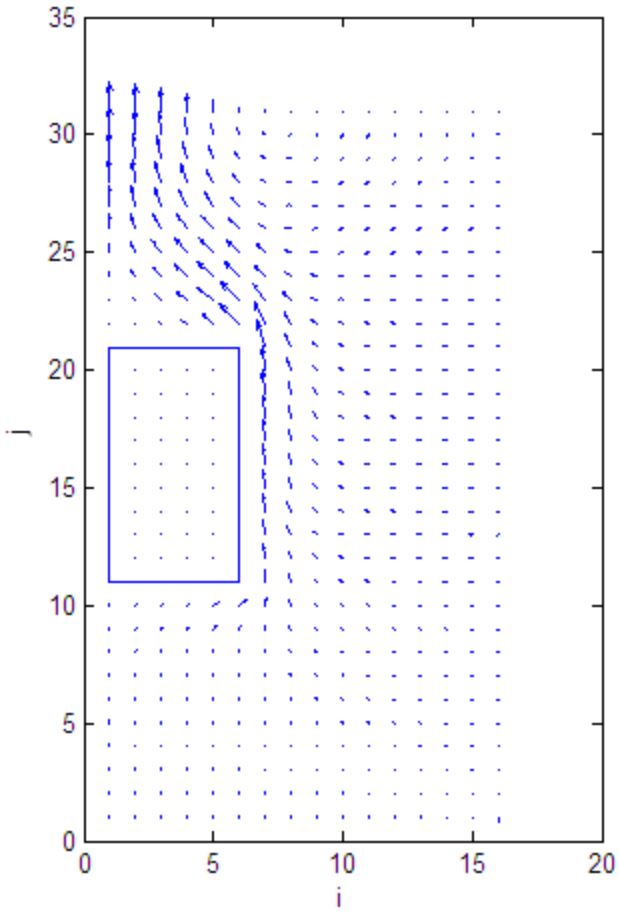

Figure 4.22 Velocity vectors near a heated rectangular duct with = 1 at Ts = 320K

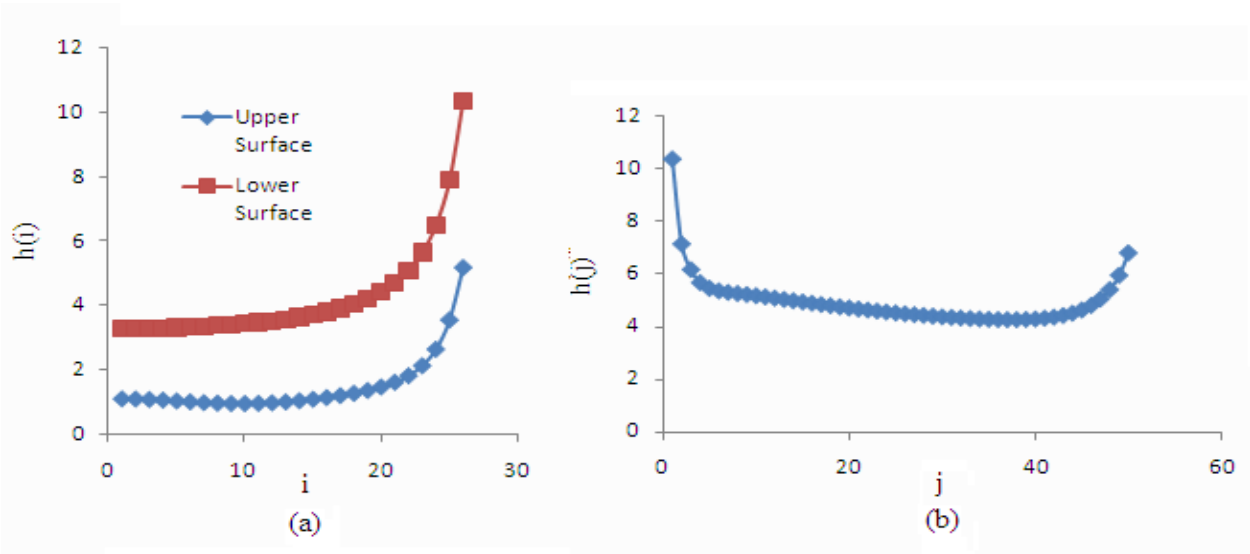

Figure 4.23 Local heat transfer coefficients for square duct, (a) at upper and lower surfaces and (b) at the vertical wall of the duct

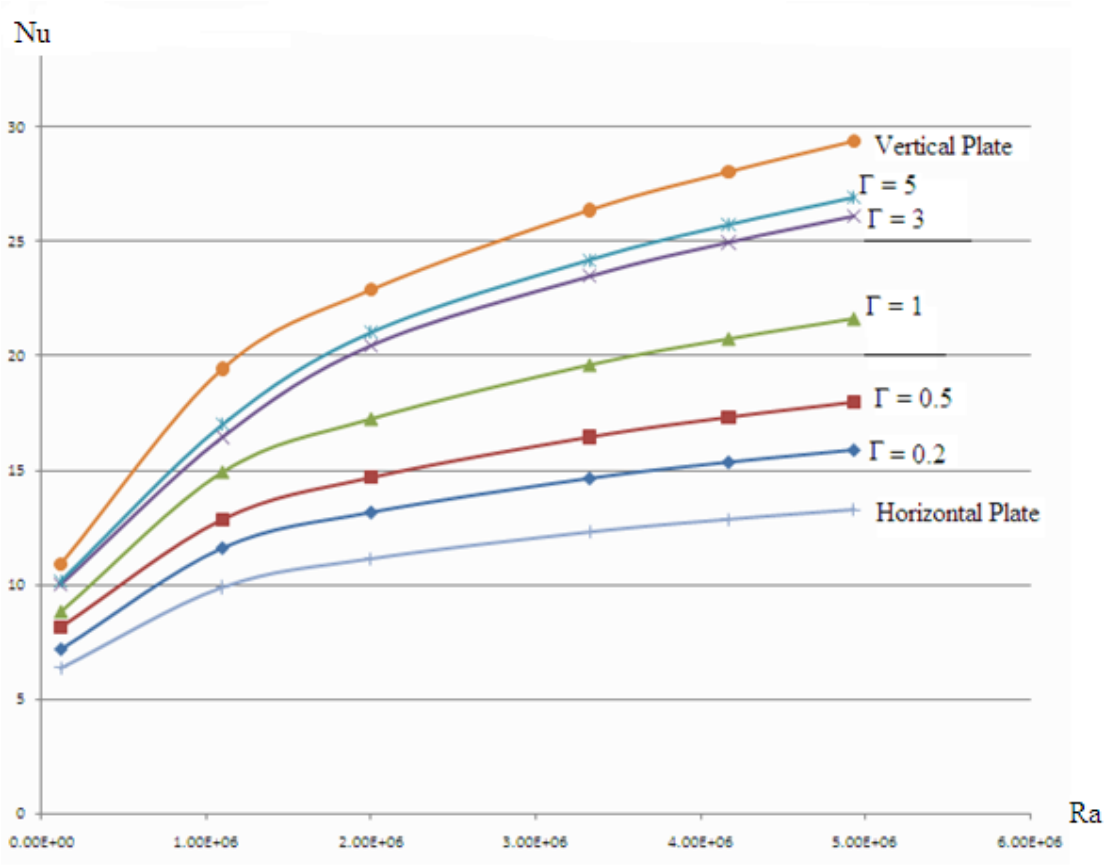

Figure 4.24 Nusselt number vs. Raleigh number for different aspect ratios of rectangular duct, vertical plate, and horizontal plate.

Figure 4.24 shows the effect of increasing aspect ratio on the average Nusselt number, it can be seen that; increasing the aspect ratio for fixed characteristic length is increasing average Nusselt number, for high aspect ratios it seems to be approaching the vertical plate solution, and approaching to horizontal flat plate for low aspect ratios.

To correlate the results; Raleigh number and the average Nusselt number were based on (H+W) as a characteristic length Lc. For various aspect ratios; Nusselt number was correlated as function of Ra number in the form of power equation $\overline{Nu} = CRa^n$, in such a way that the value of Lc is same for all geometries, and then the constant coefficient C and the power coefficient n were represented as function of the aspect ratio.

Under these considerations; the characteristic length L_c is chosen as 120 mm for different aspect ratios, and for surface temperature Ts = 301, 310, 320, 340, 360 and 400 K. The corresponding Nu numbers obtained numerically are shown in Table 4.2.

Table 4.3 Average Nusselt Number versus Rayleigh Number at various Aspect Ratios of Horizontal Ducts

		Nu					
Ts(K)	Ra(L+W)	Γ=0.2	Γ=0.5	Γ=1	Γ=3	Γ=5	Γ=11
301	119780.6	7.177384882	8.173136	8.809155	9.990177	10.18038	10.89241
310	1099560	11.59046237	12.87545	14.906	16.43	16.99983	19.40155
320	2001628	13.14631694	14.70356	17.23159	20.4361	21.01803	22.8576
340	3327920	14.62449153	16.47318	19.59169	23.45932	24.17951	26.3351
360	4170472	15.33045448	17.33777	20.72913	24.94292	25.71422	28.00217
400	4930639	15.86264394	17.99602	21.61358	26.0875	26.90121	29.34088

For each geometry, Ra numbers versus Nusselt numbers were plotted and the coefficients C and n were obtained. The effect of aspect ratio on these coefficients detected as in Table 4.3. These coefficients were represented as functions of Γ, so that average Nusselt number for any Γ can be in following form :

$$\overline{Nu} = C(\Gamma)Ra^{n(\Gamma)} \tag{4.14}$$

Table 4.4 Coefficients (n) and (C) as Function of Aspect Ratios

Γ	L	H	C	n
0.2	100	20	0.601056	0.212506
0.5	80	40	0.654456	0.214817
1	60	60	0.507754	0.243245
3	30	90	0.392732	0.272039
5	20	100	0.393238	0.273978
11	10	110	0.460746	0.269372

The best curve fitting of the these data has been obtained, which enables to correlate average Nusselt number as function of Ra number and the aspect ratio.

$$\overline{Nu} = \left(0.6362 - 0.071798\Gamma + 0.0051\Gamma^2\right) Ra^{\left(\frac{0.206 + 0.71797\Gamma}{1 + 0.19\Gamma + 0.00488\Gamma^2}\right)} \quad (4.15)$$

Correlation 4.15 was examined for the results from most of the numerical results; the correlation is satisfying the numerical results with a maximum error 10.34%. Sample of the comparison is shown in Fig. 4.25.

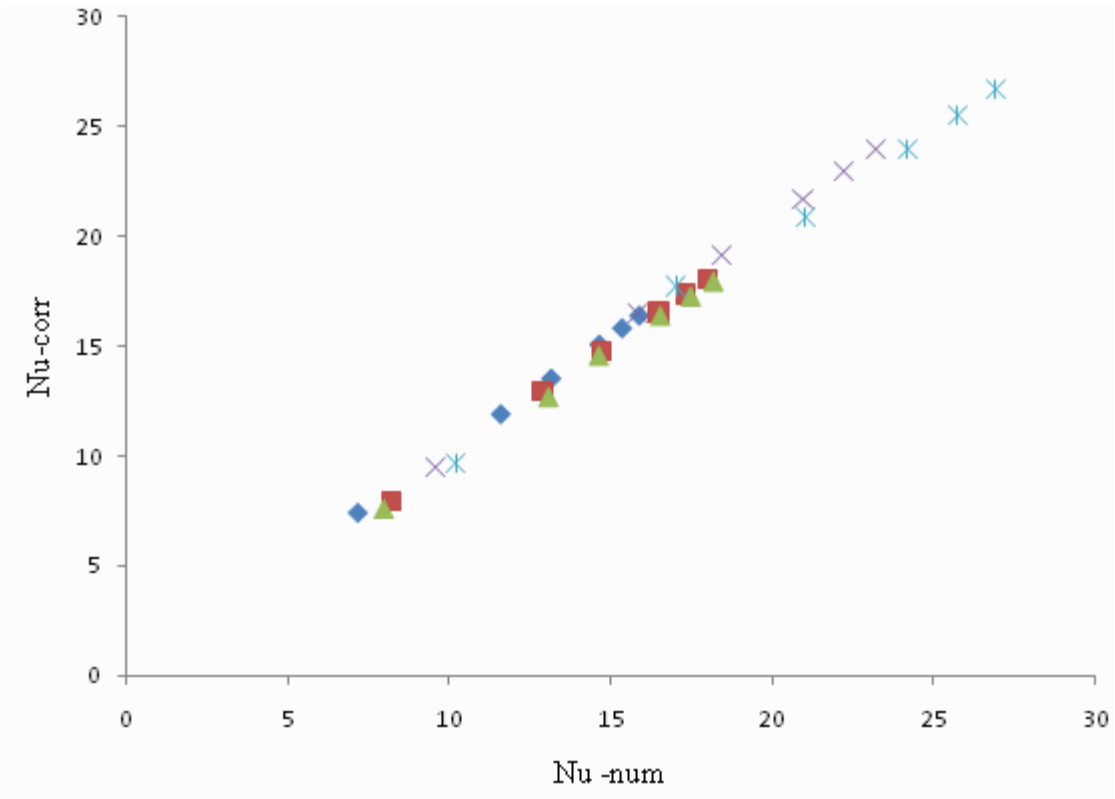

Figure 4.25 Correlated Nusselt number from Eq. (4.15) vs. numerically computed Nusselt number.

CONCLUSIONS

Natural convection from isothermal vertical plates, horizontal flat plats heated from both sides, and isothermal horizontal ducts has been investigated numerically for ranges of Rayleigh numbers and various aspects ratios for the ducts. Streamlines show that for fixed Rayleigh number, as the aspect increases, separation and circulation occurs above the top surface of the duct, and the corresponding behavior is observed through the isotherm which starts to concave down near to the top surface of the duct. The effect of the upper and lower surfaces of the duct on the velocity in y-direction at the vertical side of the duct, comparison between a duct with aspect ratio $\Gamma = 1$, and flat plate with same height, the results show that the y-component of velocity near the duct is greater than those near the flat plate. The total heat transfer from circumference of the duct was obtained and the average heat transfer coefficients put in dimensionless form of Nusselt number for various aspect ratios. Correlations for the three geometries covering wide range of Raleigh numbers were obtained, a comparison for the case of the vertical plate has been done, and it shows a good fitting with the correlation valid in the literature.

Finally, for a future works the methodology adopted can be used to solve the problem of inclined ducts; also it can be used to obtain a solution for a vertical array of horizontal isothermal ducts.

LIST OF REFERENCES

[1] J.P Holman, "*Heat Transfer*," McGraw-HILL, Tokyo, Edition 4, 1976.

[2] Eckert, E. R. G., and E. Soehnegen, "*Interferometric Studies on the Stability and Transition to Turbulence of a Free Convection Boundary Layer*," J. Heat Transfer, ASME-IME, London, 1951.

[3] Vliet, G. C., and C. K. Lin, "*An Experimental Study of Turbulent Natural Convection Boundary Layer,* " J. Heat Transfer , vol. 91 , p. 517 , 1969.

[4] Cheesewright, R., "*Turbulent Natural Convection from Vertical Surface,* " J. Heat Transfer, vol. 90, p.1, 1968.

[5] Flack, R. D., and C. L. Witt, "*Velocity Measurements in Two Natural Convection Air Flows Using a Laser velocimeter,* " J. Heat Transfer, Vol. 101, p.256, 1979.

[6] K. D. Jensen, "*Flow Measurements*," ENCIT2004 – 10th Brazilian Congress of Thermal Sciences and Engineering, Nov.2003.

[7] Stuart W. Churchill, Humbert H. S. Chu, "*Correlating equations for laminar and turbulent free convection from a vertical plate,* " International Journal of Heat and Mass Transfer vol.18(1975), p.1323-1329.

[8] Stuart W. Churchill, Humbert H. S. Chu, "*Correlating equations for laminar and turbulent free convection from a horizontal cylinder,* " International Journal of Heat and Mass Transfer vol.18 (1975), p.1049-1053.

[9] Stuart W. Churchill, Humbert H. S. Chu, "*The development and testing of a numerical method for computation of laminar natural convection in enclosures,* " Computers & Chemical Engineering, vol.1 (1977), p.101-102.

[10] Dachon Y., and T.Jianjun," *An Experimental Research on the Instability of Natural Convection Boundary Layer around A Vertical Heated Flat Plate*," Acta Mechanica (English Version), Vol.15, Feb.1999.

[11] T. Zitzmann, Malcolm C., and P. Peter "*Simulation of Steady-State Natural Convection Using CFD,* " Ninth International IBPSA Conference Montréal, Canada August 15-18, 2005.

[12] R. Comunelo and S. Güths, "*Natural Convection at Isothermal Vertical Plate: Neighborhood Influence,* " 18th International Congress of Mechanical Engineering, Ouro Preto, MG, November 6-11, 2005.

[13] Goldstein, R.L., Sparrow, E.M., Jones, D.C., "*Natural Convection Mass Transfer Adjacent to Horizontal Plates*," Int. J. Heat Mass Transfer, v.16, p.1025-1035, 1973.

[14] Lloyd, J.R., and Moran, W.R., "*Natural Convection Adjacent to Horizontal Surface of Various Plan forms*," J. Heat Transfer, v.96, p. 443-447, 1974.

[15] J. J. Wei, B. Yu, H. S. Wang and W. Q. Tao, " *Numerical Study of Simultaneous natural Convection Heat Transfer From Both Surfaces of a Uniformly Heated Thin Plate With Arbitrary Inclination*," Heat and Mass Transfer , vol.38, p. 309-317,2002.

[16] Pera, L., and Gebhart, B., "*Natural Convection Boundary Layer Flow over Horizontal and Slightly Inclined Surfaces*," Int. J. Heat Mass Transfer, v.16, p.1131-1146, 1973.

[17] Pera, L., and Gebhart, B., "*On the Stability of Natural Convection Boundary Layer Flow over Horizontal and Slightly Inclined Surfaces*," Int. J. Heat Mass Transfer, v.16, p. 1147-1163, 1973.

[18] Md .Zakerullah and J. A. D. Ackroyd, "Laminar *Natural convection Boundary Layers on Horizontal Circular Disks*, "Journal of Applied Mathematics and Physics, vol.30, 1979.

[19] C. J. Kobus and G. L. wedekind, "*An Experimental Investigation into Natural Convection Heat Transfer from Horizontal Isothermal Circular Disks*", Int. J. Heat Mass Transfer, vol. 44, p. 3381-3384, 2001.

[20] Al-Arabi, M., and El-Riedy, M.K., "*Natural Convection Heat Transfer from Isothermal Horizontal Plates of Different Shapes*," Int. J. Heat Mass Transfer, v.19, p. 1399-1404, 1976.

[21] Md. Mamun Molla, Md. Anwar Hossain, Manosh C. Paul, "*Natural convection flow from an isothermal horizontal circular cylinder in presence of heat generation*," International Journal of Engineering Science, vol. 44, p. 949–958, 2006.

[22] Massimo Corcione, "*Correlating equations for free convection heat transfer from horizontal isothermal cylinders set in a vertical array*," International Journal of Heat and Mass Transfer, vol. 48 , p. 3660–3673, 2005.

[23] Massimo Corcione, "*Interactive free convection from a pair of vertical tube arrays at moderate Rayleigh numbers*, " International Journal of Heat and Mass Transfer, vol.50, p. 1061–1074, 2007.

[24] C.O. Popiel, J. Wojtkowiak, "*Experiments on free convective heat transfer from side walls of a vertical square cylinder in air*," Experimental Thermal and Fluid Science, vol. 29, p. 1–8, 2004.

[25] O. Zeitoun and Mohamed Ali, "*Numerical investigations of natural convection around isothermal horizontal rectangular ducts*, " Numerical Heat Transfer, vol. 50, p. 189–204, 2006.

[26] S. V. Patankar and D. B. Spalding, "*Heat and Mass Transfer in Boundary Layers, A general Calculation Procedure*," 2nd Ed., Inter text Books, London, 1970.

[27] Gosman, A. D., Pun, W. M., Runchall, A.K., Spalding, D. B., and Wolfstein, M. W., *"Heat and Mass Transfer in Recirculating Flows, "*Academic Press, London, 1969.

[28] Launder, B. E., and Massey, T. H., *" The Numerical Prediction of Viscous Flow and Heat Transfer in Tube Banks, "*ASME J. Heat Transfer, vol.100, p. 329-341, 198

[29] Farouk, B. , and Güçeri, S. I. , "*Laminar and Turbulent Natural Convection Around a Horizontal Cylinder Within Confining Walls ,* "Num. Heat Transfer, vol. 5, p. 329-341, 1982.

[30] Bezzazoğlu, M., "*Laminar Natural Convection Heat Transfer over Tow Vertically Spaced Horizontal Cylinders,* "M.S. Thesis, METU, 1985.

[31] Gövsa, C., "*Laminar Natural Convective Heat Transfer in A Cavity*," M.S. Thesis, METU, 1988.

[32] M. Manzan , G. Comini, and G. Cortella*"A Stream Function-Vorticity Formulation for Spatially Periodic flows, "* Comm. Num. Meth. Engineering vol. 13, p. 867-874-1997.

[34] Yunus A. Çengel, "*Heat Transfer*," McGraw-Hill, Tokyo, Edition 2, 2003.

APPENDIX

```
//COMPUTER PROGRAM
//Constant numerical data

public class Constants
{
        public static final double EXCOEF = 0.003332;
        public static final double GCOEF = 9.81;
        public static final double CC = 0.0003;;
        public static final double RP = 0.8;
        public static final int MAXITR = 200000;
}

//Generating the grid
mport javax.swing.*;
import java.io.*;
import java.awt.*;

public class Proc
{
int iMin, imin, imax, jmax, iup, ibottom, jright, tSurface, tInfinity;
double hght, wdth, delx, dely, domaind, ae, aw, an, ass, a1, a2, a3, a4, dv, ddv, source, rs,
tAvg, muRef, roRef, pr, tk, z;
double [ ] X1, X2, be, bw, bn, bs, rsdu, hup, hbottom, hright;
double [ ][ ] Omega, Psi, Temp, u, v;

        public Proc()
        {
                imin = 1;
                imax = 31;
                jmax = 16;
                ibottom = 10;
                iup = 20;
                jright = 5;
                delx = 0.005;
                dely = 0.005;
                z = 0.1;
                tSurface = 320;
                tInfinity = 300;
                X1 = new double[1000];
                X2 = new double[1000];
                be = new double[1000];
```

```
            bw = new double[1000];
            bn = new double[1000];
            bs = new double[1000];
            hup = new double[1000];
            hbottom = new double[1000];
            hright = new double[1000];
            rsdu = new double[3];
            Omega = new double[1000][1000];
            Psi = new double[1000][1000];
            Temp = new double[1000][1000];
            u = new double[1000][1000];
            v = new double[1000][1000];
        }

public void setParams(int h, int w, double delX, double delY, int domainDiameter, int iMin)
        {
            hght = (double) h / 1000;
            wdth = (double) w / 1000;
            delx = delX / 1000;
            dely = delY / 1000;
            domaind = (double) domainDiameter / 1000;
            jright = (int)(w / (2 * delX));
            imin = iMin;
            imax = (int)(domainDiameter / delY) + 1;
            jmax = (int)(0.5 * domainDiameter / delX) + 1;
            ibottom = (int)((imax - (h / delY + 1)) / 2);
            iup = ibottom + (int)(h / delY);
        }

//Initializing the problem

        public void setTemperature(int x, int y)
        {
            if(x > 0)
            {
                tSurface = x;
            }

            if(y > 0)
            {
                tInfinity = y;
            }

        }
```

```
public void setCons()
	{
tAvg =(((tSurface+tInfinity)/2)-273);
pr = 0.73633353-(0.000266754*tAvg)-(2.98*Math.pow(10, -7)*Math.pow(tAvg,
2))+(4.45*Math.pow(10, -9)*Math.pow(tAvg, 3));
roRef = 1.2918381-(0.004686539*tAvg)+(1.52*Math.pow(10, -5)*Math.pow(tAvg, 2))-
(2.95*Math.pow(10, -8)*Math.pow(tAvg, 3));
muRef= (1.84*Math.pow(10, -5))+(1.12*Math.pow(10, -7)*tAvg)+(3.95*Math.pow(10, -
10)*Math.pow(tAvg, 2))-(1.13*Math.pow(10, -12)*Math.pow(tAvg, 3));
tk= 0.023633907+(7.56*Math.pow(10, -5)*tAvg)-(2.46*Math.pow(10, -8)*Math.pow(tAvg,
2));
	}
	public void initRes()
	{
		for(int i=0; i<3; i++)
		{
			rsdu[i] = 0;
		}
	}
	public double getRes(int i)
	{
		return rsdu[i];
	}

	public void grid()
	{
		double dx1, dy1;

		for(int i = imin - 1; i < imax; i++)
		{
			X1[i] = i * delx;
		}

		for(int j = 0; j < jmax; j++)
		{
			X2[j] = j * dely;
		}

		for (int i = imin; i < imax - 1; i++)
		{
			dx1 = 1 / (X1[i+1] - X1[i-1]);
			bn[i] = dx1 / (X1[i+1] - X1[i]);
			bs[i] = dx1 / (X1[i] - X1[i-1]);
		}

		for (int j = 1; j < jmax - 1; j++)
```

```
        {
                dy1 = 1 / (X2[j+1] - X2[j-1]);
                bw[j] = dy1 / (X2[j] - X2[j-1]);
           be[j] = dy1 / (X2[j+1] - X2[j]);
        }
}

public void init()
{
        for (int i = imin - 1; i < imax; i++)
        {
                for (int j = 0; j < jmax; j++)
                {
                        Psi[i][j] = 0.1;
                        Omega[i][j] = 0.1;
                        Temp[i][j] =tSurface;
                }
        }

        for (int j = 0; j <= jright; j++) //Upper and lower surface of the sequare
        {
                Psi[iup][j] = 0;
                Psi[ibottom][j] = 0;
                Temp[iup][j] =tSurface;
                Temp[ibottom][j] = tSurface ;
        }

        for (int i = ibottom + 1; i < iup; i++) //Right side of the sequare
        {
                Psi[i][jright] = 0;
                Temp[i][jright] = tSurface;
        }

        for(int i = imin - 1; i < imax; i++) //Right far field temp condition
        {
                Temp[i][jmax-1] = tInfinity;
        }
                for (int j = 0; j < jmax; j++) //Bottom far field temp condition
        {
                Temp[imin - 1][j] = tInfinity;
        }
        a1 = 0.1;
        a2 = 0.1;
        a3 = 0.1;
        a4 = 0.1;
}
```

```
//Source
	public void sorce(int i, int j, int k)
	{
		dv = (X1[i + 1] - X1[i - 1]) * (X2[j + 1] - X2[j - 1]);
		if (k == 1)
		{
			ddv = (X1[i + 1] - X1[i - 1]) * (Temp[i][j + 1] - Temp[i][j - 1]);
			source = Constants.GCOEF * roRef * Constants.EXCOEF * ddv / dv;
		}
		if (k == 2)
		{
			source = Omega[i][j];
		}
	}

//Convection
public void convec(int i, int j)
	{
		int myApp = 1;
		double g1pe, g1pw, g2pn, g2ps;

		dv = 2 * (X1[i + 1] - X1[i - 1]) * (X2[j + 1] - X2[j - 1]);

		g1pw = (Psi[i + 1][j] - Psi[i - 1][j] + Psi[i + 1][j - 1] - Psi[i - 1][j - 1]) / dv;
		g1pe = (Psi[i + 1][j] - Psi[i - 1][j] + Psi[i + 1][j + 1] - Psi[i - 1][j + 1]) / dv;
		g2ps = (Psi[i][j - 1] - Psi[i][j + 1] + Psi[i - 1][j - 1] - Psi[i - 1][j + 1]) / dv;
		g2pn = (Psi[i][j - 1] - Psi[i][j + 1] + Psi[i + 1][j - 1] - Psi[i + 1][j + 1]) / dv;

		ae = myApp * (Math.abs(g1pe) - g1pe);
		aw = myApp * (Math.abs(g1pw) + g1pw);
		ass = myApp * (Math.abs(g2ps) + g2ps);
		an = myApp * (Math.abs(g2pn) - g2pn);
	}

//Boundary conditions
	public void bound()
	{
		double z;
		for (int j = 0; j < jmax; j++)
		{
		Temp[imin - 1][j] = tInfinity;
		Psi[imin - 1][j] = (4 * Psi[imin][j] - Psi[imin + 1][j]) / 3;
		z = Omega[imin -1][j];
		Omega[imin - 1][j] = 0;//(4 * Omega[imin][j] - Omega[imin + 1][j]) / 3;
		Omega[imin-1][j] = z + Constants.RP * (Omega[imin-1][j] - z);
		Temp[imax - 1][j] = (4 * Temp[imax - 2][j] - Temp[imax - 3][j]) / 3;
```

```
                Psi[imax - 1][j] = (4 * Psi[imax - 2][j]- Psi[imax - 3][j]) / 3;
                z = Omega[imax-1][j];
                Omega[imax - 1][j] = (4 * Omega[imax - 2][j] - Omega[imax - 3][j]) / 3;
                Omega[imax-1][j] = z + Constants.RP * (Omega[imax-1][j] - z);
                }

                for (int i = imin; i < ibottom; i++) //left boundary flowin domain
                {
                Omega[i][0] = 0;// (4 * Omega[i][1] - Omega[i][2]) / 3;
                Psi[i][0] = 0;
                Temp[i][0] = (4 * Temp[i][1] - Temp[i][2]) / 3;
                }
                for (int i = iup + 1; i < imax - 1; i++) // Left  bounadry of  flowout domain
                {
                Omega[i][0] =0;//(4 * Omega[i][1] - Omega[i][2]) / 3;
                Psi[i][0] = 0;
                Temp[i][0] = (4 * Temp[i][1] - Temp[i][2]) / 3;
                }
                for(int i = imin; i < imax - 1; i++)  //Right boundary
                {
                z = Omega[i][jmax-1];
                Omega[i][jmax - 1] =0;//(4 * Omega[i][jmax - 2] - Omega[i][jmax - 3]) / 3;
                Omega[i][jmax-1] = z + Constants.RP * (Omega[i][jmax-1] - z);
                Psi[i][jmax - 1] = (4 * Psi[i][jmax - 2] - Psi[i][jmax - 3]) / 3;
                Temp[i][jmax - 1] = tInfinity;
                }
                for (int j = 0; j <= jright; j++) //for object
                {
         z = Omega[ibottom][j];
        Omega[ibottom][j] = -2 * (Psi[ibottom - 1][j]- Psi[ibottom][j]) / (roRef * dely * dely);
        Omega[ibottom][j] = z + Constants.RP * (Omega[ibottom][j] - z);
        z = Omega[iup][j];
        Omega[iup][j] = -2 * (Psi[iup + 1][j] - Psi[iup][j]) / (roRef * dely * dely);
        Omega[iup][j] = z + Constants.RP * (Omega[iup][j] - z);
                }

                for (int i = ibottom ; i <= iup; i++)
                {
        z = Omega[i][jright];
        Omega[i][jright] = -2 * (Psi[i][jright + 1] - Psi[i][jright]) / (roRef * dely * dely);
        Omega[i][jright] = z + Constants.RP * (Omega[i][jright] - z);
                }
        a1 = a1 + Constants.RP * ((-2 * Psi[ibottom - 1][jright] - Psi[ibottom][jright]) / (roRef
* dely * dely) - a1);
        a2 = a2 + Constants.RP * ((-2 * Psi[ibottom][jright + 1] - Psi[ibottom][jright]) / (roRef
* delx * delx) - a2);
```

```
        a3 = a3 + Constants.RP * ((-2 * Psi[iup][jright + 1] - Psi[iup][jright]) / (roRef * delx *
delx) - a3);
        a4 = a4 + Constants.RP * ((-2 * Psi[iup + 1][jright] - Psi[iup][jright]) / (roRef * dely *
dely) - a4);
        Omega[ibottom][jright] = a2;
        Omega[iup][jright] = a4;
        }

//Equation
public void eqn()
        {
                int jstart;
                double anum, adnm;
                double bbe, bbw, bbn, bbs;

                for (int i = imin; i < imax - 1; i++) //vorticity sub cycle
                {
                        if (i >= ibottom && i <= iup)
                        {
                                jstart = jright + 1;
                        }

                        else
                        {
                                jstart = 1;
                        }

                        for (int j = jstart; j < jmax - 1; j++)
                        {
                                sorce(i, j, 1);
                                convec(i, j);

                                bbe = 2 * be[j];
                                bbw = 2 * bw[j];
                                bbn = 2 * bn[i];
                                bbs = 2 * bs[i];
                                if((i == ibottom - 1) && (j == jright))
                                {
                                        Omega[i+1][j] = a1;
                                }


                           if ((i == ibottom) && (j == jright + 1))
                           {
                                Omega[i][j-1] = a2;
                           }
```

```
                    if ((i == iup) && (j == jright + 1))
                    {
                        Omega[i][j-1] = a3;
                    }

                    if ((i == iup + 1) && (j == jright))
                    {
                        Omega[i - 1][j] = a4;
                    }

anum = (ae + muRef * bbe) * Omega[i][j+1] + (aw + muRef * bbw) * Omega[i][j-1] +
(an + muRef * bbn) * Omega[i+1][j] + (ass + muRef * bbs) * Omega[i-1][j] + source;
adnm = ae + aw + ass + an + muRef * (bbe + bbw + bbs + bbn);

                        if (adnm != 0)
                        {
                            z = Omega[i][j];
                            Omega[i][j] = anum / adnm;
                        }

                        if (Omega[i][j]!= 0)
                        {
                            rs = 1 - z / Omega[i][j];
                            Omega[i][j] = z + Constants.RP * (Omega[i][j] - z);
                        }

                        if (Math.abs(rs) > rsdu[0]) //store max residual
                        {
                            rsdu[0] = rs;
                        }
                    }
                }

                for(int i = imin; i < imax-1; i++) //stream function sub cycle
                {
                    if ((i >= ibottom) && (i <= iup))
                    {
                        jstart = jright + 1;
                    }

                    else
                    {
                        jstart = 1;
                    }
```

```
		for(int j = jstart; j < jmax - 1; j++)
		{
			sorce(i, j, 2);

			bbe = (2 / roRef) * be[j];
			bbw = (2 / roRef) * bw[j];
			bbn = (2 / roRef) * bn[i];
			bbs = (2 / roRef) * bs[i];

			anum = bbe * Psi[i][j + 1] + bbw * Psi[i][j - 1]
				+ bbs * Psi[i - 1][j] + bbn * Psi[i + 1][j] + source;
			adnm = bbe + bbw + bbn + bbs;

			if (adnm != 0)
			{
				z = Psi[i][j];
				Psi[i][j] = anum / adnm;
			}

			if (Psi[i][j] != 0)
			{
				rs = 1 - z / Psi[i][j]; //store max residual
			}

			if (Math.abs(rs) > rsdu[1])
			{
				rsdu[1] = rs;
			}
		}
	}

	for (int i=imin; i < imax-1; i++)
	{
		if (i >= ibottom && i <= iup)
		{
			jstart = jright + 1;
		}

		else
		{
			jstart = 1;
		}

		for(int j = jstart; j < jmax - 1; j++)
		{
```

```
                                convec(i, j);

                                bbe = (2 * muRef / pr) * be[j];
                                bbw = (2 * muRef / pr) * bw[j];
                                bbn = (2 * muRef / pr) * bn[i];
                                bbs = (2 * muRef / pr) * bs[i];

anum = (ae + bbe) * Temp[i][j + 1] + (aw + bbw) * Temp[i][j - 1] + (an + bbn)
* Temp[i + 1][j] + (ass + bbs) * Temp[i - 1][j];
adnm = ae + aw + an + ass + bbe + bbw + bbn + bbs;

                                if (adnm != 0)
                                {
                                        z = Temp[i][j];
                                        Temp[i][j] = anum / adnm;
                                }

                                if (Temp[i][j] != 0 )
                                {
                                        rs = 1 - z / Temp[i][j];
                                }

                                if (Math.abs(rs) > rsdu[2])   //store max residual
                                {
                                        rsdu[2] = rs;
                                }
                        }
                }
        }

// Saving the result
        public void saveAs(int a)
        {
                String filePath;

                try
                {
                FileDialog fd = new FileDialog(new Frame(), "Save As", FileDialog.SAVE);

                   fd.setFile("*.csv");
                   fd.setDirectory(".\\");
                   fd.setLocation(50, 50);
                   fd.show();

                   filePath = fd.getDirectory() + fd.getFile();
```

```
FileWriter writer = new FileWriter(filePath);
int k = 0;
for(int i = imin - 1; i <= imax - 1; i++)
{
	for(int j = 0; j < jmax; j++)
	{
		if(a == 1)
		{
			writer.write(String.format("%.4f", Temp[i][j]));
		}

		if(a == 2)
		{
			writer.write(String.format("%.4f", Omega[i][j]));
		}

		if(a == 3)
		{
			writer.write(String.format("%g", Psi[i][j]));
		}

		if(a == 4)
		{
			writer.write(String.format("%.4f", u[i][j]));
		}

		if(a == 5)
		{
			writer.write(String.format("%.4f", v[i][j]));
		}

		if(a == 6)
		{
			if(i == ibottom)
			{
		if(j <= jright)
		writer.write(String.format("%g", hbottom[j]));
			}

			else if(i == iup)
			{
		if(j <= jright)
		writer.write(String.format("%g", hup[j]));
			}

		else if((i > ibottom) && (i < iup))
```

```
                                        {
                                    if(j == jright)
                                    writer.write(String.format("%g", hright[k]));
                                        }
                                    }

                                    if(j < jmax - 1)
                                    {
                                        writer.write(',');
                                    }
                                }

                                    writer.write('\n');
                                    k++;
                            }

                                writer.close();
                        }

                        catch (IOException e)
                        {
                                System.out.println("Error — " + e.toString());
                        }
                }

//Velocity components
public void vel()
        {
                int jstart;
                double conx = roRef * 2 * delx;
                double cony = roRef * 2 * dely;
                for(int i =imin ; i < imax-1; i++)
                {
                        if( (i >= ibottom) && (i <= iup) )
                        {
                                jstart = jright+1 ;
                        }

                        else
                        {
                                jstart = 1;
                        }
                        for(int j = jstart; j < jmax-1 ; j++)
                                {
                                u[i][j] = (Psi[i +1][j] - Psi[i - 1][j]) / cony;
                           v[i][j] = -(Psi[i][j + 1] - Psi[i][j -1]) / conx;
```

```
                        }
                    }
                //velocity components on the surface are zero
                for(int j = 0; j <= jright; j++)
                {
                    u[ibottom][j] = 0;
                   v[ibottom][j]= 0;
                   u[iup][j] = 0;
                   v[iup][j] = 0;
                }

                for(int i = ibottom; i <= iup; i++)
                {
                    u[i][jright] = 0;
                   v[i][jright] = 0;
                }
                    // velocity components on the boundaries should be calculated
  for (int j = 1 ; j<jmax - 1;j++)
  {
     u[0][j] = (-3 * Psi[0][j] + 4 * Psi[1][j] - Psi[2][j]) / cony;
     v[0][j] = -(Psi[0][j + 1] - Psi[0][j - 1]) / conx;
     u[imax-1][j] = (3 * Psi[imax-1][j] - 4 * Psi[imax - 2][j] + Psi[imax - 3][j]) / cony;
     v[imax-1][j] = -(Psi[imax-1][j + 1] - Psi[imax-1][j - 1]) / conx;
  }
     u[0][0] = (-3 * Psi[0][0] + 4 * Psi[1][0] - Psi[2][0]) / cony;
     v[0][0] = -(-3 * Psi[0][0] + 4 * Psi[0][1] - Psi[0][2]) / conx;
     u[imax-1][0] = (3 * Psi[imax-1][0] - 4 * Psi[imax - 2][0] + Psi[imax - 3][0]) / cony;
     v[imax-1][0] = -(-3 * Psi[imax-1][0] + 4 * Psi[imax-1] [1] - Psi[imax-1][2]) / conx;

     u[0][jmax-1] = (-3 * Psi[0][jmax-1] + 4 * Psi[1][jmax-1] - Psi[2][jmax-1]) / cony;
     v[0][jmax-1] = -(3 * Psi[0][jmax-1] - 4 * Psi[0][jmax - 2] + Psi[0][ jmax]) / conx;
     u[imax-1][jmax-1] = (3 * Psi[imax-1][jmax-1] - 4 * Psi[imax - 2][jmax-1] + Psi[imax -
3][jmax-1]) / cony;
     v[imax-1][jmax-1] = -(3 * Psi[imax-1][jmax-1] - 4 * Psi[imax-1][jmax - 2] + Psi[imax-
1][jmax - 3]) / conx;

//Left Boundary

for (int i = imin ;i< ibottom ;i++)
        {
        u[i][0] = (Psi[i + 1][0] - Psi[i - 1][0]) / cony;
  v[i][0] = -(-3 * Psi[i][0] + 4 * Psi[i][1] - Psi[i][2]) / conx;
        }
for (int i = iup + 1; i<imax - 1;i++)
        {
  u[i][0] = (Psi[i + 1][0] - Psi[i - 1][0]) / cony;
```

```
   v[i][0] = -(-3 * Psi[i][0] + 4 * Psi[i][1] - Psi[i][2]) / conx;
        }
//Right BoundPsiry
for(int i = imin; i<imax - 1;i++)
        {
u[i][jmax-1] = (Psi[i][jmax-1] - Psi[i - 1][jmax-1]) / cony;
   v[i][jmax-1] = -(3 * Psi[i][jmax-1] - 4 * Psi[i][jmax - 2] + Psi[i][jmax - 3]) / conx;
        }

        }
//Local heat transfer coefficient
        public void heat()
        {
                //for local heat transfer coefficients

                for(int i = ibottom; i <= iup; i++)
                {
        hright[i] = (-1 * tk* (4 * Temp[i][jright + 1] - 3 * Temp[i][jright] - Temp[i][jright +
        2]))/ (2* delx * (tSurface - tInfinity));
                }

                for(int j = 0; j <= jright; j++)
                {
        hup[j] = (-1 * tk * (4 * Temp[iup + 1][j] - 3 * Temp[iup][j] - Temp[iup + 2][j]))
                  / (2* dely * (tSurface - tInfinity));

        hbottom[j] = (-1 * tk * (4 * Temp[ibottom - 1][j] - 3 * Temp[ibottom][j] -
        Temp[ibottom 2][j])) / (2* dely * (tSurface - tInfinity));

                }
        }
}

//Printing the results

import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

public class Test extends JFrame
{
        Proc obj = new Proc();
        drawGrid drawGridObj = new drawGrid(obj);

        String sBar;
```

```
	JLabel statusBar;

	JTextField objHeight;
	JTextField objWidth;
	JTextField delX;
	JTextField delY;
	JTextField domainDiameter;
	JTextField iMin;

	JTextField tSurface;
	JTextField tInfinity;

	JTextField nItr;
	JTextField maxRes;

	public Test()
	{
		super();
		setDefaultCloseOperation(EXIT_ON_CLOSE);
		setLayout(new GridBagLayout());
		GridBagConstraints c = new GridBagConstraints();

		sBar = String.format("iMin: %d iMax: %d jMax: %d iUp: %d iBottom: %d
jRight: %d",
			obj.imin, obj.imax, obj.jmax, obj.iup + 1, obj.ibottom + 1, obj.jright +
1);

		JPanel gridCharacteristicsPane = new JPanel(new GridLayout(6, 2, 5, 5));
		gridCharacteristicsPane.setBorder(BorderFactory.createTitledBorder("Grid
Characteristics"));

		JPanel temperaturePane = new JPanel(new GridLayout(2, 1, 5, 5));
		temperaturePane.setBorder(BorderFactory.createTitledBorder("Temperature"));

		JPanel progressMonitorPane = new JPanel(new GridLayout(2, 1, 5, 5));
		progressMonitorPane.setBorder(BorderFactory.createTitledBorder("Progress
Monitor"));


		JMenu m1 = new JMenu("File");

		JMenu saveAs = new JMenu("Save Results");
		m1.add(saveAs);

		JMenuItem temperatureItem = new JMenuItem("Temperature");
		saveAs.add(temperatureItem);
```

```
temperatureItem.addActionListener(

        new ActionListener()
        {
                public void actionPerformed(ActionEvent event)
                {
                        obj.saveAs(1);
                }
        }
);

JMenuItem vorticityItem = new JMenuItem("Vorticity");
saveAs.add(vorticityItem);

vorticityItem.addActionListener(

        new ActionListener()
        {
                public void actionPerformed(ActionEvent event)
                {
                        obj.saveAs(2);
                }
        }
);

JMenuItem streamFunction = new JMenuItem("Stream Function");
saveAs.add(streamFunction);

streamFunction.addActionListener(

        new ActionListener()
        {
                public void actionPerformed(ActionEvent event)
                {
                        obj.saveAs(3);
                }
        }
);

JMenuItem xVelocity = new JMenuItem("X-Velocity");
saveAs.add(xVelocity);

xVelocity.addActionListener(

        new ActionListener()
```

```
        {
                public void actionPerformed(ActionEvent event)
                {
                        obj.saveAs(4);
                }
        }
);

JMenuItem yVelocity = new JMenuItem("Y-Velocity");
saveAs.add(yVelocity);

yVelocity.addActionListener(

        new ActionListener()
        {
                public void actionPerformed(ActionEvent event)
                {
                        obj.saveAs(5);
                }
        }
);

JMenuItem heatTC = new JMenuItem("Heat Transfer Coefficient");
saveAs.add(heatTC);

heatTC.addActionListener(

        new ActionListener()
        {
                public void actionPerformed(ActionEvent event)
                {
                        obj.saveAs(6);
                }
        }
);

m1.addSeparator();

JMenuItem exit = new JMenuItem("Exit");
m1.add(exit);
exit.addActionListener(

        new ActionListener()
        {
                public void actionPerformed(ActionEvent event)
                {
```

```
                        System.exit(0);
                }
        }
);

JMenuBar menubar = new JMenuBar();
menubar.add(m1);
setJMenuBar(menubar);

JLabel progressMonitorLabel = new JLabel("Progress Monitor");
JLabel objHeightLabel = new JLabel("Object Height");
JLabel objWidthLabel = new JLabel("Object Width");
JLabel delXLabel = new JLabel("Del-X");
JLabel delYLabel = new JLabel("Del-Y");
JLabel domainDiameterLabel = new JLabel("Domain Diameter");
JLabel iMinLabel = new JLabel("i-Min");

JLabel tSurfaceLabel = new JLabel("T-Surface");
JLabel tInfinityLabel = new JLabel("T-Infinity");

JLabel nItrLabel = new JLabel("Number of Iterations");
JLabel maxResLabel = new JLabel("Maximum Residual");

statusBar = new JLabel(sBar);

objHeight = new JTextField(5);
objWidth = new JTextField(5);
delX = new JTextField(5);
delY = new JTextField(5);
domainDiameter = new JTextField(5);
iMin = new JTextField(5);

tSurface = new JTextField(5);
tInfinity = new JTextField(5);

nItr = new JTextField(5);
nItr.setEditable(false);
maxRes = new JTextField(5);
maxRes.setEditable(false);

JButton changeCharacteristics = new JButton("Change Characteristics");
changeCharacteristics.addActionListener(

        new ActionListener()
        {
                public void actionPerformed(ActionEvent event)
```

```
                        {
obj.setParams(Integer.parseInt(objHeight.getText()), Integer.parseInt(objWidth.getText()),
Double.parseDouble(delX.getText()), Double.parseDouble(delY.getText()),
Integer.parseInt(domainDiameter.getText()), Integer.parseInt(iMin.getText()));

sBar = String.format("iMin: %d iMax: %d jMax: %d iUp: %d iBottom: %d jRight: %d",
obj.imin, obj.imax, obj.jmax, obj.iup + 1, obj.ibottom + 1, obj.jright + 1);

statusBar.setText(sBar);
                        }
                    }
                );

                JButton changeTemperature = new JButton("Change Temperature");
                changeTemperature.addActionListener(

                    new ActionListener()
                    {
                        public void actionPerformed(ActionEvent event)
                        {
                            obj.setTemperature(Integer.parseInt(tSurface.getText()),
Integer.parseInt(tInfinity.getText()));
                        }
                    }
                );

                JButton startCalculation = new JButton("Start Calculation");
                startCalculation.addActionListener(

                    new ActionListener()
                    {
                        public void actionPerformed(ActionEvent event)
                        {
                            calculate();
                        }
                    }
                );

                JButton redrawMesh = new JButton("Redraw Mesh");
                redrawMesh.addActionListener(

                    new ActionListener()
                    {
                        public void actionPerformed(ActionEvent event)
                        {
                            drawGridObj.setGridParams(obj);
```

```
                        drawGridObj.repaint();
                }
        }
);

gridCharacteristicsPane.add(objHeightLabel);
gridCharacteristicsPane.add(objHeight);
gridCharacteristicsPane.add(objWidthLabel);
gridCharacteristicsPane.add(objWidth);
gridCharacteristicsPane.add(delXLabel);
gridCharacteristicsPane.add(delX);
gridCharacteristicsPane.add(delYLabel);
gridCharacteristicsPane.add(delY);
gridCharacteristicsPane.add(domainDiameterLabel);
gridCharacteristicsPane.add(domainDiameter);
gridCharacteristicsPane.add(iMinLabel);
gridCharacteristicsPane.add(iMin);

c.gridx = 0;
c.gridy = 0;
c.fill = GridBagConstraints.HORIZONTAL;
add(gridCharacteristicsPane, c);

c.gridx = 0;
c.gridy = 1;
c.fill = GridBagConstraints.CENTER;
c.insets = new Insets(5,0,0,0);
add(changeCharacteristics, c);

temperaturePane.add(tSurfaceLabel);
temperaturePane.add(tSurface);
temperaturePane.add(tInfinityLabel);
temperaturePane.add(tInfinity);

c.gridx = 0;
c.gridy = 2;
c.fill = GridBagConstraints.HORIZONTAL;
add(temperaturePane, c);

c.gridx = 0;
c.gridy = 3;
c.fill = GridBagConstraints.CENTER;
add(changeTemperature, c);

progressMonitorPane.add(nItrLabel);
progressMonitorPane.add(nItr);
```

```
        progressMonitorPane.add(maxResLabel);
        progressMonitorPane.add(maxRes);

        c.gridx = 0;
        c.gridy = 4;
        c.fill = GridBagConstraints.HORIZONTAL;
        add(progressMonitorPane, c);

        c.gridx = 0;
        c.gridy = 5;
        c.fill = GridBagConstraints.CENTER;
        add(redrawMesh, c);

        c.gridx = 0;
        c.gridy = 6;
        c.fill = GridBagConstraints.CENTER;
        add(startCalculation, c);

        c.gridx = 0;
        c.gridy = 7;
        c.fill = GridBagConstraints.CENTER;
        c.anchor = GridBagConstraints.PAGE_END;
        add(statusBar, c);

        c.gridx = 2;
        c.gridy = 0;
        c.gridheight = 8;
        c.fill = GridBagConstraints.VERTICAL;
        add(drawGridObj, c);

        pack();
        setVisible(true);
}

public void calculate()
{
        int nitr = 0;
double res = 1;
String maxR;
final int nmax =200000;
obj.setCons();
obj.grid();
obj.init();

while((Math.abs(res) > Constants.CC) && (nitr < nmax))
{
```

```
                obj.initRes();
                obj.eqn();
                obj.bound();

                res = 0;

                for(int i=0; i<3; i++)
                {
                        if(Math.abs(res) < Math.abs(obj.getRes(i)))
                        {
                                res = obj.getRes(i);
                        }
                }

                nitr = nitr + 1;

                if( (nitr % 100) == 0)
                {
                        nItr.setText(String.valueOf(nitr));
                }

                maxR = String.format("%.4f", Math.abs(res));
                maxRes.setText(maxR);
        }

        nItr.setText(String.valueOf(nitr));

        obj.vel();
        obj.heat();
        }

   public static void main(String[] args)
   {
        Test t = new Test();
   }
}
```

```
// Drawing mesh
import javax.swing.*;
import java.awt.*;
import java.awt.geom.Line2D;
```

```
public class drawGrid extends JPanel
{
	int imax, jmax, iup, ibottom, jright;

	public drawGrid()
	{
		setBackground(Color.WHITE);
		setPreferredSize(new Dimension(340, 640));
	}

	public drawGrid(Proc obj)
	{
		setBackground(Color.WHITE);
		setPreferredSize(new Dimension(340, 640));

		imax = obj.imax;
		jmax = obj.jmax;
		iup = obj.iup;
		ibottom = obj.ibottom;
		jright = obj.jright;
	}

	public void setGridParams(Proc obj)
	{
		imax = obj.imax;
		jmax = obj.jmax;
		iup = obj.iup;
		ibottom = obj.ibottom;
		jright = obj.jright;
	}

	public void paintComponent(Graphics g)
	{
		super.paintComponent(g);

		int xScale = 600 / (imax - 1);
		int yScale = 300 / (jmax - 1);

		g.drawRect(20, 20, 300, 600);

		for(int i = 0; i < imax; i++)
		{
			g.drawLine(20, i*xScale + 20, 320, i*xScale + 20);
		}

		for(int j = 0; j < jmax; j++)
```

```
                {
                        g.drawLine(j*yScale + 20, 20, j*yScale + 20, 620);
                }

g.drawArc(-280, 20, 600, 600, 90, -180);

g.setColor(Color.RED);

g.fillRect(20, 18 + (ibottom * xScale), (jright * yScale) + 2, ( (iup - ibottom) * xScale ) + 2 );
        }
}
```