Forschungszentrum Karlsruhe
in der Helmholtz-Gemeinschaft

Wissenschaftliche Berichte
FZKA 6959

Buoyant Convection in the HCLL Blanket in a Strong, Uniform Magnetic Field

A. Kharicha, S. Molokov,
S. Aleksandrova, L. Bühler

Institut für Kern- und Energietechnik
Programm Kernfusion

Juni 2004

Forschungszentrum Karlsruhe

in der Helmholtz-Gemeinschaft

Wissenschaftliche Berichte

FZKA 6959

Buoyant convection in the HCLL blanket in a strong, uniform magnetic field

A. Kharicha[1], S. Molokov[1], S. Aleksandrova[1], L. Bühler

Institut für Kern- und Energietechnik
Programm Kernfusion

[1]Coventry University, School of Mathematical and Information Sciences,
Priory Street, Coventry CV1 5FB, United Kingdom

Forschungszentrum Karlsruhe GmbH, Karlsruhe
2004

Impressum der Print-Ausgabe:

Als Manuskript gedruckt
Für diesen Bericht behalten wir uns alle Rechte vor

Forschungszentrum Karlsruhe GmbH
Postfach 3640, 76021 Karlsruhe

Mitglied der Hermann von Helmholtz-Gemeinschaft
Deutscher Forschungszentren (HGF)

ISSN 0947-8620

urn:nbn:de:0005-069596

Buoyant convection in the HCLL blanket in a strong, uniform magnetic field

Abstract

Modelling of steady-state magneto-convective flows has been performed for typical geometries of currently investigated Helium-Cooled Lithium Lead (HCLL) blankets. For fusion relevant magnetic fields the flow has been studied for a purely diffusive heat transfer regime and for a convective regime with finite Peclet number. The effect of the direction of the gravity has been investigated by studying the two limiting cases, i.e. horizontal liquid-metal layers and vertical layers, related to blanket modules in the equatorial plane and near the top poloidal position, respectively.

Depending on the cavity's orientation, the flow is organised in a single convective circulation or in double recirculation cells located at the hot part of the cavity, where the volumetric power density has its maximum. The main component of vorticity of these recirculation cells is aligned with the magnetic field. For superimposed forced flows with Pe = 1.2, the buoyant convection is sufficiently strong to impose its flow pattern on the cross flow in parts of the cavity. A remarkable result of the present analysis is that the buoyancy induced flow may reach, or even overcome the magnitude of the imposed forced flow that is required for tritium removal. For that reason buoyant MHD flows in HCLL blanket modules is a phenomenon that has to be considered for reliable designs.

Konvektionsströmungen im HCLL Blanket in einem starken, konstanten Magnetfeld

Zusammenfassung

Für typische, aktuell betrachtete Geometrien eines heliumgekühlten Blei-Lithium-Blankets (Helium-Cooled Lead Lithium, HCLL) werden stationäre magnetohydrodynamische Konvektionsströmungen modelliert. Die Strömungsberechnung für fusionsrelevante Magnetfelder erfolgt zunächst unter der Annahme reiner Wärmeleitung und anschließend in einem konvektiven Regime für endliche Peclet-Zahlen. Anhand zweier Grenzfälle, der horizontalen und der vertikalen Anordnung der Blanket Module, wird der Einfluss der Orientierung der Blanketbox bezüglich der Schwerkraftrichtung untersucht. Diese Anordnungen entsprechen Positionen im Fusionsreaktor in der Äquatorialebene und nahe der höchsten poloidalen Position.

Abhängig von der Orientierung der Behältergeometrie bildet die Strömung einen einzelnen Konvektionswirbel oder ein System von zwei Zirkulationsgebieten, wobei das Wirbelzentrum in beiden Fällen in der Nähe der höchsten volumetrischen Beheizung auftritt. Die Hauptkomponente der Wirbelstärke ist entlang magnetischer Feldlinien ausgerichtet. Selbst bei der Überlagerung einer Zwangsströmung mit Peclet-Zahlen bis Pe=1.2 ist die Konvektionsströmung stark genug, um die Strömungsmuster zu dominieren. Ein wichtiges Ergebnis der Untersuchungen ist die Tatsache, dass die schwerkraftgetriebene Strömung Geschwindigkeiten erreichen kann, die eine vergleichbare Größenordnung haben wie die zur Tritiumextraktion erforderliche Zwangsströmung, oder diese sogar übersteigen. Für eine zuverlässige Auslegung von HCLL Blankets ist die Kenntnis magnetohydrodynamischer Konvektionsströmungen erforderlich.

Buoyant convection in the HCLL blanket in a strong, uniform magnetic field

Contents

1. INTRODUCTION

Lithium and lithium lead alloy Pb-17Li are attractive breeder materials foreseen in blankets of fusion power reactors because they allow breeding of sufficient amount of tritium. In the case of self-cooled blankets, their high thermal conductivity facilitates the transfer of the heat generated within the liquid metal and the walls of the cooling ducts to an external heat exchanger. In blankets Pb-17Li is submitted to very strong heating, giving rise to a buoyant flow. If these blankets are cooled by a separate coolant, the velocities of a forced flow required for tritium removal and purification can be kept at low values so that buoyant convection may become of similar importance as the forced flow.

When an electrically conducting liquid flows in presence of a magnetic field, electric currents are induced. The interaction of these electric currents with the imposed magnetic field gives rise to an electromagnetic force called the Lorentz force. Usually this force results in strong braking of the liquid motion, especially in the core of the flow. This braking is the cause of higher pressure drop compared with hydrodynamic flows. However, in some flow subregions such as the parallel layers located along the walls parallel to the magnetic field, the Lorentz force can act in the direction of the flow and therefore give rise to strong jets.

The present work deals with the most recent blanket concept, the so-called Helium Cooled Lead Lithium blanket (HCLL). This concept consists of almost rectangular, liquid metal filled boxes, the walls of which are cooled internally by a high-pressure helium flow. In the present design the walls in contact with the liquid metal are electrically conducting

There are also strong temperature gradients present in the blankets, which give rise to convective motion. The convective motion and the presence of the jets can be beneficial in HCLL because it helps to transport tritium from all of the blanket regions to the outlets located at the top of the blanket. At the same time, it can create zones of recirculation and reversed flows, which can trap tritium. Thus, the study of the convective motion in rectangular geometries is required in order to estimate the strength of the convective motion in fusion blankets. This investigation is performed here using CFX 4.6.

The report is organized in the following way. The next chapter presents the background and the general form of the governing equations relating to all subsequent work. This is followed by a brief literature review. Chapter 2 describes the numerical methods, and the subroutines created to adapt CFX 4.6 for the modelling of MHD problems. Chapter 3 deals briefly with the validation of the numerical code using previous results found in the literature. In the same chapter, the practical case, corresponding to the present design of the future blanket, is explored.

1.2 Background

This investigation deals with the effect of a steady magnetic field on laminar buoyancy-driven convection of an electrically conducting fluid in enclosed cavities. The walls in contact with the fluid are electrically conducting and impose though this fact their influence on the fluid motion.

Buoyant convection in a rectangular geometry has been studied by many authors. Analytical, numerical and experimental work has been carried out for geometries with different aspect ratios, various orientations of the magnetic field and the temperature gradient, purely diffusive heat transfer or with convective heat transfer present. Only papers directly relevant to the present problem will be discussed in the following.

It is well known that introduction of a uniform magnetic field has a damping effect on the velocity and can dramatically affect the flow. Analytical study by Garandet et al (1992) and numerical studies by Alchaar *et al.* (1995) and Ben Hadid *et al.* (1997) were made for two-dimensional horizontal cavity in a vertical magnetic field, with isothermal vertical end-walls and the assumption of a uniform horizontal temperature gradient. For high Hartmann number, *Ha*, the velocity profile is linear in the core with exponential decay to zero within viscous Hartmann layers along the walls perpendicular to the imposed magnetic field. These layers are of thickness $Ha^{-1}H$, where H is the distance between the horizontal walls. The damping effect of the magnetic field was found to scale core velocity as $Gr/Ha^2(\nu/H)$, where Gr is the Grashof number. Recirculation of the parallel flow occurs at the end-regions of thickness $Ha^{-1/2}H$, again less than the height of the cavity. These end-regions are referred to as `the parallel layers', a generic term which is used for layers parallel to the applied magnetic field. Bühler (1998) studied buoyancy-driven convection in a tall rectangular cavity with purely diffusive heat transfer. He considered the effect of the conductivity of the walls and the orientation of the magnetic field on the flow.

Ozoe and Okada (1989,1992) studied the effect of the magnetic field orientation on the damping of the convective flow both numerically and experimentally. They considered a cube with two differentially heated opposite sidewalls, all other walls being thermally insulated. The thermal gradients were sufficiently large for the convective heat transfer to be significant. The authors came to an important conclusion that the magnetic field had the greatest effect on damping the velocity when it was aligned with the temperature gradient and perpendicular to the gravity.

In the absence of a magnetic field, the Rayleigh number, Ra, is a measure of the importance of buoyant forcing compared to viscous damping and diffusive heat transfer. The effect of the magnetic field is to reduce the effective Rayleigh number. It has been shown by Cowley (1996) that for a strong magnetic field the non-dimensional parameter group Ra/Ha^2 plays an analogous role to that of the Rayleigh number in the absence of the magnetic field.

One of the most important properties of this kind of flow has been noted by Ma & Walker (1995) and Alboussière, Garandet & Moreau (1996). It has been shown in both studies that the magnitude of the convective flow may differ by orders of magnitude depending on whether the temperature is symmetric or anti-symmetric in the direction of the magnetic field. This is explored in more detail by Aleksandrova (2001) and Aleksandrova & Molokov (2000, 2002, 2004), who showed that the structure of the buoyant flow depends on many factors, such as the orientation of the field with respect to gravity, thermal conductivity of the walls, symmetry of the temperature with respect to the magnetic field co-ordinate, the heat fluxes in all three directions, etc.

For cavities with electrically conducting walls these ideas have been further developed by Molokov & Bühler (2003). They have pointed out that jets with velocities on the order O ($Ha^{1/2}$) may be present in the parallel layers. These jets are created by strong gradients of the electric potential normal to the layer. It is the jump of the electric potential between the wall and the core, which causes the high flow rate carried by the jets. Nevertheless, there exist some types of temperature distribution, for which the induced electric current is tangential to all the walls. This may exclude the presence of jets in parallel layers. Although the investigation has been performed for various values of the conductance ratio, the model used can applied only to the situations where the walls are better conducting than the parallel layers ($c \geq \mathrm{Ha}^{-1/2}$). Another numerical model is needed for smaller conductance ratios, as it may occur in the fusion blankets.

Concerning the internal heat flux, only a few computational and analytical studies exist. Bühler (1998) and Di Piazza (1999) have performed an analysis for fully three-dimensional configurations. These studies deal with cubic or vertical cavities with constant internal heat sources.

In the absence of magnetic field, if the imposed thermal gradient responsible for buoyancy-driven convection is increased sufficiently, then the flow will eventually become unstable. In other words, there is a critical Rayleigh number, Ra_{C}, above which the flow is unstable. There are different types of instabilities and the onset of these will also depend on the Prandtl number, Pr. It is of particular interest for the current study that the application of a steady magnetic field results generally in damping of the unsteady oscillations and increasing the critical Rayleigh number. It has been found in all cases that Ra_{c} increases with Ha. Stability analysis was considered to be outside the scope of this work and it will generally be assumed that the magnetic field is sufficiently strong so that the Rayleigh number is below the critical value, which is assumed to be the case for the fusion blankets.

A comprehensive study of the heat transfer characteristics of free and forced convective MHD flows at fusion-relevant conditions is carried out. Various analytical and numerical models describing MHD heat transfer phenomena in this parameter regime are discussed.

1.3 Governing equations

In the following we will be concerned with the buoyant convective flow, which sets in owing to differences in temperature T^* implying temperature dependent density ρ^* within the fluid. Assuming that the Boussinesq approximation is valid, the fluid density is expressed in terms of temperature as follows:

$$\rho^* = \rho_0^*(1-\beta^*(T^* - T_0^*)) . \tag{1.1}$$

Here the density of the fluid at temperature T_0^* is denoted by ρ_0^*, and the thermal expansion coefficient by β^*. T_0^* is a reference temperature at a certain, fixed point within the flow or at a wall.

It is assumed throughout this study that the magnetic Reynolds number $Re_m = \mu^* \sigma^* v_0^* a^*$ is sufficiently small so that the magnetic field is not perturbed by the presence of the flow. The electrical conductivity, σ^*, kinematic viscosity, ν^*, and the magnetic permeability μ^* are constant and independent of temperature. The variable a^* is a typical length scale measuresd along magnetic field lines and v_0^* a characteristic velocity.

Consider the flow of a viscous, electrically conducting, incompressible fluid subject to a strong magnetic field $\mathbf{B}_0^* = B_0^* \hat{\mathbf{y}}$. Then the steady, dimensional, inductionless equations governing the flow are the Navier-Stokes equations:

$$\rho^* \frac{\partial \mathbf{v}^*}{\partial t^*} + \rho(\mathbf{v}^* \cdot \nabla^*)\mathbf{v}^* = -\nabla^* p^* + \rho^* \nu^* \nabla^{*2} \mathbf{v}^* + \mathbf{F}^* , \tag{1.2}$$

the Ohm's law

$$\mathbf{j}^* = \sigma^* \left(-\nabla^* \phi^* + \mathbf{v}^* \times \mathbf{B}^* \right), \tag{1.3}$$

and conservation of mass and charge

$$\nabla^* \cdot \mathbf{v}^* = 0 , \tag{1.4}$$

$$\nabla^* \cdot \mathbf{j}^* = 0 . \tag{1.5}$$

The temperature is governed by the energy equation

$$\rho^* C_P^* (\frac{\partial T^*}{\partial t^*} + (\mathbf{v}^* \cdot \nabla^*) T^*) = k^* \nabla^{*2} T^* + Q^* . \qquad (1.6)$$

In the above, t^* is time, **v*** is the fluid velocity, p^* is the pressure, ϕ^* is the electric potential, $\mathbf{F}^*$ is the volumetric body force, $\mathbf{j}^*$ is the electric current density, k^* is the thermal conductivity and C^*_P is fluid heat capacity. The fluid may be heated both externally (or cooled at the walls) or by internal volumetric heat sources Q^*. The heat produced by the Joule's heating is neglected in this study. In MHD problems considered here, the body force $\mathbf{F}^*$ in the Navier-Stokes equation (1.2) includes the Lorentz force and gravity, i.e.

$$\mathbf{F}^* = \mathbf{j}^* \times \mathbf{B}^* + \rho^* \mathbf{g}^* , \qquad (1.7)$$

where $\mathbf{g}^*$ is the gravity vector. Using expression (1.3) for the electric current, the body force can be rewritten as

$$\mathbf{F}^* = -\sigma \nabla^* \phi^* \times \mathbf{B}^* + \sigma \mathbf{v}^* \times \mathbf{B}^* \times \mathbf{B}^* + \rho \mathbf{g}^* . \qquad (1.8)$$

Taking divergence of Eq. (1.3) and using Eq. (1.5), an equation governing the electric potential is obtained. It reads

$$\nabla^{*2} \phi^* = \nabla^* \cdot (\mathbf{v}^* \times \mathbf{B}^*) . \qquad (1.9)$$

Thus, a set of Eqs. (1.2- (1.9) for fluid velocity, electric potential and pressure has to be solved subject to appropriate boundary conditions. The Ohm's law (1.3) is used for calculation of the electric current.

1.4 Dimensionless equations

The characteristic values of the length, the fluid velocity, time, the electric potential and the pressure are a^*, $v_0{}^*$ = $\rho_0^* \beta^* g^* \Delta T^* / (\sigma^* B_0^{*2})$, $a^*/v_0{}^*$, $a^* v_0{}^* B_0{}^*$ and $a^* \sigma^* v_0 B_0^{*2}$, respectively. The characteristic temperature difference ΔT^* depends on the nature of the heating. It may be specified in terms of the external heat flux, temperature applied at the walls, or characteristic intensity of internal heat sources. The co-ordinate direction is chosen so that the magnetic field is aligned with the *y*-axis, and the gravity with the *z*-axis.

The non-dimensional equations governing the flow are

$$\frac{Gr}{Ha^4}\left[\frac{\partial \mathbf{v}}{\partial t}+(\mathbf{v}\cdot\nabla)\mathbf{v}\right]=-\nabla p+Ha^{-2}\nabla^2\mathbf{v}+\mathbf{j}\times\mathbf{B}+T\hat{\mathbf{z}}, \qquad (1.10)$$

$$\nabla^2\phi=\nabla\cdot(\mathbf{v}\times\mathbf{B}),\ \nabla\cdot\mathbf{v}=0, \qquad (1.11\text{-}1.12)$$

$$Pe(\frac{\partial T}{\partial t}+(\mathbf{v}\cdot\nabla)T)=\nabla^2 T+Q. \qquad (1.13)$$

The ratio of the electromagnetic to the viscous force is determined by the square of the Hartmann number, expressed as:

$$Ha=a^*B_0^*\sqrt{\sigma^*/\rho^*\nu}. \qquad (1.14)$$

The Grashof number

$$Gr=a^{*3}\beta^*g^*\Delta T^*/\nu^{*2}, \qquad (1.15)$$

characterises the importance of the buoyancy forces with respect to viscous forces.

The Peclet number,

$$Pe=\nu_0^*a^*\ \rho^*C_P^*/k^*, \qquad (1.16)$$

determines the ratio of convective and conductive heat fluxes. It can be expressed in terms of the previously defined groups using the Prandtl number $Pr=\rho^*\nu^*C_P^*/k^*$ as

$$Pe=Pr\ Gr\ /Ha^2.$$

The electric current can be calculated from the non-dimensional form of the Ohm's law (1.3) that reads

$$\mathbf{j}=-\nabla\phi+\mathbf{v}\times\mathbf{B}. \qquad (1.17)$$

1.5 Boundary conditions

Velocity boundary conditions

For low to moderate values of the Hartmann number ($Ha \leq 500$) the set of equations is solved in the whole fluid domain, including Hartmann layers. The boundaries of the domain are then the solid walls. The boundary condition for the fluid velocity at each wall is the no-slip condition

$$\mathbf{v} = 0. \tag{1.18}$$

When the Hartmann number is high ($Ha > 500$), the need for the resolution of thin Hartmann layers is avoided by integrating analytically all the equations across it, as has been used by Leboucher (1999). The method consists of suppressing the viscous braking at the Hartmann walls, i.e. the second derivative in the direction normal to the walls of the components of velocity parallel to the walls (*u*,*w*) is set to zero. The normal velocity *v* is also set to zero. The boundary condition is then set at the boundary between the core and the Hartmann layers as follows:

$$\frac{\partial^2 w}{\partial y^2} = \frac{\partial^2 u}{\partial y^2} = 0 \text{ and } v = 0\,. \tag{1.19-1.20}$$

At the walls parallel to the magnetic field, the condition (1.18) remains valid.

The electric boundary conditions

The closure of the electric current depends strongly on the electric properties of the walls (thickness and conductivity). Consequently, the Lorentz force and the velocity distribution are governed by these properties. Depending on the thickness, electrical conductivity, and magnetic permeability of the walls, various possible combinations of hydrodynamic and electromagnetic boundary conditions can be obtained. The electromagnetic boundary conditions depend on the conductivity ratio and the ratio of the wall thickness e^*_w to the fluid depth a^*. In the general case, these two ratios do not play the same role, but when the wall thickness is much smaller than the width of the cavity, the expression for the boundary conditions is simplified. In the present work, only the electric properties are taken into account and the focus is on thin walls. The thin wall hypothesis implies that the electric potential does not vary across the wall. This means that the electric current density inside the wall is tangential to the wall. In this case the important parameter is the conductance ratio of the wall

$$c_w = \frac{\sigma_w^* e_w^*}{\sigma^* \; a^*}$$

Thus, the wall thickness $e_w^*$and its conductivity σ_w^* play similar roles. This hypothesis leads to the following boundary condition for the electric current (Müller& Bühler 2001)

$$\mathbf{j}\cdot\hat{\mathbf{n}} = -\frac{\partial\phi}{\partial n} = c_W \nabla_W^2 \phi \text{ at the walls.} \tag{1.21}$$

Here $\hat{\mathbf{n}}$ is the inward-directed unit vector normal to the wall, and $\nabla_W^2\phi$ is the two-dimensional Laplace operator in the plane of the wall.

When the conductance ratio vanishes, which corresponds to electrically insulating walls, the boundary condition (1.21) becomes

$$\mathbf{j}\cdot\hat{\mathbf{n}} = -\frac{\partial\phi}{\partial n} = 0\,. \tag{1.22}$$

This means that no electric current enters the walls.

An infinite conductance ratio corresponds to a perfectly conducting wall. In this case the boundary condition is

$$\nabla_W^2\phi = 0 \rightarrow \phi = const\,\mathbf{.} \tag{1.23}$$

In the case where the resolution of the Hartmann layer is omitted, coherently, the dimensionless wall conductance ratio c_W at the Hartmann walls is replaced by $c_W+\delta_{H,}$, where $\delta_H = Ha^{-1}$ is the conductance of the Hartmann layer.

2. THE NUMERICAL MODEL

There are several commercial codes available for modelling ordinary hydrodynamic flows, such as CFX, FLUENT, FLOW-3D, etc. Although most of them do not model MHD flows automatically, they can be modified using user-defined subroutines to include the body forces, such as the Lorentz force, and couple the momentum equations with other MHD equations.

We have decided to base the MHD code on CFX, which is a very flexible finite-volume code widely used in the industrial applications of MHD. The user can introduce several scalar equations and even modify the accuracy of the numerical scheme (O. Widlund (2000), private communication).

In this chapter, we present the mechanisms of the MHD implementation and validation of the numerical schemes used. Details about the software and the subroutines used are available in Appendix 1 and 4 respectively. The command file options are detailed in Appendix 2.

2.1 MHD flow modelling with CFX

We will first discuss how the governing equations are implemented in the CFX. Since the magnetohydrodynamic features are not included in CFX by default, certain modifications are made in order to include MHD into the CFX model. This involves the introduction of the Lorentz force into the momentum equation, and solution of the electric potential equation using the additional scalar option available in CFX. A special treatment is given to the set of the electric boundary conditions for the potential. By default, CFX deals with dimensional quantities. In order to use the non-dimensional formulation, appropriate coefficients should be introduced as discussed below.

Implementation of the dimensionless equations in CFX

As has been mentioned above, the CFX code is built to work with dimensional equations. Thus, Eqs. (1.2) and (1.4) can be implemented directly by setting physical properties of the fluid in the command file. However, the electromagnetic force has to be introduced as a source term for the momentum equation. The treatment of the electromagnetic force is discussed below.

Equation (1.7) can be solved treating the electric potential as a user-defined scalar governed by the steady version of diffusion-convection equation (1.9) with the source term $\nabla^* \cdot (\mathbf{v}^* \times \mathbf{B}^*)$ calculated for each control volume.

However, non-dimensional formulation of the problem, as given by Eqs. (1.10)-(1.13), is preferable. In order to use non-dimensional quantities, appropriate non-dimensional parameters should be used rather than actual viscosity, density etc. It follows from Eqs. (1.10), (1.2) that Gr/Ha^4 stands for fluid density while Ha^{-2} corresponds to the fluid viscosity. For the heat equation it is sufficient to set $PeHa^4/Gr$ for the heat capacity, and 1 for the thermal conductivity. If the flow is considered being inertialless (Gr/Ha^4<<1), the fluid density is set to unity, and the heat capacity to *Pe*. The details of how to introduce the Lorentz and buoyant forces into the momentum equations and the source term into the electric potential equation, are considered below.

a) Electric potential equation and electric current

The electric potential has to satisfy Eq. (1.11). For the magnetic field $\mathbf{B} = \hat{\mathbf{y}}$, applied in the *y*-direction, the right-hand side of this equation can be written as follows:

$$\nabla \cdot (\mathbf{v} \times \mathbf{B}) = -\frac{\partial w}{\partial x} + \frac{\partial u}{\partial z}. \tag{2.1}$$

The straightforward approach to the problem is to approximate the source defined in the expression (2.1) at the centre of each cell and to multiply this value by the volume of the cell. Introducing such a source term would involve using utility subroutine GRADV for calculation of the gradients of velocity components at the cell centre and then multiplying it by the volume of the corresponding cell. In Appendix 5, such an approach is used in user subroutine USRSRC. (Note that the source term in CFX appears at the same side of the equation as the diffusion term, therefore -∇·(**v**×**B**) times cell volume is actually used).

For fully three-dimensional flows, non-zero three-dimensional currents defined by Eq. (1.17) are present in the fluid. For the magnetic field $\mathbf{B} = \hat{\mathbf{y}}$, applied in the y-direction, the three components of the electric current are

$$j_x = -\frac{\partial \phi}{\partial x} - w,\ j_y = -\frac{\partial \phi}{\partial y},\ j_z = -\frac{\partial \phi}{\partial z} + u\,. \tag{2.2}$$

Values of the electric current at cell centres can be calculated easily in CFX using actual values of the velocity stored in arrays u and w, and the utility subroutine GRADS for calculation of the gradient of the electric potential.

Implementation of the potential equation and Lorentz forces

b) Lorentz force.

The Lorentz force in a transverse magnetic field $\mathbf{B} = \hat{\mathbf{y}}$ has only two components:

$$F_x = -j_z\,,\ F_z = j_x \tag{2.3}$$

with the electric current components determined by Eq. (2.2). Again, the straightforward approach illustrated in Appendix 5 is to evaluate the electric current at the cell centres as described above and to introduce it into the momentum equations.

Implementation of the boundary conditions

There are several types of boundary conditions available in CFX. Here only the “wall” and “outlet” type have been used. Most of the common boundary conditions can be specified in the command files. In more complicated problems, user subroutines are used.

To specify a boundary where the appropriate boundary condition is applied, special subdomains must be created beforehand. These are called "patches" and can be either two-dimensional or three-dimensional. Thus, a boundary condition in CFX command file consists of the patch name where the condition holds, name of the variable, and the appropriate value or the flux of this variable at the patch. To implement more sophisticated boundary conditions, such as thin wall conditions, Fortran subroutine USRBCS must be used instead.

A subroutine is then necessary to implement the thin wall boundary condition.

Default conditions at solid walls are no-slip conditions for the velocity and zero flux conditions for temperature and user scalars. Alternatively, non-zero values or shear stress can be defined for the velocity, and general type of boundary conditions can be set for user scalars. Command >>WALL BOUNDARIES is used for this purpose.

In problems considered in this report, the following flow variables must be set at the wall boundaries: temperature and the user scalar standing for the electric potential. For $Ha > 500$ the velocities at the Hartmann walls should also be set. Before the details of the boundary conditions are discussed, some information about the grid organization should be given.

a) Grid organization

A grid node is placed at the centre of each cell. These nodes are called the internal grid nodes, and they are numbered from 1 to NCELL, where NCELL is the total number of cells. Additional grid nodes, called the boundary nodes are associated with the cell faces on each surface patch. For walls, these coincide with the centroid of the cell face. Thus, the total number of boundary nodes, NBDRY, is equal to the total number of surface patch cell faces. It should be noted here that there is no node in the corner, since the corner is not at the centre of a boundary face (fig. 1). This will have an effect on the way of how to write the boundary condition (1.21) just at the immediate vicinity of the edges.

The total number of grid nodes, NNODE is assumed to include the interior and boundary nodes, NNODE = NCELL + NBDRY. All variables U, V, W, P and T are dimensioned (NNODE, NPHASE), where NPHASE is the phase number. For example, W(INODE, 1) refers to the *w*-velocity at the node INODE. The additional scalars are stored in the array SCAL(NNODE, NPHASE, NSCAL), NSCAL being the number of scalars used. In the present investigation the electric potential is always the first scalar, referred to as SCAL(INODE, 1,1).

In the computational space, some transformed co-ordinate system is introduced for each block or patch. For convenience, the values of the co-ordinates are integer values at each grid node. These values are referred to as (I, J, K) and the corresponding directions to *I, J,* and *K.*

The *I*-direction is chosen so that it has the largest number of control volumes. That is, each block or patch can have an arbitrary orientation in terms of positive *I, J*, and *K* relative to other blocks in the grid.

The implementation of the boundary conditions needs the identification of the computational direction (*I, J, K*) with respect to the physical direction (*X, Y, Z*). A test has been performed on each patch to get the identification. It has been found that for all patches the direction (*I, J, K*) was simply (*X, Y, Z*).

Each patch is organised in a 3D Array (ILEN, JLEN, KLEN). For instance, (1, J, K) and (ILEN, J, K) give the co-ordinates of the first and the last node in the I-direction, respectively.

b) The implementation of the thin wall boundary conditions

User subroutine USRBCS is used to set variable boundary conditions at solid walls. There are three possible types of the boundary conditions for the velocities, the temperature and the scalars at the wall. These are constant value, constant flux, or a mixture of these two. In the last case the boundary condition must be written in the following form:

$$A_\varphi \phi + B_\varphi Q \quad = \quad C_\varphi \, , \tag{2.4}$$

where ϕ is the value of the scalar at the wall; Q is the scalar flux at the wall defined as $Q = -\partial\phi / \partial n$, and A_φ, B_φ and C_φ are constants or variables changing with time and (or) iterations. As the boundary condition for the temperature is simple (T = 0 or Q = 0), it will be set in the command file. Only the complex boundary conditions (1.19) and (1.21) need to be written in the form of (2.4).

a) The thin wall Boundary condition (1.21) can be written as follows:

$$Q = c\nabla^2 \phi \, . \tag{2.5}$$

The second derivatives of the potential at the wall must be written in the form of C_ϕ - $A_\phi\phi$. This can be done by the discretization of these derivatives on the grid.

Let us perform it for a surface parallel to the plane (x, y) (upper or bottom wall). For an equidistant grid, the central difference approximation to the second derivative in the *x*-direction is:

$$\frac{\partial^2\varphi}{\partial x^2} = \frac{\phi_{I+1} + \phi_{I-1} - 2\phi_I}{\Delta x^2} + O(\Delta x^2)\,, \tag{2.6}$$

which has a truncation error of the second order. After discretization of the second derivatives with respect to y, the formula (2.4) is fulfilled by taking the following set:

$$A_\phi = 2(\frac{1}{\Delta x^2} + \frac{1}{\Delta y^2})\,,\ B_\phi = \frac{1}{c} \text{ and } C_\phi = \frac{\phi_{I+1} + \phi_{I-1}}{\Delta x^2} + \frac{\phi_{J+1} + \phi_{J-1}}{\Delta y^2}\,. \tag{2.7}$$

However, the grid used in this report is not always equidistant. The central difference for the second derivative on a non-uniform orthogonal grid takes a more complicated form as follows:

$$\frac{\partial^2\phi}{\partial x^2} = 2(\frac{\phi_{I+1}/\Delta x_{+1} + \phi_{I-1}/\Delta x_{-1}}{\Delta x_{+1} + \Delta x_{-1}} - \frac{\phi_I}{\Delta x_{+1}\Delta x_{-1}}) + \frac{1}{3}(\Delta x_+ - \Delta x_-)\frac{\partial^3\phi}{\partial x^3} + O(\Delta x^2)\,. \tag{2.8}$$

The distance between the node *I*+*K* and the node *I*, is Δx_k. It is apparent that the use of the non-uniform grid leads here to the appearance of a first-order truncation error. One more point is then needed to get the second order scheme. The aim is to find three values for a_x, b_x, c_x such as:

$$\frac{\partial^2\phi}{\partial x^2} = 2(\frac{a_x\phi_{I+1} + b_x\phi_{I-1} + c_x\phi_{I-2} - (a_x + b_x + c_x)\phi_I}{a_x\Delta x_{+1}^2 + b_x\Delta x_{-1}^2 + c_x\Delta x_{-2}^2}) + O(\Delta x^2)\,. \tag{2.9}$$

These are found with the help of Taylor expansions written at each node, with the aim of eliminating the terms in Δx and Δx^3, which represents the first and the third derivatives of the potential. In the following a_x+b_x+c_x is chosen to be equal to one.

We find for $\Delta x_{+1} \neq \Delta x_{-1}$:

$$c_x = \frac{\Delta x_{+1}}{(\Delta x_{+1} - \Delta x_{-2}) \cdot (1 - \dfrac{\Delta x_{-2}(\Delta x_{+1} + \Delta x_{-2})}{\Delta x_{-1}(\Delta x_{+1} + \Delta x_{-1})})} \,, \tag{2.10}$$

$$b_x = -c_x \frac{\Delta x_{-2}(\Delta x_{+1}^2 - \Delta x_{-2}^2)}{\Delta x_{-1}(\Delta x_{+1}^2 - \Delta x_{-1}^2)} \,, \tag{2.11}$$

$$a_x = 1 - b_x - c_x \,. \tag{2.12}$$

For a uniform grid there is no need of four points to get a second order scheme. If $\Delta x_{+1} = \Delta x_{-1}$ then (2.11) has no meaning.

Another set a_y, b_y, c_y can be calculated for the second derivative in the y-direction. The boundary condition can then be written with the following coefficients A_ϕ, B_ϕ and C_ϕ:

$$B_\phi = 1/c \,, \tag{2.13}$$

$$C_\phi = 2(\frac{a_x\phi_{I+1} + b_x\phi_{I-1} + c_x\phi_{I-2}}{a_x\Delta x_{+1}^2 + b_x\Delta x_{-1}^2 + c_x\Delta x_{-2}^2} + \frac{a_y\phi_{J+1} + b_y\phi_{J-1} + c_y\phi_{J-2}}{a_y\Delta y_{+1}^2 + b_y\Delta y_{-1}^2 + c_y\Delta y_{-2}^2}) \,, \tag{2.14}$$

$$A_\phi = 2(\frac{1}{a_x\Delta x_{+1}^2 + b_x\Delta x_{-1}^2 + c_x\Delta x_{-2}^2} + \frac{1}{a_y\Delta y_{+1}^2 + b_y\Delta y_{-1}^2 + c_y\Delta y_{-2}^2}) \,. \tag{2.15}$$

This will be set at all the boundary nodes except the nodes just at the vicinity of the edges. This is due to the absence of nodes on the edges. For instance, the last nodes in the *I*-direction of co-ordinate I=ILEN have no neighbour nodes of co-ordinate I+1. To solve this problem, two methods have been used. In the first method the value at the corner ILEN+1 is interpolated by taking an average between the value of the potential at the last node ILEN+1 and the closest node (named INODEW) on the adjacent wall (see Fig. 1). Physically this corresponds to equalizing the current leaving the wall with the one entering into the adjacent wall

$$\frac{\phi(ILEN+1,J,K)-\phi(ILEN,J,K)}{\Delta x_{+1}} = -\frac{\phi(ILEN+1,J,K)-\phi(INODEW)}{\Delta z}, \quad (2.16)$$

where ϕ_w is the value of the potential at the closest node in the adjacent wall, Δz is the distance between the corner and INODEW. The co-ordinate of INODEW is found by using two successive functions provided by CFX. A first one is IPNODB which is used to get the co-ordinate of the cell adjacent to the boundary node, then the pointer IPNODN gives the neighbouring boundary node on the adjacent wall.

This procedure was found to be very stable, since it makes links between walls through the potential. Nevertheless, the convergence of the potential could hardly reach 10^{-5}. This is due to the fact that relation (2.16) uses first-order upward, difference while (2.9) is of second order.

This method was abandoned in favour of the second method, where the second order backward difference was used. This avoids the need of the corner node (*ILEN* + 1). In practice it consists in replacing the node (I+1) by (I-3), i.e. ϕ_{I+1} is replaced by ϕ_{I-2} in Eq. (2.9). The same procedure is applied to the low I-direction, since there are no (I-1) and (I-2) neighbouring nodes for the node number I = 1, and no (I-2) node for node number I = 2. The same procedure is applied at all the common boundaries between the walls.

This method provides a solution, which is 5-10% different from the averaged method, but the convergence was improved for the potential that could reach residual values below 10^{-13}. Moreover, by not using the very slow pointers IPNODB and IPNODN the calculation becomes three times faster than the previous method described by Eq. (2.16).

Despites the fact that (2.9) is very efficient, its calculation time was four times higher than when (2.8) was used. In addition the results were not significantly different (less than 1%). This is why the scheme defined by Eq. 2.8 has been selected but keeping (2.9) for the last nodes at the edges.

c) Velocity boundary conditions at the Hartmann walls (Ha **> 500***)*

For high Hartmann numbers (*Ha* > 500) the Hartmann layers are omitted, and the special boundary condition (1.19) is set at the Hartmann walls. All simulations in this report will assume that the magnetic field is aligned with the *y*-axis. The Hartmann walls are then located at a surface parallel to the (*x*, *z*) plane.

A boundary condition of the same type as the one used for the potential (2.4) can also be used for the velocities. A set of A_U, B_U, C_U should be determined for each of the tangential velocities (*u, w*), i.e. the second derivative in Eqs. (1.19) must be written as a function of the value and the first derivative (flux) of the velocity at the boundary node.

Let us consider the second derivative of the *u* velocity in the *y*-direction at a boundary node.

$$\frac{\partial^2 u}{\partial y^2} = \frac{u_{N+1} + u_{N-1} - 2u_N}{\Delta y^2} + O(\Delta y^2) = 0\,. \qquad (2.17)$$

The node number N is the last node in the *y*-direction, the node N-1 is the adjacent node inside the fluid, and the node N+1 is a fictitious node beyond the boundary (see Fig. 1). The node (N) is at an equal distance *Δy* from both node (N-1) and node (N+1). The CFX velocity “flux” at the boundary node is defined as follows:

$$Flux = \frac{\partial u}{\partial n} = \frac{u_{N+1} - u_{N-1}}{2\Delta y} + O(\Delta y^2)\,, \qquad (2.18)$$

where *n* is the outwards direction. Using (2.18) to eliminate u_{N+1} in (2.17), we then have

$$u_N - dyFlux = u_{N-1}\,. \qquad (2.19)$$

The second derivative (2.19) can be written using the set

$$A_U = 1,\ B_U = -\ dy \text{ and } C_U = u_{N-1}. \qquad (2.20)$$

3. RESULTS FOR THE HCLL BLANKET

Before carrying out the investigation, some comparison with known solutions is given in order to validate the MHD implementation.

3.2 Validation of the numerical model

Comparison with Di Piazza (1999)

The implementation of the CFX code for MHD flows developed here can be used to model various flows. Di Piazza *et al.* (2002) have investigated the problem of MHD free convection in a cubic enclosure. The geometry of the problem is shown in Fig. 2. The enclosure is volumetrically heated by a uniform power density and is cooled along two opposite vertical walls at $x = \pm 0.5$, all other walls being adiabatic. A uniform magnetic field is orthogonal to both the gravity vector and to the temperature gradient. The Prandtl number is *Pr*=0.0321 (characteristic of Pb-17Li at 573K).

We are going to perform comparison for $Ra = Gr^*Pr = 10^5$, c = 0.01, and for two values of the Hartmann number (*Ha* = 100*, Ha* = 500). Di Piazza et al. (2002) have chosen to omit the modelling of the Hartmann layers despite the presence of relatively low Hartmann numbers. We perform the calculations for exactly the same conditions that have been used by Di Piazza. Thus, this model is also a test for the special boundary condition of the velocity at the Hartmann wall.

Fig. 3 shows the vertical velocity profiles along the *x*-axis with some reported values given by Di Piazza (2002). The agreement is better for *Ha* = 100 than for *Ha* =500. In particular the maximum positive velocity at x = 0 is a bit shifted. The explanation comes probably by the model used by Di Piazza for the thin wall boundary condition. The model used is detailed in Di Piazza and Bühler (1999), where the authors use the "conducting wall" option provided by CFX4. It consists in surrounding the whole fluid domain by a conducting solid wall of finite thickness e_w^* and conductivity σ_w^*. To ensure the global conservation of charge, a zero 'flux' is set at the external faces of the solid walls. This method uses a thin "thick wall", which is different from the "thin wall" method that does not include the walls in the calculation domain. Thus the two methods give rise to slightly different closure for the electric current density, which create a different intensity of the Lorentz force in the core that damps the buoyancy forces. That can explain the small difference between the results.

The damping effect of the magnetic field can be observed as *Ha* increases, since velocities remain of the same order despite the fact that velocity scale is much lower for high Hartmann number (each plot has its own scale $u_0^* = (k^* / D^*) \cdot (Ra / Ha^2)$). Fig. 4 shows the dimensionless temperature profiles along the same axis, and here also the agreement is good. The influence of *Ha* on temperature is more evident in Fig. 5, where the isotherms in the midplane $y = 0$ are shown, they are very similar to those given by Di Piazza (2002) page 1501. A consequence of the damping of the convection is that for $Ha = 500$ the purely conductive distribution *T* is less modified by the flow than the case of $Ha = 100$.

Comparison with Molokov & Bühler (2003)

Another comparison has been made with the asymptotic model developed by Molokov and Bühler (2003). Consider the steady flow in a tall rectangular box in a strong, uniform magnetic field applied in the *y*-direction (Fig.6) The non-dimensional height of the box is 10, the (*x*, *y*) cross-section is a square with the side length 2. The walls at $y = \pm 1$ are heated and cooled, respectively, so that

$$T = \pm 1 \text{ at } y = \pm 1. \tag{3. 1}$$

Other walls are thermally insulating, and no volumetric heating is present.

When the Peclet number is sufficiently small, the solution of the energy equation (1.13) is

$$T = y. \tag{3.2}$$

Therefore, the hydrodynamic part of the problem described by Eqs. (1.10)-(1.12) is decoupled from the energy equation (1.13) and thus can be solved separately. This problem is first solved for a cavity with electrically insulating walls, then for a cavity with thin conducting walls.

A limiting case of a cavity with perfectly insulating walls ($c = 0$) can be modelled by CFX using the standard boundary condition

$$\frac{\partial \phi}{\partial n} = 0 \text{ at the walls} \tag{3.3}$$

instead of the condition developed in Sec. 2.4.3 (b). Another calculation has been carried out with a finite wall conductance ratio ($c = 0.1$). Both Calculations were performed for $Ha = 100$. False time stepping had to be used in order to ensure convergence.

The results agree well with the previous studies of similar flows. In the middle plane $z = 0$ the flow is fully developed. Such flow in an infinitely long duct has been considered by Bojarevics (1995), and current results are close to the Bojarevics solution. The results also agree with the three-dimensional numerical simulation by Ben Hadid & Henry (1997) for a horizontal cavity with aspect ratio 4:1:1 and temperature $T = x$. Their configuration can be reduced to the present case by a simple co-ordinate transformation.

Asymptotic analyses of flows for electrically insulating and thin conducting walls have been performed by Aleksandrova (2001), Molokov (2003) and Aleksandrova & Molokov (2004) respectively. Let us first consider the flow in the middle plane $z = 0$, where it is fully developed. The temperature difference between the walls at $y = \pm 1$ initiates convective motion in the positive (negative) z-direction near these walls, respectively. This flow interacts with the applied magnetic field, and therefore a non-zero electric current is induced in the $\pm\ x$ direction in the middle of the cavity. Depending on the conductivity of the walls, this current may close its loop either inside the boundary layers near the walls, or inside the Hartmann/parallel walls. Strong electric currents flowing inside the thin boundary layers may cause considerable flow redistribution. Thus, the electric current closure pattern will determine the flow pattern in the cavity, and it will depend on the conductivity of the walls.

When all the walls are *electrically insulating*, the current is bound to close its loop inside the boundary layers (Fig. 7). The thickness of the parallel layers is $O(Ha^{1/2})$ times higher than that of the Hartmann layers, therefore most of the current will close its loop inside the parallel layers near the walls $x = \pm 1$. With an $O(1)$ total electric current entering the parallel layers, current conservation requires the tangential current j_y near the parallel walls to be $O(Ha^{1/2})$. It follows from the y-component of the Ohm's law (1.17) that an $O(Ha^{1/2})$ jump in the electric potential is induced. Finally, the x-component of the Ohm's law (1.17) shows that an $O(Ha^{1/2})$ jump of the electric potential drives $O(Ha)$ w-velocity near the walls $x = \pm 1$ (Fig. 8). Thus, there is an $O(1)$ flow inside the core region, and ascending/descending high-velocity jets near the walls $y = \pm 1$, respectively (Fig. 8). The main part of the flow rate is confined to the thin parallel layers. High-velocity jets ascend along the heated wall $y = 1$, reach the top wall at $z = 5$, turn inside the parallel layer near the wall and descend along the opposite side wall $y = -1$ (Fig. 9).

When the walls have finite conductance ratio, the electric current closure pattern changes. With the conductance ratio of the walls equal to $c = 0.1$, part of the electric current may close its loop inside the walls (Fig. 10). Since the conductance ratio of the Hartmann walls is the same as that of the parallel walls, the electric current chooses the shortest possible way to close its loop. Therefore, the electric current approaching the parallel wall near the symmetry plane $y = 0$ either closes its loop inside the parallel layer, or shortcuts inside the parallel wall. On the other hand, electric currents approaching the parallel walls near $y = \pm 1$ shortcut through the Hartmann walls instead.

Since significant part of the electric current enters the parallel wall without actually turning inside the parallel layers, the tangential currents inside the parallel layers are much smaller than those in the insulating case. Consequently, the jump in the electric potential is smaller, and the resulting velocities are $O(Ha^{1/2})$ times lower than in the insulating case (Fig. 11). Comparison of the velocities for $c = 0$ and $c = 0.1$ is given in Fig. 12.

However, the asymptotic model is valid for high Hartmann numbers only and includes only the leading terms of the asymptotic expansion. This is the reason why it does not capture such features of the current closure pattern as the two-recirculation loops in the core region (Fig. 10) as well as the four small current loops between the core region and the Hartmann layers.

3.3 Channels in the HCLL blanket

The arrangement of the channels in the HCLL blanket is presented in Fig. 13. Current study focuses only on one channel, which has 208 mm in the toroidal direction, 700 mm in the radial direction and a gap of width of 43 mm. Since our aim is to study the intensity of the buoyancy effects, the channel is replaced by a cavity filled with Pb-17Li. An internal heat source is present in the entire cavity due to neutrons. This internal heat source decreases in intensity with the radial distance. The cooling plates are the upper and bottom walls with dimensions 208mm*700mm and effective conducting thickness of 2 mm. The others walls are 5 mm thick. The cooling plates are assumed to be isothermal and other walls are assumed to be thermally insulating. The magnetic field is toroidal, i.e. parallel to the cooling plates.

Concerning the gravity, the blanket modules have different orientations at different places. Some have inclination angles close to vertical and others are close to horizontal. Both situations should be investigated. In any case, the gravity is always perpendicular to the magnetic field.

Since the numerical model is dimensionless, the values of the characteristic parameters have to be specified first.

Dimensionless parameters

The physical properties of Pb-17Li relevant to this study are given in Table 1. The characteristic distance is the Hartmann distance a^* = 0.208m/2 = 0.104m, which is half the distance separating the Hartmann walls. The intensity of the magnetic field is in the range 5-10 T, and the characteristic temperature difference is taken as ΔT^* = 100 ^oC. This choice fixes the Grashof number to the value shown in Table 2.

The origin of the co-ordinate system is placed at the centre of the box. Then the cavity has the following dimensionless extensions: ± 3.365, ± 1.0, ± 0.207 in the x, *y*, and *z*-directions, respectively. Table 2 gives the values of dimensionless parameters that control the flow:

Table 1. Physical properties of Lead-Lithium at 450° C (Müller& Bühler 2001)

	Pb-17LI
Density $\rho^*_{0,}$ kg/m^3	9234
Kinematic viscosity ν^*, m^2/s	2.025×10^{-7}
Thermal expansion coefficient β^*, K^{-1}	1.817×10^{-4}
Thermal conductivity k*, W/mK	16.2
Thermal capacity C_p^*, J/Kg.K	189.78
Electric conductivity σ^*, A/(V.m)	0.7513×10^6

Parameter Gr/Ha^2 plays the role of the Reynolds number, *Re*, while Ha^4/Gr plays that of the interaction parameter, *N* (Albousièrre, Garandet & Moreau (1996), Bühler (1998)). The flow is inertialess in all the flow subregions if $Gr << Ha^{5/2}$, which is valid in the present configuration. Therefore, in the Navier-Stokes equations (1.10) the convective term is neglected. The Peclet number is not very small, so the convective heat transfer has to be considered. Thus, the temperature distribution is coupled to the fluid flow.

Taking the value for the steel equal to $\sigma^*_W = 1.05667\times10^6$ 1/Ωm, we get the conductance ratio $c = 0.026$ for the cooling plates and $c = 0.065$ for the others walls. Thus the wall conductance ratio of the cooling plates is the third of that of the other walls. This will have an effect on the distribution and the closure of the electric current lines.

Table 2. Dimensionless parameters at 450°C

	Ha	*Pr*	*Gr*	*Pe*	$Gr/Ha^{5/2}$
Pb-17Li	10000-20000	2.20×10^{-2}	4.889×10^9	0.27-1.07	$0.88\text{-}5\times10^{-1}$

Solution of the energy equation for Pe = 0.

a) Heat source

The dependence of the internal heat flux Q^* on the radial distance is given in Fig. 14. Before performing calculations, we need to fit the data points with an analytic expression. It presents some difficulty because of the lack of data for $x^* < 5.5$ cm. There are many types of curves that can fit the experimental data. These functions provide considerably different values for the internal heat source for $x^* < 5.5$ cm. As a result, the calculations can give various temperature distributions, and various flow patterns in the cavity. To overcome this problem we use the fact that the neutronic shelling is a flux, so it should decrease as the square of the radial distance $(x^*)^2$. However, this is not sufficient. The second condition can be found by using similarity with another set of data obtained for steel, the Eurofer, submitted to the same neutronic shelling. The evolution of the corresponding internal heat source Q^*_E is also given in Fig. 14. Both curves, for Pb-17 and for Eurofer, have a similar decreasing shape for x *>5.5. One can see that at $x^* = 0$ the Eurofer's curve has almost *a* zero derivative. We assume that the same property is satisfied in the Pb-17Li alloy.

There are still many functions that can satisfy these two conditions. A simple expression

$$Q^*(x^*) = \frac{Q_0^*}{1+(\frac{x^*}{x_0^*})^2} \tag{3.4}$$

has been chosen. In the above, Q_0^* is the maximum heat source and $x_0{}^*$ is the position where $Q^*(x_0) = Q_0{}^*/2$.

Fitting the experimental data with the curve described by Eq. (3.4) gives the following equations for Eurofer:

$$Q_E(x) = \frac{17.43}{1+(\frac{x^*}{6.961})^2}, \tag{3.5}$$

and for Pb-17Li:

$$Q_O(x) = \frac{20.23}{1+(\frac{x^*}{9.909})^2} \tag{3.6}$$

The corresponding dimensionless values are Q_0 = 165.96 and x_0^* = 0.994. If we place the origin of the co-ordinates at the centre of the cavity, the formula (3.6) becomes, in the non-dimensional form,

$$Q(x) = \frac{165.964}{1+(\frac{x+3.365}{0.994})^2}\,. \tag{3.7}$$

b) Temperature distribution

Once an equation for Q is determined, one can solve the diffusion heat equation (1.13). The boundary conditions are constant temperature $T = 0$ for the two cooling plates, and no flux at all the others walls. Fig. 15 shows the dimensionless temperature in the midplanes $y = 0$ and $z = 0$.

The temperature distribution is symmetric with respect to the plane $z = 0$, and does not vary along the magnetic field, i.e. along y. The shape of the temperature in the z-plane is almost the same as Q, and reaches the maximum dimensionless value of 3.436. This corresponds to a maximum temperature difference of 344°C between the Pb-17Li and the cooling plates.

To illustrate the importance of values of Q near x = -3.365, another calculation has been performed with a function which also fits the Pb-17Li data but does not obey the $1/x^2$ law. The function used is

$$Q = \frac{16.711-3.4603x}{1+0.2835*x}\,. \tag{3.8}$$

In Fig. 16, one can observe that the maximum temperature is 850 °C higher than the temperature at the cooling plates. This is more than the double of the one calculated with the right formula (3.3).

This comparison shows that it is absolutely necessary to know the values of the heat source at the vicinity of the plasma.

3.4 Results for a cavity with *Pe* = 0

The results for MHD flows depend strongly on the direction of the gravity with respect to the geometry of the box. Since the problem is linear, it is sufficient to investigate the two limiting cases, the horizontal and the vertical configuration, in order to have the full solution of the problem. For instance, the results for an inclined cavity is a superposition of the weighted results for a horizontal (0^{o}) and the vertical (90^{o}) cavity. In any case, the (poloidal) magnetic field is always perpendicular to the gravity vector. The investigation is performed for a range of Hartmann numbers $100 \leq Ha \leq 10\,000$, in order to observe the influence of the magnetic field strength on the flow. However, only the case of $Ha = 10000$ is of primary interest for the fusion blanket.

The characteristic velocity is found by balancing the buoyancy force with the electromagnetic force:

$$v_0^* = \rho_0^* \beta^* g^* \Delta T^* / \sigma^* B_0^{*2} = \frac{\nu^*}{a^*} \frac{Gr}{Ha^2}\,. \tag{3.9}$$

The characteristic velocity variation with the Hartmann number is

$$v_0^* = \frac{9514}{Ha^2}\ \text{m/s.} \tag{3.10}$$

For $Ha = 100$ the corresponding value is $v_0^* = 0.95\,\text{m/s}$, and for $Ha = 10000$ it is $v_0^* = 0.095\ \text{mm /s}$.

The calculation time for both the following configuration was about 45 hours for Ha=100, and 75 hours for Ha =10000. The calculations were stopped when the residuals for the potential equation reaches 10^{-6}.

Horizontal cavity

Fig. 17 shows the geometry of the situation. The gravity is aligned with the z-axis, and the magnetic field with the y-axis**.**

Fig. 18 shows the velocity vectors in the plane $y = 0$ for $Ha = 10000$. It can be seen that the flow is organised in a single primary recirculation cell with the flow moving upwards, i.e. towards the upper cooling plate, in the hot region, and downwards in the rest of the cavity. This recirculation cell known as the Hadley circulation, occupies only one third of the length of the cavity, leaving the rest of the fluid domain almost immobile. The presence of a wall at immediate proximity of the hot zone compels the fluid to form a upward jet (w velocity), and a very large zone of negative and small magnitude w velocity (Fig. 19).

Due to conservation of mass, the liquid has to flow horizontally towards the high x-direction in the upper part of the cavity, and in the opposite direction near the bottom wall (Fig. 20). The spanwise structure of the flow is more clear in Fig. 21, where one can see some three-dimensional effects for $Ha = 100$, which are not present for $Ha = 10000$. The structure of the flow is changed by the presence of a stronger magnetic field, so that it becomes increasingly uniform over the length of the cavity. The flow becomes quasi two-dimensional.

Due to the small height (compared to the depth) of the cavity, the maximum axial velocity $u_{\max}$ is higher than the maximum vertical velocity $w_{\max}$ (Fig.19 and Fig.20).

Another action of the magnetic field concerns its damping effect. Here the damping of the convective circulation can be easily understood in terms of the induced electric currents and potential. The magnetic body force is proportional to $\mathbf{j} \times \mathbf{y}$, where $\mathbf{j}$ is the induced electric current. This force acts in the direction against the velocities that are perpendicular to the magnetic field (u and w) and it is Ha^2 times larger than the viscous forces. The currents have two distinctive origins, $(\mathbf{v} \times \mathbf{B})$ due to the movement of the fluid through the magnetic flux lines, and $-\nabla\phi$, the electric field, which results from the reorganization and the conservation of the eclectic current over the domain.

In the present case, the distribution of the electric currents is largely influenced by the strong gradients of the electric potential induced on the sidewalls and on the cooling plates, where the velocity is very small. The positive (negative) axial velocity w induces an electric current $\mathbf{j}$ directed towards the negative (positive) x-direction. The current penetrates the sidewall and owing to the potential gradient, closes through the Hartmann walls (Fig. 22). This explains the “hole” in the shape of the potential distribution which is nothing else than a stagnation point of electric currents (Fig. 23a, Fig.24). In any other horizontal plane the electric current closure pattern is different due to the high value of the potential at the cooling plates (Fig. 23b). Indeed, in the central part of the cavity the induced electric current density is weak and almost vertical since u is the dominant component of the velocity there. This vertical current distorts the electric current lines towards the top or the bottom wall. In the walls, the current is mainly directed toward the high potential zone located at the cooling plates. In other words an important part of the current enter the Hartmann walls, and re-enter the fluid domain through the cooling plates or the parallel layer located at low x (Fig. 24). This closure of the electric current explains the presence of the loop located near the Hartmann walls. This is illustrated by the loops present at the both of the Hartmann walls (Fig. 22a), and by the electric current vectors plotted in the x = -1 plane (Fig. 25).

In the blankets the velocity of the Pb-17Li is expected to be around 1mm/s, which for Ha = 10000 corresponds to a dimensionless velocity of 10.53.

Despite the damping effect of the magnetic field the velocity caused by the buoyancy is still not negligible. The non-dimensional values $Vmax = \pm 16.81$ and $Umax = \pm 12{,}7$ (along the cavity) correspond to 1.35 mm/s and 1 mm/s. Thus the buoyant flow could oppose the main flow with the same strength, and can perturb locally the flow rate.

We consider the Pb-17 Li flow perturbed if the buoyant flow has velocity of 0.5 mm/s. Fig. 26 shows that perturbed zone is not negligible. For a non-zero Peclet number the iso-temperature lines could be strongly perturbed in this area. This will be considered in detail in Sec. 3.4

Vertical cavity

When the cavity is oriented vertically, the gravity is aligned with the *x*-axis, and the magnetic field with the *y*-axis (Fig. 27).

The cavity is now hotter near the bottom. Fig. 28 shows the velocity vectors in the plane $y = 0$. It demonstrates that the flow is symmetric with respect to the $z = 0$ plane, and is organized in two recirculation cells. The mean velocity is *u*, the fluid is flowing upwards in the hot region, and downwards along the vertical cooling plates (Fig. 29). Despite the fact that the cavity is oriented along the gravity **g**, the flowing domain is only located in the bottom third of the cavity (Fig. 28). The recirculation cells are well represented by the velocity streamlines in Fig. 30. In contrast with the horizontal cavity, the streamlines are as flat for $Ha = 100$ as they are for $Ha = 10000$. The main reason is that the non-MHD buoyancy-driven circulation cell is already essentially two-dimensional, with a vorticity aligned with the *y*-axis. Thus, the presence of a strong magnetic field does not modify this aspect of the shape of the flow.

The conservation of the flow rate compels the fluid to flow with high horizontal *w* velocity near the bottom wall making the *Wmax* to be the highest velocity. For high Hartmann numbers, it reaches the dimensionless value of 40, which corresponds to 3.86 mm/s. The vertical velocity, *u*, reaches the dimensionless value of 20, corresponding to 1.9mm/s. The *v* velocity is aligned with the magnetic field, and is essentially present in jets inside the layers parallel to the cooling plates (Fig. 31).

The *u* velocity induces strong electric currents aligned with the *z*-axis, directed towards the high *z* in the centre of the cavity. Near the cooling plates axial components of current are visible (Fig. 32). It explains why the maximum and minimum of the electric potential are located in the fluid (Fig. 33). The current closes through the parallel layers adjacent to the cooling plates (Figs. 32, 34, 35).

At the bottom cooling plate, the presence of the high *w* velocity induces a vertical electric current (aligned with x). Combined with the horizontal current (aligned with z), this results in particular current loops in the fluid, starting and ending at the same Hartmann wall (Fig. 35). Some currents close through the walls according to the strong potential gradient at the walls (Fig. 34). For instance, a (negative) current

J_z is present in the sidewalls in the opposite direction to the core current, giving rise to a loop at both sidewalls (Fig. 35).

Here also the strength of the velocity is compared with the expected 1mm/s for the fusion blankets. As mentioned previously, *Umax* and *Wmax* are much higher than those in the horizontal cavity. This is due to the fact that the gravity is oriented along x, i.e. the buoyancy force is acting along the mean direction of the cavity. Nevertheless, the size of the zone where the buoyant flow has a velocity higher than 0.5 mm/s is almost the same than for the horizontal cavity (Fig.36).

3.5 Finite Peclet number.

Closed horizontal and vertical Cavity

In the following the convective heat transfer is no longer neglected. We study the couple (Ha = 10000, Pe = 1). The analysis will be focused on the shift of temperature and velocity profiles with respect to the pure diffusive configuration (Pe = 0).

In Fig. 37 is represented the evolution of the temperature profile along the axis (y = 0). As could be expected the effect of the convection is stronger for the vertical cavity. While the temperature in the horizontal case, exhibits a very small difference with the purely diffusive case. The difference of can be easily explained by considering the results for Pe = 0. We can see that in the diffusive horizontal case the velocity is small (maximum) in the area where the temperature gradient is maximum (small). Let us illustrate this by taking the axis y = 0, where $\frac{\partial T}{\partial x}$ is maximum but as $u = 0$ there is no convection in the x direction. We also notice that at the same line y = 0, $\frac{\partial T}{\partial z} = 0$ while w is maximum. Therefore we have no convection in the z direction.

Doing the same investigation for the vertical cavity, we find that u is maximum where $\frac{\partial T}{\partial x}$ is maximum, and w is maximum where $\frac{\partial T}{\partial z}$ is maximum, leading to an important rule for the convection. In addition to that the magnitude of the velocities were found to be higher in the vertical configuration. These entire put together makes the vertical configuration much more interesting in term of convection. In the following we will focus only in the vertical case.

As in the vertical configuration the mean velocity is u, the temperature iso-lines are shifted towards high x (Fig. 38a). Another effect is to create a bow shape of the temperature distribution along y (Fig. 38b). This results from the fact that the u velocity being weaker near the Hartman walls, the heat is less driven at the walls than in the centre.

The position of the T maximum is no more close to the wall, but in the core and a little spread. This makes the maximum smaller than the pure diffusive case (Tmax=2.92), the same effect is present in the horizontal configuration but with a much smaller intensity (Tmax = 3.3).

Thus the distribution of the velocity should be very different from the diffusive one. Fig. 39 gives the evolution of u along the axis y = 0 for both Pe = 0 and Pe = 1. One can see that the spreading of the hot region in the core leads to a higher u velocity. The modification generated by the finite Peclet number on the flow, change only the magnitude of the potential but the main characteristics of the potential distribution remain the same(Fig.40 and Fig. 34).

Buoyant and forced convection in ducts

Now the presence of a cross flow is investigated. The boundary conditions at the low and high x walls are modified by using an inlet and an outlet, located at the entrance and at the exit, respectively. A constant dimensionless velocity u of 10.53 corresponding to the dimensional value 1 mm/s is set at the inlet. The flow entering the duct is supposed to have temperature 100^0C higher than that of the cooling plates, which corresponds to a dimensionless temperature T = 1.

In reality the flow at the entrance is different, but it is difficult to know precisely the exact velocity profiles at these ends. The presence of the magnetic field makes the successive channels completely coupled through the electric currents that can circulate from one channel to another. Thus, a better knowledge can be achieved only by carrying out the full modelling of the blanket, which is considered to be beyond the aim of this work. Nevertheless, the inlet and outlet boundary conditions are sufficiently good to study the perturbation generated by the buoyancy convection on both the cross flow and the temperature distribution.

Concerning the boundary conditions for the electric potential, the inlet allows only to set a constant value for the scalars, we have chosen to set the potential to a constant equal to 0.

Two configurations are studied here, a horizontal and a vertical duct. The aim of this section is to investigate the magnitude of the perturbation generated by the presence of the buoyancy. In previous section we have seen that the effect can be expected to be stronger in the case of finite Peclet number. Nevertheless the effect of the cross flow makes the problem more complicated.

The non-zero Peclet number makes the problem non-linear, therefore the results for the 45^0 inclined duct should be between the results for horizontal and vertical ducts but no more proportional to the exact average. In the following section we concentrate on the effect of the buoyant flow on the main cross flow (*u* velocity) and on the temperature distribution.

Horizontal duct.

Figures 41 and 42 show the temperature distribution in the midplanes y = 0 and along the *x*-axis. The isolines of the temperature are modified and shifted by the axial velocity *u*. The maximum of the temperature is located near the Hartmann walls, reaching the value 292^0C higher than that of the cooling plates.

Due to the small values of the *u* velocity near the walls, the removal of the internal heat produced there is less efficient near the walls than in the core. This explains why the position of the temperature maximum is near the Hartmann walls. Despite the presence of the convection of the heat by the flow, the liquid is almost totally cooled at the outlet. This means that the cooling plates are theoretically efficient in extracting the heat.

The *u* velocity at the entrance is constant, both the buoyancy and the Lorentz force act on this flow to modify it according to the equilibrium of the forces. It is known that without buoyancy the electromagnetic forces act quickly on this kind of flow to make a fully developed and symmetric “M”-shape. On can easily see the strong influence of buoyancy since the flow is not symmetric with respect to the y = 0 plane (Fig. 43). Despite the significant perturbation generated by the buoyancy, the *u* velocity profile recovers its M-shaped profile at quite short distance from the entrance (Fig. 43a).

The shift of the zone of high temperature creates an interesting and non expected effect on the shape of the u velocity. In fact we could expect that just after the entrance the maximum of *u* to be located in the half upper part of the cavity as in Fig. 18 or Fig. 20. We see in Fig.43a that the maximum of u is first located in the bottom part of the cavity before being located in upper part. Since the high temperature zone is not located at the immediate entrance, the flow (*w*) driven in the high z direction by the buoyancy compels the fluid coming from the inlet to come down to replace the flow going up at the hot region (Fig 43b). At the hot region the maximum of *u* becomes located in the upper part of the duct.

The effect of buoyancy can also be illustrated by taking the lines of iso-pressure(Fig. 44). The natural convection drives liquid towards the upper cooling plate. This creates a pressure gradient at the entrance directed towards the hot region (Fig 44). The lines of constant pressure are shifted towards the cooling plates by the *w* velocity. At the entrance the pressure gradient is stronger than the buoyancy, giving rise to a negative *w* near the inlet.

The distribution of the potential (Fig. 45) looks like a combination of the distribution created by buoyancy in a cavity (Fig. 24), and the well known distribution in a fully developed duct flow where the iso-surfaces of potential are aligned with the length of the cavity.

This coupled problem needed more calculation time to reach convergence than the previous configurations (75 hours). The calculation was stopped when the potential residual reached value of 10^{-4}. This can also be due to the poor knowledge we have of the entrance conditions.

Vertical duct

Since the buoyancy force is directed along the duct, we expect stronger effect of the buoyant convection on the duct flow and temperature. In Fig. 42 we can see that the temperature distribution is affected more than that of the horizontal duct. The maximum temperature is 270^0C, which is smaller than that for the horizontal duct (Fig. 46). Lower fluid temperatures are desirable with respect of corrosion issues and limiting temperatures at the walls. Here also the flow has been almost totally cooled when it reaches the outlet.

We can see in Fig. 47 and Fig. 48 that the velocity shape at the entrance of the duct resembles more the buoyant profile of Fig. 28 than the usual M-shaped profile associated with a fully developed flow. It needs also longer distance than in the horizontal duct to become an “M” shaped. The presence of negative velocity near the cooling plates is also due to buoyancy (Fig. 48). This means that the buoyancy can significantly perturb and modify the distribution of the flow in successive ducts such as in the future blankets (Fig 13). The magnitude of the velocity is high as expected, the maximum velocity is 57% higher than in the horizontal cavity (*Umax=45*). The potential distribution is here (Fig.49) also a combination between the one of Fig. 40 and one of the fully developed flow.

We have seen here that buoyancy can be expected to have a considerable effect on the cross flow. Nevertheless, the fact that we have to fix an entry temperature, fixed to unity here, lower than Tmax, can under estimate the buoyancy force. In real duct the flow entering the ducts is already heated, and its temperature could be of the order of Tmax.

It was difficult to reach convergence in this problem. Nevertheless after 75 hours we could reach residuals of 10^{-4} for the potential and velocity, and 10^{-3} for temperature. These convergence difficulties could be due to a difficulty to reach fully developed flow at the outlet. But it could also be a strong indication that the flow is transient and can develop instabilities. This possibility needs to be investigated in future.

4. CONCLUSIONS

Modelling of a steady buoyant flow in a strong magnetic field has been performed. The geometry considered corresponds to the present design for the ducts in the HCLL blanket, with thin electrically conducting walls. The flow is supposed to be inertialess and coupled to the heat transfer caused by the neutron flux. The purely diffusive heat transfer regime (Pe = 0) and the convective regime with Pe =1 have been investigated. The effect of the direction of the gravity has been investigated by studying the two limiting cases, i.e. the gravity aligned with the *z*- and *x*-axis.

For a weakly coupled flow (Pe = 0) in a closed cavity, the flow is organised in a single or double recirculation cells located at the hot part of the cavity. The vorticity of these cells is aligned with the magnetic field so that it is perpendicular to the length of the cavity. For flow with Pe = 1 the horizontal configuration does not change much the temperature distribution, and therefore only a week coupling is present between the flow and the temperature equations. The vertical configuration, however, gives rise to a strong thermal coupling, modifying importantly the magnitude of the flow.

Finaly the effect of the buoyancy on pressure driven duct flows has been investigated. The buoyant convection was found to be sufficiently strong to impose its flow pattern on the cross flow in the region of intense volumetric heating. In particular, for the vertical configuration, counter flows are present near the entrance of the duct. This phenomenon can affect the transport of tritium by the flow.

However, it is necessary to further investigate what happens near the entrance to the ducts. The MHD effects make the successive ducts fully coupled to each other by means of the electric current that can penetrate through the electrically conducting walls. Flow can choose to enter some ducts and to avoid other ducts, according to the distribution of the Lorentz force. Another very important question to be investigated in the future is the presence of instabilities in such kind of flows. Such instabilities may prevent trapping of tritium in closed recirculation loops, but can also significantly change the flow and temperature distribution pattern.

5. REFERENCES

CFX Material properties database, AEA Technology Engineering Software Ltd

T. Alboussiere, J. P. Garandet, P. Lehmann, and R. Moreau. Convective effects in the measurement of diffusivities and thermotransport coefficients. Liquid metal alloys and the use of a magnetic field. *Entropie,* 218:59-62, (1999).

T. Alboussiere, J. P. Garandet, and R. Moreau. Asymptotic analysis and symmetry in MHD convection, *Phys. Fluids* 8, 2215-2226, (1996)

S. Alchaar, P. Vasseur, and E. Bilgen. Natural convection heat transfer in a rectangular enclosure with transverse magnetic field. *Trans. of the ASME*, 117:668-673, (1995).

S. Aleksandrova. Buoyant convection in cavities in a strong magnetic field. *PhD thesis,* Coventry University (2001)

S. Aleksandrova and D. Molokov. Classification of buoyant convective flows in cylindrical cavities in strong, uniform, axial magnetic field, Proc 5th Intern. PAMIR Conf. on Fundamental and Applied MHD, Ramatuelle, France, Sept. 16-20, I:81-86 (2002)

S. Aleksandrova and D. Molokov. Three-dimensional buoyant convection in a rectangular cavity with differentially heated walls in a strong magnetic field, Accepted for publication in *Fluid Dynamics Research,* reference number TM16. (2004)

H. Ben Hadid, D. Henry. Numerical simulation of convective three-dimensional flows in a horizontal cylinder under the action of a constant magnetic field. Part 1. Two-dimensional flow. *Journal of Crystal Growth,* 166:436-445, (1996)

H. Ben Hadid, D. Henry, and S. Kaddeche. Numerical study of convection in the horizontal Bridgman configuration under the action of a constant magnetic field. *J. Fluid Mech*., 333:29-56, (1997).

H. Ben Hadid, D. Henry, and S. Kaddeche. Numerical study of convection in the horizontal Bridgman configuration under the action of a constant magnetic field. Part 2. Three-dimensional flow. *J. Fluid Mech*., 333:57-83, 1997

V. Bojarevics. "Buoyancy driven flow and its stability in a horizontal rectangular channel with an arbitrary oriented transversal magnetic field," *Magnetohydrodynamics* 31, 245-253 (1995).

L. Bühler. Laminar magnetohydrodynamic flow in vertical rectangular ducts. *Phys. Fluids.*, 10:223-236, (1998).

M.D. Cowley. Natural convection in rectangular enclosures of arbitrary orientation with vertical magnetic field. *Magnetohydrodynamics*, 32 (4):390-424, (1996)

L. Davoust, R. Moreau, M. D. Cowley, P. A. Tanguy, and F. Bertrand. Numerical and analytical modelling of the MHD buoyancy-driven flow in a Bridgman crystal growth configuration. *Journal of Crystal Growth*, 180:422-432, (1997).

J. P. Garandet, T. Alboussiere, and R. Moreau. Buoyancy-driven convection in a rectangular enclosure with a transverse magnetic field. *Int. J Heat Mass transfer,* 35 (4): 741-748 (1992)

B. G. Karasev, A. V. Tananaev. *Liquid metal fusion reactor systems, Plasma devices and operations* **1**, 11-30. (1990)

L. Leboucher. Monotone scheme and boundary conditions for finite volume simulation of magnetohydrodynamic internal flows at high Hartmann number, *J. Comput. Phys.* 150, 181-198. (1999)

N.Ma and J.S. Walker. Liquid-metal buoyant convection in a vertical cylinder with a strong vertical magnetic field and with a nonaxisymmetric temperature, *Phys. Fluids* 8, 2061-2071 (1995)

S.Molokov and L. Bühler, Three-dimensional buoyant convection in a rectangular box with Thin Conducting walls in a strong horizontal magnetic field. *Forschungszentrum Karlsruhe. Report FZKA 6817* . (April 2003)

K. Okada and H. Ozoe. Experimental heat transfer rates on natural convection of molten gallium suppressed under an external magnetic fielding either the X, Y, or Z direction. *Journal of Heat Transfer*, 114:107-114, (1992).

K. Okada and H. Ozoe. The effect of the direction of the external magnetic field on the three-dimensional natural convection in a cubic enclosure. *Int. J Heat Mass Transfer*, 32(10):1939-1954, (1989).

Di Piazza, L. Bühler. Numerical simulations of Buoyant Magnetohydrodynamic flows using the CFX code, *Forschungszentrum Karlsruhe Report FZKA 6354*. (1999)

I. Di Piazza, M. Ciofalo. MHD free convection in a liquid-metal filled cubic enclosure. II. Internal heating, *Int. J. Heat and Mass Transfer* 45, 1493-1511. (2002)

O. Widlund. Modelling of magnetohydrodynamic turbulence, *Technical Report ISRN KTH/MEK/TR--99/11--SE, TRITA-MEK.* (2000)

T. Tagawa, R. Moreau, G. Authié. Buoyant flow in long vertical enclosures in the presence of a strong horizontal magnetic field, *Proceeding of the 4th Pamir International conference, France,* 153-158 (2000)

6. APPENDIX

6.1 Appendix 1: The Software

The commercial fluid dynamics software package CFX 4.4 is used for solving MHD problems described below. CFX consists of the following modules:

- Pre-processing tools

  - CFX-Build 4.4

    An MSC-Patran based geometry and grid generator. It is used for creating the computational domain, specifying patches (i. e. parts of the domain where the boundary conditions will be defined – such as walls, inlets, outlets, symmetry planes, pressure boundaries etc.) and generating grid. Grid generation tools allow creation of non-uniform grids, which enables one to resolve boundary layers where needed.

  - CFX-Setup

    This module is used for creating the command file, containing information about fluid properties, physical models, boundary conditions and solver data.

  - Fortran routines

    Additional Fortran routines are used in order to implement the Magnetohydrodynamic flows and other extra features as described below.

- Solver tools

  - CFX-Solver

    The basis of the code is a conservative finite-volume method. All variables are defined at the centre of control volumes, and the equations are integrated over each control volume to obtain discrete equations. The complete set of equations is solved by iterative method. Pressure-correction algorithm is used to ensure mass conservation.

- Post-Processing tools

  - CFX-Analyse 4.4

  Graphic post-processor used for analysis and presentation of results obtained from the solver.

In addition to CFX, the following software tools have been used:

- Digital Fortran

  It is used for compilation of the user subroutines that are used by the CFX solver.

- Matlab and Sigmaplot 8.0

  Although CFX Analyse provides a wide range of graphic functions, sometimes others software are used for visualising solution.

Solution process

The process of problem solution in CFX consists roughly of the following stages:

- Pre-processing

  - Problem formulation.

  - Geometry creation.

    The geometry is created using CFX-Build. First, all blocks forming the computational domain are created. Then patches (two-dimensional or three-dimensional subdomains where the boundary conditions will be set) are specified. Finally, computational mesh is set up and the geometry file mXX.geo containing the information about mesh points, patches and block gluing information is created. Simple geometries can be created using the command file.

  - Command file.

The command file mXX.fc containing information about the model is created using CFX-Setup program or any text editor.

- Fortran subroutines.

  Additional Fortran subroutines mXX.f are required in order to implement MHD model. They are also used for specifying variable boundary conditions, setting convergence criteria etc.

- Solving

  All files created during the pre-processing stage (geometry file mXX.geo, command file mXX.fc and Fortran file mXX.f) are passed to the CFX-solver. It reserves the workspace, checks the validity of the information provided and runs the problem. During the solution process, some information about it can be displayed on dynamic graphs. Usually it is the residual values for each of the flow variables. It allows user to control the solution process and to see the progress of the solution.

  When the calculation is finished, CFX-Solver writes output data into the dump file mXX.dmp. Additional information about the solution process (e.g. summary of input data, residuals) and some statistics (e.g. drag on the walls, flow rates at all inlets and outlets etc.) are written to an output file mXX.fo. Any other information may be included into the dump/output file using command file options and user subroutines.

- Post-processing

  Data produced by CFX-Solver can be post-processed using CFX-Analyse or any other graphical tool. In the latter case, data should be written in required format. This can be achieved by using special options in the command file or by modifying user subroutines. CFX-Analyse is the built-in post-processing tool that allows user to create 2D plots, mesh plots, streamlines, contour plots, animations etc. It also has some basic calculation tools.

Hardware

All three-dimensional problems have been run on a Compaq Alpha Server DS20. It features 2 processors, 500 MHz each and 768 Mb RAM.

6.2 Appendix 2 : command file options

Command files are used in CFX for specifying fluid properties, physical models, boundary conditions and solver data. It is a text file with the following structure:

```
>>CFX4
      (Subcommands and Keywords)
>>MODEL TOPOLOGY
      (Subcommands and Keywords)
>>MODEL DATA
      (Subcommands and Keywords)
>>SOLVER DATA
      (Subcommands and Keywords)
>>CREATE GRID
      (Subcommands and Keywords)
>>MODEL BOUNDARY CONDITIONS
      (Subcommands and Keywords)
>>OUTPUT OPTIONS
      (Subcommands and Keywords)
>>STOP
```

Hereafter a brief review of options relevant to the problems considered in this report is given.

Command CFX4

This command specifies the major flow options, the list of the additional Fortran routines used and names of variables and phases. The following subcommands are available:

>> *Options*

The following options are used in this study:

TWO DIMENSIONS / THREE DIMENSIONS

Specifies whether two-dimensional or three-dimensional model is used. If two dimensions option is chosen, then the grid must have only one grid step in the *z*-direction.

USER SCALAR EQUATIONS

Additional scalars may be introduced in CFX. These satisfy a general diffusion/convection equation as discussed above. However, if the name of user scalar (defined as described below) starts with USRD, it is a dummy scalar and no equation is solved for it. Dummy scalars can be used for calculating additional variables, such as electric currents in three-dimensional flows.

RECTANGULAR GRID / BODY FITTED GRID

Only rectangular grid has been used in this study.

LAMINAR FLOW / TURBULENT FLOW / TURBULENT FLOW WITH WILCOX MODEL

Only laminar flows have been considered in this study.

ISOTHERMAL FLOW / HEAT TRANSFER

Heat transfer option is switched on if the temperature equation needs to be solved.

COMPRESSIBLE FLOW / INCOMPRESSIBLE FLOW

Only incompressible flows have been considered in this report.

TRANSIENT FLOW / STEADY STATE

The transient flow option is switched on when a time-dependent flow is considered. It can also be switched on for complex steady flows if convergence difficulties are experienced. Here the flow is considered steady.

>> User Fortran

Here the list of the CFX Fortran subroutines used in the problem is given.

>> Variable names

Here the names of the variables used in the problem can be changed. For example, standard names for user scalars are USER SCALAR N, where N is the number of the scalar. Using a special name starting with USRD turns off the solution of the transport equation for the scalar and allows one to use it for storing arbitrary information. An example of using the command is:

```
 U VELOCITY 'RADIAL VELOCITY'
 V VELOCITY 'POLOIDAL VELOCITY'
 W VELOCITY 'AZIMUTHAL VELOCITY'
 USER SCALAR1 'POTENTIAL'
 USER SCALAR2 'USRDCC_JX'
 USER SCALAR3 'USRDCC_JY'
 USER SCALAR3 'USRDCC_JZ'
```

Command Model data

This command is used to specify properties of the fluids, physical and numerical models etc.

PATCH NAME

Specifies the name of a three-dimensional patch where the body forces will act.

PHASE NAME / ALL PHASES

This option allows one to set the body forces acting on all phases or on a particular phase. For MHD flows, the Lorentz force will be acting on the liquid metal only, since the electrical conductivity of the air is much lower than that of the metal.

>>Differencing scheme

This option specifies which differencing scheme is chosen for the advection term.

ALL EQUATIONS / <*variable name*>

Specifies whether the differencing scheme applies to all equations or to a certain variable.

NO CONVECTION / UPWIND / HYBRID / HIGHER UPWIND / QUICK / CENTRAL / CONDIF / CCCT / MIN-MOD / VAN LEER / SUPERBEE / NO MATRIX

This keyword chooses the corresponding differencing scheme. The "no convection" option is used, for example, for the electric potential equation where no advection takes place, or in inertialess flows for the Navier-Stokes equations. It can be also used for the diffusive case of the energy equation.

>>***Physical properties***

>>*Buoyancy parameters*

This command allows one to introduce gravity force. We will consider only options relevant to heat transfer flow flows considered in this report, the buoyant force is set in the USRBF subroutine.

PHASE NAME / ALL PHASES

Shows whether the options apply to all phases or a particular phase only.

>>*Fluid parameters*

This subcommand is for specification of fluid properties.

PHASE NAME / ALL PHASES

Shows whether the options apply to all phases or a particular phase only.

VISCOSITY

Fluid viscosity is specified by this command.

DENSITY

Fluid density is specified by this command.

>>*Scalar parameters*

> >> *Diffusivities*
>
> Specifies diffusivity coefficients in additional scalar equations. For example, in the electric potential equation (8) the coefficient in front of the diffusive term is equal to unity.

>>**Set initial guess**

>>Set constant guess

<*variable name*>

Specifies variables (u, v, w, pressure, volume fraction, enthalpy, user scalar n) and their values used as the initial guess.

PATCH NAME

Name of the two- or three-dimensional patches where the source term appears. In case of a two-dimensional patch, the source term is given per unit area of the corresponding cell.

>>Title

Here a title can be given to the problem that will appear in the output file.

Command Solver data

>> False timesteps

This command allows improving problem convergence by using the false time steps method. It can be used in transient problems together with real time stepping. It can also be combined with underrelaxation method.

ALL EQUATIONS / <*variable name*>

Specifies which equations should be integrated using false time steps and the corresponding values of the false time steps.

>> Equation solvers

The set of linearised difference equations for a particular variable, one equation for each control volume in the flow, is passed to a simultaneous linear equation solver, which uses an iterative solution method. The alternative methods, available for this purpose, are specified within this command.

>> **Pressure correction**

SIMPLE / SIMPLEC / PISO / NON ITERATIVE PISO

By default, SIMPLE algorithm (Semi-Implicit Method for Pressure-Linked Equations) is used for updating pressure and correcting velocity components in order to ensure mass conservation. SIMPLEC is a modification of SIMPLE, which differs in its derivation of a simplified momentum equation. In the so-called PISO algorithm, a second pressure-correction equation is solved in order to improve the solution of the momentum equations while maintaining continuity. SIMPLEC method has been used in this study.

>> **Program control**

MINIMUM NUMBER OF ITERATIONS

Specifies minimum number of iterations performed by solver before the convergence testing is done.

MAXIMUM NUMBER OF ITERATIONS

Specifies maximum number of iterations performed by solver.

MASS SOURCE TOLERANCE

Specifies the convergence criterion in terms of mass residual.

>> **Reduction factors**

On each "outer" iteration step, the set of linearised difference equations for a particular variable is solved using an iterative solution method. An exact solution is not required because this is just one step in the non-linear "outer" iteration. The computational effort in obtaining a reasonable solution to the set of equations is controlled using command >> Reduction factors and >>Sweeps information.

The residual in a particular cell is the amount by which the linear equation there is not satisfied. Residual reduction factor is the amount by which the residual should reduce compared to its initial value.

<variable name> <factor>

Specifies a variable and a corresponding reduction factor. Here the reduction factor for pressure and for velocities depends on the Hartmann number. For the temperature it depends if either the convection plays or not an effective role. If Peclet number is null we set a higher value than for a convective heat transfer.

6.3 Appendix 3: A sample of a command file

This command file correspond to the configuration with cross flow, Ha = 10000 and Pe = 1.

```
>>CFX4
  >>OPTIONS
    THREE DIMENSIONS
    RECTANGULAR GRID
    CARTESIAN COORDINATES
    LAMINAR FLOW
    HEAT TRANSFER
    BUOYANT FLOW
    STEADY STATE
    USER SCALAR EQUATIONS 4
  >>USER FORTRAN
    USRBCS
    USRBF
    USRSRC
    USRTRN
  >>VARIABLE NAMES
    U VELOCITY 'U VELOCITY'
    V VELOCITY 'V VELOCITY'
    W VELOCITY 'W VELOCITY'
    PRESSURE 'PRESSURE'
    DENSITY 'DENSITY'
    VISCOSITY 'VISCOSITY'
    TEMPERATURE 'TEMPERATURE'
    ENTHALPY 'ENTHALPY'
    USER SCALAR1 'POTENTIAL'
    USER SCALAR2 'USRDCC JX'
    USER SCALAR3 'USRDCC JY'
    USER SCALAR4 'USRDCC JZ'

>>MODEL DATA
  >>AMBIENT VARIABLES
    TEMPERATURE 0.0000E+00
  >>DIFFERENCING SCHEME
    U VELOCITY 'NO CONVECTION'
    V VELOCITY 'NO CONVECTION'
    W VELOCITY 'NO CONVECTION'
    ENTHALPY 'QUICK'
    POTENTIAL 'NO CONVECTION'
  >>TITLE
    PROBLEM TITLE 'BOITE MHD'
  >>PHYSICAL PROPERTIES
    >>STANDARD FLUID
      FLUID 'AIR'
      STANDARD FLUID REFERENCE TEMPERATURE 1.0000E+00
    >>BUOYANCY PARAMETERS
      THERMAL EXPANSION COEFFICIENT 1.0000E+00
      GRAVITY VECTOR -1.000000E+00 0.000000E+00 1.000000E+00
      BUOYANCY REFERENCE TEMPERATURE 0.0000E+00
    >>FLUID PARAMETERS
      VISCOSITY 1.0000E-08
      DENSITY 1.0000E+00
    >>HEAT TRANSFER PARAMETERS
      THERMAL CONDUCTIVITY 1.0000E+00
      FLUID SPECIFIC HEAT 1.000000E+00
      ENTHALPY REFERENCE TEMPERATURE 0.0000E+00
    >>SCALAR PARAMETERS
      >>DIFFUSIVITIES
        POTENTIAL 1.0000E+00
>>SOLVER DATA
  >>PROGRAM CONTROL
    MAXIMUM NUMBER OF ITERATIONS 60000
    MINIMUM NUMBER OF ITERATIONS 1
    MASS SOURCE TOLERANCE 1.0000E-16
  >>FALSE TIMESTEPS
    ALL EQUATIONS 1.0000E-01
  >>UNDER RELAXATION FACTORS
    U VELOCITY 1.0000E-01
    V VELOCITY 1.0000E-01
    W VELOCITY 1.0000E-01
    PRESSURE 1.0000E-01
    TEMPERATURE 9.9990E-01
    POTENTIAL 9.900E-01
>>MODEL BOUNDARY CONDITIONS
  >>INLET BOUNDARIES
    PATCH NAME 'INLETA'
    U VELOCITY 1.0530E+01
    TEMPERATURE 1.000E+00
    POTENTIAL 0.0000E+00
  >>WALL BOUNDARIES
    PATCH NAME 'WALLZB'
    TEMPERATURE 0.0000E+00
  >>WALL BOUNDARIES
    PATCH NAME 'WALLZA'
    TEMPERATURE 0.0000E+00
>>STOP
```

6.4 Appendix 4: Additional subroutines used

Additional Fortran subroutines are used when an option cannot be specified via the command file (for example, non-constant boundary conditions). All user subroutines used in a problem are written into a file mXX.f (for example, m01.f).

Flag IUSED should be set to 1 in each subroutine. Every subroutine should also be listed in the command file under command >>USER FORTRAN. Hereafter only features relevant to this study are listed. User has access to the following variables:

- U, V, W, P, VFRAC, T, SCAL

Contain values of velocity components, pressure, volume fraction, temperature and user scalars. All these arrays have size (NNODE, NPHASE), where NNODE is the number of all nodes, including boundary nodes and dummy nodes which surround the calculation domain; NPHASE is the number of phases. The size of the scalar array is (NNODE, NPHASE, NSCAL), where NSCAL is number of user scalars.

- XP, YP, ZP, VOL, AREA, WFACT

Contain geometry information: co-ordinated of nodes, volumes of cells, areas of cell faces, weight factors.

Other data is available such as topological information etc. There are also work arrays WORK, IWORK and CWORK that allow user to reserve workspace for additional data and to pass data between subroutines.

Every cell in the computational domain can be referred to by either naming the block on which the cell resides and giving local co-ordinates (I, J, K) of the cell, or by using its internal, 1D, address, which lies between 1 and NNODE (NNODE is the number of all internal grid nodes - NCELL, including dummy cells around each block, plus the number of boundary nodes placed in the centre of each patch - NBDRY).

Utility routines are available from all user subroutines. They perform such tasks as finding addresses of patches, finding variable numbers etc. The most commonly used utility routines are described below.

All user subroutines contain clearly specified user areas where the modifications can be done.

The following subroutines have been used in this study: USRBCS, USRBF, USRCVG, USRINT, USRSRC, USRTRN. A brief description of each subroutine is given below.

Utility routines

GETSCA

This subroutine is used to find the number of a user scalar within the SCAL array. The alias name (CHARACTER*24) of the scalar is passed to the subroutine, and the number of the corresponding scalar is returned.

GETVAR

In most cases, the name of the variable is passed to subroutines (the CHARACTER*6 name such as 'W ', 'VFRAC ' etc.). However, sometimes the number of variable is required. Subroutine GETVAR returns the number of variable if its name is given.

GRADS

GRADS may be used to compute the physical space gradients of a scalar variable. It creates a temporary array GRAD(NCELL,3) where values of three components of the gradient are stored for all internal cells of the computational domain.

GRADV

GRADV may be used to compute the physical space gradients of velocity. It creates three temporary arrays UGRAD, VGRAD and WGRAD of size (NCELL,3) where values of three components of the gradient for each velocity component are stored for all internal cells of the computational domain.

IPALL

This subroutine allows user to get addresses of several blocks or patches. User specifies the name of the block or patch, whether it is a block or a patch and whether the addresses of cell centres or vertices are required. In addition, the patch type has to be specified. The patch type and/or the patch or block name may be specified as ‘*’ which means all the patch types and/or all the names. An array IPT of size NPT is returned, which contains 1D addresses of patch (block) cells.

IPREC

IPREC is used to return 1D addresses for all the centres or vertices in a rectangular group such as a block or patch. In contrast with IPALL, only one and only rectangular block or patch can be processed. The subroutine returns dimensions ILEN, JLEN and KLEN of the block/patch, and an array of corresponding 1D addresses.

Subroutine USRBCS

This subroutine enables user to specify more complicated boundary conditions. Simple boundary conditions can be set using the >>MODEL BOUNDARY CONDITIONS command.

User must set the flag IUBCSF to state whether the boundary conditions are to vary with iteration, time, or time and iteration (IUBCSF = 1, 2 or 3 respectively). In our case IUBCF = 2. Array VARBCS of size (NVAR, NPHASE, NCELL+1:NNODE) can be changed in order to set values of flow variables (NVAR is the total number of variables). In the Appendix 5 the Subroutine USRBCS is used to set the thin wall boundary condition, with a special treatment near at the edges.

Subroutine USRBF

This subroutine allows user to add body force to the momentum equations. Simple body forces of this type can be included using the >>BODY FORCES command. In this study subroutine USRBF is used for including the Lorentz and the buoyant forces into momentum equations.

In order to enable the code to linearize the body force source term correctly, **F** is expressed in the following form in CFX:

$$\mathbf{F} = \mathbf{C} - R\mathbf{v} \, .$$

In the above **C** is a vector, and *R* is a diagonal matrix; **v** is the fluid velocity. Arrays BX, BY and BZ of size NCELL are filled with the three components of the vector **C** for each internal cell of the computational domain. Similarly, arrays BPX, BPY and BPZ of the same size contain factors *R*.

In the Appendix 5, the subroutine USRBF is used to include the two-dimensional body force $F_X = -(U + \frac{d\phi}{dz})$ and $F_Z = -(W + \frac{d\phi}{dz}) + T$ into the momentum equation.

Subroutine USRSRC

This subroutine allows user to intervene and change the equations; in particular to add source terms into convection-diffusion equations. In this study subroutine USRSRC is used for specifying the source term in the electric potential equation. It can also be used to include body forces into the momentum equations, instead of USRBF. In contrast to USRBF, the source term in USRSRC must be integrated over the control volume (in simple cases, the original source per volume must be multiplied by the volume of the cell).

In Appendix 5 the source term $S_P = \nabla\cdot(\mathbf{v}\times\mathbf{B}) = \frac{\partial U}{\partial z} - \frac{\partial W}{\partial x}$ is added to the electric potential equation, and $S_T(x)$ (the dimensionless power density) is added to the temperature equation.

Subroutine USRTRN

Subroutine USRTRN is called after the initial guess and at the end of each time step. The routine can be used to monitor the calculation, or to produce special output for each time step. It may also be used at the end of the job to calculate additional quantities from the basic solution variables. In this study subroutine USRTRN is used for calculation of the electric currents (J_x, J_y, J_z.). It allows one to analyse the electric current closure pattern during the post-processing.

6.5 Appendix 5: A sample of a Fortran file

This Fortran file model the configuration in a closed cavity, Ha= 10000, Pe =1.

```
   SUBROUTINE USRBF(IPHASE,BX,BY,BZ,BPX,BPY,BPZ,U,V,W,P,VFRAC,DEN,
   +              VIS,TE,ED,RS,T,H,RF,SCAL,XP,YP,ZP,VOL,AREA,VPOR,
   +              ARPOR,WFACT,IPT,IBLK,IPVERT,IPNODN,IPFACN,IPNODF,
   +              IPNODB,IPFACB,WORK,IWORK,CWORK)
C
C***********************************************************************
C
C  UTILITY SUBROUTINE FOR USER-SUPPLIED BODY FORCES
C
C  >>> IMPORTANT                                         <<<
C  >>>                                                   <<<
C  >>> USERS MAY ONLY ADD OR ALTER PARTS OF THE SUBROUTINE WITHIN  <<<
C  >>> THE DESIGNATED USER AREAS                         <<<
C
C***********************************************************************
C
C  THIS SUBROUTINE IS CALLED BY THE FOLLOWING SUBROUTINES
C    BFCAL
C
C***********************************************************************
C  CREATED
C    24/01/92  ADB
C  MODIFIED
C    03/06/92  PHA  ADD PRECISION FLAG AND CHANGE IVERS TO 2
C    23/11/93  CSH  EXPLICITLY DIMENSION IPVERT ETC.
C    03/02/94  PHA  CHANGE FLOW3D TO CFDS-FLOW3D
C    03/03/94  FHW  CORRECTION OF SPELLING MISTAKE
C    23/03/94  FHW  EXAMPLES COMMENTED OUT
C    09/08/94  NSW  CORRECT SPELLING
C              MOVE 'IF(IUSED.EQ.0) RETURN' OUT OF USER AREA
C    19/12/94  NSW  CHANGE FOR CFX-F3D
C    31/01/97  NSW  EXPLAIN USAGE IN MULTIPHASE FLOWS
C    02/07/97  NSW  UPDATE FOR CFX-4
C
C***********************************************************************
C
C  SUBROUTINE ARGUMENTS
C
C   IPHASE - PHASE NUMBER
C
C  * BX    - X-COMPONENT OF VELOCITY-INDEPENDENT BODY FORCE
C  * BY    - Y-COMPONENT OF VELOCITY-INDEPENDENT BODY FORCE
C  * BZ    - Z-COMPONENT OF VELOCITY-INDEPENDENT BODY FORCE
C  * BPX   -
C  * BPY   - COMPONENTS OF LINEARISABLE BODY FORCES.
C  * BPZ   -
C
C N.B. TOTAL BODY-FORCE IS GIVEN BY:
C
C      X-COMPONENT = BX + BPX*U
C      Y-COMPONENT = BY + BPY*V
C      Z-COMPONENT = BZ + BPZ*W
```

```
C
C     U      - U COMPONENT OF VELOCITY
C     V      - V COMPONENT OF VELOCITY
C     W      - W COMPONENT OF VELOCITY
C     P      - PRESSURE
C     VFRAC  - VOLUME FRACTION
C     DEN    - DENSITY OF FLUID
C     VIS    - VISCOSITY OF FLUID
C     TE     - TURBULENT KINETIC ENERGY
C     ED     - EPSILON
C     RS     - REYNOLD STRESSES
C     T      - TEMPERATURE
C     H      - ENTHALPY
C     SCAL   - SCALARS (THE FIRST 'NCONC' OF THESE ARE MASS FRACTIONS)
C     XP     - X COORDINATES OF CELL CENTRES
C     YP     - Y COORDINATES OF CELL CENTRES
C     ZP     - Z COORDINATES OF CELL CENTRES
C     VOL    - VOLUME OF CELLS
C     AREA   - AREA OF CELLS
C     VPOR   - POROUS VOLUME
C     ARPOR  - POROUS AREA
C     WFACT  - WEIGHT FACTORS
C
C     IPT    - 1D POINTER ARRAY
C     IBLK   - BLOCK SIZE INFORMATION
C     IPVERT - POINTER FROM CELL CENTERS TO 8 NEIGHBOURING VERTICES
C     IPNODN - POINTER FROM CELL CENTERS TO 6 NEIGHBOURING CELLS
C     IPFACN - POINTER FROM CELL CENTERS TO 6 NEIGHBOURING FACES
C     IPNODF - POINTER FROM CELL FACES TO 2 NEIGHBOURING CELL CENTERS
C     IPNODB - POINTER FROM BOUNDARY CENTERS TO CELL CENTERS
C     IPFACB - POINTER FROM BOUNDARY CENTERS TO BOUNDARY FACESS
C
C     WORK   - REAL WORKSPACE ARRAY
C     IWORK  - INTEGER WORKSPACE ARRAY
C     CWORK  - CHARACTER WORKSPACE ARRAY
C
C   SUBROUTINE ARGUMENTS PRECEDED WITH A '*' ARE ARGUMENTS THAT MUST
C   BE SET  BY THE USER IN THIS ROUTINE.
C
C   NOTE THAT OTHER DATA MAY BE OBTAINED FROM CFX-4 USING THE
C   ROUTINE GETADD, FOR FURTHER DETAILS SEE THE VERSION 4
C   USER MANUAL.
C
C***********************************************************************
C
      DOUBLE PRECISION BX
      DOUBLE PRECISION BY
      DOUBLE PRECISION BZ
      DOUBLE PRECISION BPX
      DOUBLE PRECISION BPY
      DOUBLE PRECISION BPZ
      DOUBLE PRECISION U
      DOUBLE PRECISION V
      DOUBLE PRECISION W
      DOUBLE PRECISION P
      DOUBLE PRECISION VFRAC
      DOUBLE PRECISION DEN
      DOUBLE PRECISION VIS
      DOUBLE PRECISION TE
      DOUBLE PRECISION ED
      DOUBLE PRECISION RS
      DOUBLE PRECISION T
      DOUBLE PRECISION H
      DOUBLE PRECISION RF
      DOUBLE PRECISION SCAL
      DOUBLE PRECISION XP
      DOUBLE PRECISION YP
      DOUBLE PRECISION ZP
      DOUBLE PRECISION VOL
      DOUBLE PRECISION AREA
      DOUBLE PRECISION VPOR
      DOUBLE PRECISION ARPOR
      DOUBLE PRECISION WFACT
      DOUBLE PRECISION WORK
      DOUBLE PRECISION SMALL
      DOUBLE PRECISION SORMAX
      DOUBLE PRECISION TIME
      DOUBLE PRECISION DT
      DOUBLE PRECISION DTINVF
      DOUBLE PRECISION TPARM
      LOGICAL LDEN,LVIS,LTURB,LTEMP,LBUOY,LSCAL,LCOMP,LRECT,LCYN,LAXIS,
     +        LPOROS,LTRANS
C
      CHARACTER*(*) CWORK
C
C+++++++++++++++++ USER AREA 1 ++++++++++++++++++++++++++++++++++++++++++
C---- AREA FOR USERS EXPLICITLY DECLARED VARIABLES
      DOUBLE PRECISION TEMPER

C
C+++++++++++++++++ END OF USER AREA 1 ++++++++++++++++++++++++++++++++++++
C
      COMMON /ALL/NBLOCK,NCELL,NBDRY,NNODE,NFACE,NVERT,NDIM,
     +       /ALLWRK/NRWS,NIWS,NCWS,IWRFRE,IWIFRE,IWCFRE,/ADDIMS/NPHASE,
     +       NSCAL,NVAR,NPROP,NDVAR,NDPROP,NDXNN,NDGEOM,NDCOEF,NILIST,
     +       NRLIST,NTOPOL,/CHKUSR/IVERS,IUCALL,IUSED,/DEVICE/NREAD,
     +       NWRITE,NRDISK,NWDISK,/IDUM/ILEN,JLEN,/LOGIC/LDEN,LVIS,
     +       LTURB,LTEMP,LBUOY,LSCAL,LCOMP,LRECT,LCYN,LAXIS,LPOROS,
     +       LTRANS,/MLTGRD/MLEVEL,NLEVEL,ILEVEL,/SGLDBL/IFLGPR,ICHKPR,
     +       /SPARM/SMALL,SORMAX,NITER,INDPRI,MAXIT,NODREF,NODMON,
     +       /TRANSI/NSTEP,KSTEP,MF,INCORE,/TRANSR/TIME,DT,DTINVF,TPARM
C
C+++++++++++++++++ USER AREA 2 ++++++++++++++++++++++++++++++++++++++++++
C---- AREA FOR USERS TO DECLARE THEIR OWN COMMON BLOCKS
C     THESE SHOULD START WITH THE CHARACTERS 'UC' TO ENSURE
C     NO CONFLICT WITH NON-USER COMMON BLOCKS
C
C+++++++++++++++++ END OF USER AREA 2 ++++++++++++++++++++++++++++++++++++
C
      DIMENSION BX(NCELL),BY(NCELL),BZ(NCELL),BPX(NCELL),BPY(NCELL),
     +          BPZ(NCELL)
C
      DIMENSION U(NNODE,NPHASE),V(NNODE,NPHASE),W(NNODE,NPHASE),
     +          P(NNODE,NPHASE),VFRAC(NNODE,NPHASE),DEN(NNODE,NPHASE),
     +          VIS(NNODE,NPHASE),TE(NNODE,NPHASE),ED(NNODE,NPHASE),
     +          RS(NNODE,NPHASE,*),T(NNODE,NPHASE),H(NNODE,NPHASE),
     +          RF(NNODE,NPHASE,4),SCAL(NNODE,NPHASE,NSCAL)
C
      DIMENSION XP(NNODE),YP(NNODE),ZP(NNODE),VOL(NCELL),AREA(NFACE,3),
     +          VPOR(NCELL),ARPOR(NFACE,3),WFACT(NFACE),IPT(*),
     +          IBLK(5,NBLOCK),IPVERT(NCELL,8),IPNODN(NCELL,6),
     +          IPFACN(NCELL,6),IPNODF(NFACE,4),IPNODB(NBDRY,4),
     +          IPFACB(NBDRY),IWORK(*),WORK(*),CWORK(*)
C
C+++++++++++++++++ USER AREA 3 ++++++++++++++++++++++++++++++++++++++++++
C---- AREA FOR USERS TO DIMENSION THEIR ARRAYS
C
C---- AREA FOR USERS TO DEFINE DATA STATEMENTS
C
C+++++++++++++++++ END OF USER AREA 3 +++++++++++++++++++++++++++++++++++
C
C---- STATEMENT FUNCTION FOR ADDRESSING
      IP(I,J,K) = IPT((K-1)*ILEN*JLEN+ (J-1)*ILEN+I)
C
C----VERSION NUMBER OF USER ROUTINE AND PRECISION FLAG
C
      IVERS = 2
      ICHKPR = 2
C
C+++++++++++++++++ USER AREA 4 ++++++++++++++++++++++++++++++++++++++++++
C---- TO USE THIS USER ROUTINE FIRST SET IUSED=1
C
      IUSED = 1
C
C+++++++++++++++++ END OF USER AREA 4 +++++++++++++++++++++++++++++++++++
C
      IF (IUSED.EQ.0) RETURN
C
C---- FRONTEND CHECKING OF USER ROUTINE
      IF (IUCALL.EQ.0) RETURN
C
C+++++++++++++++++ USER AREA 5 ++++++++++++++++++++++++++++++++++++++++++
C
C  THIS ROUTINE IS ENTERED REPEATEDLY FOR EACH PHASE IN A MULTIPHASE
C  CALCULATION. BODY FORCES CAN BE SET FOR A PARTICULAR PHASE USING
C  THE VARIABLE IPHASE. EG. IF (IPHASE.EQ.2) WOULD ALLOW BODY FORCES
C  FOR THE SECOND PHASE.
C
C----ADD USER-DEFINED BODY FORCES.
C
C----EXAMPLE 1: LOCALISED MOMENTUM SOURCE, EG. PROPELLOR.
C
C----USE IPREC TO FIND ADDRESSES
C
C     CALL IPREC('DUCT','BLOCK','CENTRES',IPT
C    +           ,ILEN,JLEN,KLEN,CWORK,IWORK)
C
C     IST = ILEN/2 + 1
C     IFN = IST
C     JST = 1
C     JFN = JLEN/2
C
C     SMOM = 10.0
C     DO 103 K = 1, KLEN
C       DO 102 J = JST, JFN
C         DO 101 I = IST,IFN
C           INODE = IP(I,J,K)
C           BX(INODE) = BX(INODE) + SMOM
C 101       CONTINUE
C 102     CONTINUE
```

```
C 103 CONTINUE
C
C----EXAMPLE 2: LOCALISED RESISTANCE
C
C----USE IPREC TO FIND ADDRESSES
C
C     CALL IPREC('DUCT','BLOCK','CENTRES',IPT
C    +              ,ILEN,JLEN,KLEN,CWORK,IWORK)
C
C     IST = ILEN/4 + 1
C     IFN = 3*ILEN/4
C     JST = 1
C     JFN = JLEN
C
C     RESIST = 1.0E+2
C     DO 203 K = 1, KLEN
C        DO 202 J = JST, JFN
C           DO 201 I = IST,IFN
C              INODE = IP(I,J,K)
C              BPX(INODE) = BPX(INODE) - RESIST
C              BPY(INODE) = BPY(INODE) - RESIST
C              BPZ(INODE) = BPZ(INODE) - RESIST
C 201       CONTINUE
C 202    CONTINUE
C 203 CONTINUE
C
C----EXAMPLE 3: LOCALISED RESISTANCES (DISCONTINUOUS CHANGE)
C
C----USE IPREC TO FIND ADDRESSES
C
C     CALL IPREC('DUCT','BLOCK','CENTRES',IPT
C    +              ,ILEN,JLEN,KLEN,CWORK,IWORK)
C
C     IST1 = ILEN/4 + 1
C     IFN1 = IST1 + ILEN/4 - 1
C     IST2 = IFN1 + 1
C     IFN2 = ILEN - 1
C
C     DO 313 K = 1, KLEN
C        DO 312 J = 1, JLEN
C
C           RESIST = 1.0
C           DO 311 I = IST1,IFN1
C              INODE = IP(I,J,K)
C              BPX(INODE) = BPX(INODE) - RESIST
C              BPY(INODE) = BPY(INODE) - RESIST
C              BPZ(INODE) = BPZ(INODE) - RESIST
C 311       CONTINUE
C
C           RESIST = 10.0
C           DO 321 I = IST2,IFN2
C              INODE = IP(I,J,K)
C              BPX(INODE) = BPX(INODE) - RESIST
C              BPY(INODE) = BPY(INODE) - RESIST
C              BPZ(INODE) = BPZ(INODE) - RESIST
C 321       CONTINUE
C
C 312    CONTINUE
C 313 CONTINUE
C
```

```
      CALL GETVAR('USRBF','SCAL  ',IVAR)
      IPHASE = 1

      CALL SETWRK('USRBF','WORK  ','GRADT ',3*NCELL,JGRADT)

      CALL GRADS('USRBF','SCAL  ',IVAR,IPHASE,SCAL(1,1,1)
     +           ,WORK(JGRADT),XP,YP,ZP,VOL,AREA
     +           ,IBLK,IPVERT,IPNODN,IPFACN,IPNODF,IPNODB
     +           ,IPFACB,WORK,IWORK,CWORK)


      CALL IPALL('*','*','BLOCK','CENTRES',IPT
     +            ,NPT,CWORK,IWORK)

      DO 203 K = 1, NPT
        INODE = IPT(K)
        BX(INODE) = BX(INODE)+WORK(JGRADT+2*NCELL+INODE-1)
        BPX(INODE) = BPX(INODE) + (-1.0D+0)
        BZ(INODE)  = BZ(INODE)  + (-WORK(JGRADT+INODE-1))
        BPZ(INODE) = BPZ(INODE) + (-1.0D+0)


  203 CONTINUE

      CALL DELWRK('USRBF','WORK  ','GRADT ')

C***********************************************************************

C+++++++++++++++++ END OF USER AREA 5 ++++++++++++++++++++++++++++++++++
C
      RETURN
C
      END
      SUBROUTINE USRSRC(IEQN,ICALL,CNAME,CALIAS,AM,SP,SU,CONV,U,V,W,P,
     +                  VFRAC,DEN,VIS,TE,ED,RS,T,H,RF,SCAL,XP,YP,ZP,VOL,
     +                  AREA,VPOR,ARPOR,WFACT,IPT,IBLK,IPVERT,IPNODN,
     +                  IPFACN,IPNODF,IPNODB,IPFACB,WORK,IWORK,CWORK)
C
C**********************************************************************
C
C  UTILITY SUBROUTINE FOR USER-SUPPLIED SOURCES
C
C  >>> IMPORTANT                                                    <<<
C  >>>                                                              <<<
C  >>> USERS MAY ONLY ADD OR ALTER PARTS OF THE SUBROUTINE WITHIN  <<<
C  >>> THE DESIGNATED USER AREAS                                    <<<
C
C**********************************************************************
C
C  THIS SUBROUTINE IS CALLED BY THE FOLLOWING SUBROUTINES
C     CUSR  SCDF  SCDS  SCED  SCENRG  SCHF  SCMOM  SCPCE  SCSCAL
C     SCTE  SCVF
C
C***********************************************************************
C  CREATED
C     08/03/90  ADB
C  MODIFIED
C     04/03/91  ADB  ALTERED ARGUMENT LIST.
C     28/08/91  IRH  NEW STRUCTURE
```

```
C     28/09/91  IRH  CHANGE EXAMPLE + ADD COMMON BLOCKS
C     10/02/92  PHA  UPDATE CALLED BY COMMENT, ADD RF ARGUMENT,
C                    CHANGE LAST DIMENSION OF RS TO 6 AND IVERS TO 2
C     03/06/92  PHA  ADD PRECISION FLAG AND CHANGE IVERS TO 3
C     23/11/93  CSH  EXPLICITLY DIMENSION IPVERT ETC.
C     07/12/93  NSW  INCLUDE CONV IN ARGUMENT LIST AND CHANGE IVERS
C                    TO 4
C     03/02/94  PHA  CHANGE FLOW3D TO CFDS-FLOW3D
C     03/03/94  FHW  CORRECTION OF SPELLING MISTAKE
C     08/03/94  NSW  CORRECT SPELLING
C     09/08/94  NSW  CORRECT SPELLING.
C                    MOVE 'IF(IUSED.EQ.0) RETURN' OUT OF USER AREA.
C                    INCLUDE COMMENT ON MASS SOURCES.
C     19/12/94  NSW  CHANGE FOR CFX-F3D
C     02/07/97  NSW  UPDATE FOR CFX-4
C
C**********************************************************************
C
C  SUBROUTINE ARGUMENTS
C
C   IEQN   - EQUATION NUMBER
C   ICALL  - SUBROUTINE CALL
C   CNAME  - EQUATION NAME
C   CALIAS - ALIAS OF EQUATION NAME
C   AM     - OFF DIAGONAL MATRIX COEFFICIENTS
C   SU     - SU IN LINEARISATION OF SOURCE TERM
C   SP     - SP IN LINEARISATION OF SOURCE TERM
C   CONV   - CONVECTION COEFFICIENTS
C   U      - U COMPONENT OF VELOCITY
C   V      - V COMPONENT OF VELOCITY
C   W      - W COMPONENT OF VELOCITY
C   P      - PRESSURE
C   VFRAC  - VOLUME FRACTION
C   DEN    - DENSITY OF FLUID
C   VIS    - VISCOSITY OF FLUID
C   TE     - TURBULENT KINETIC ENERGY
C   ED     - EPSILON
C   RS     - REYNOLD STRESSES
C   T      - TEMPERATURE
C   H      - ENTHALPY
C   RF     - REYNOLD FLUXES
C   SCAL   - SCALARS (THE FIRST 'NCONC' OF THESE ARE MASS FRACTIONS)
C   XP     - X COORDINATES OF CELL CENTRES
C   YP     - Y COORDINATES OF CELL CENTRES
C   ZP     - Z COORDINATES OF CELL CENTRES
C   VOL    - VOLUME OF CELLS
C   AREA   - AREA OF CELLS
C   VPOR   - POROUS VOLUME
C   ARPOR  - POROUS AREA
C   WFACT  - WEIGHT FACTORS
C
C   IPT    - 1D POINTER ARRAY
C   IBLK   - BLOCK SIZE INFORMATION
C   IPVERT - POINTER FROM CELL CENTERS TO 8 NEIGHBOURING VERTICES
C   IPNODN - POINTER FROM CELL CENTERS TO 6 NEIGHBOURING CELLS
C   IPFACN - POINTER FROM CELL CENTERS TO 6 NEIGHBOURING FACES
C   IPNODF - POINTER FROM CELL FACES TO 2 NEIGHBOURING CELL CENTERS
C   IPNODB - POINTER FROM BOUNDARY CENTERS TO CELL CENTERS
C   IPFACB - POINTER FROM BOUNDARY CENTERS TO BOUNDARY FACESS
C
```

```
C     WORK   - REAL WORKSPACE ARRAY
C     IWORK  - INTEGER WORKSPACE ARRAY
C     CWORK  - CHARACTER WORKSPACE ARRAY
C
C   SUBROUTINE ARGUMENTS PRECEDED WITH A '*' ARE ARGUMENTS THAT MUST
C   BE SET  BY THE USER IN THIS ROUTINE.
C
C   NOTE THAT WHEN USING MASS SOURCES, THE FLOWS THROUGH MASS FLOW
C   BOUNDARIES ARE UNCHANGED. THE USER SHOULD THEREFORE INCLUDE AT
C   LEAST ONE PRESSURE BOUNDARY FOR SUCH A CALCULATION.
C
C   NOTE THAT OTHER DATA MAY BE OBTAINED FROM CFX-4 USING THE
C   ROUTINE GETADD, FOR FURTHER DETAILS SEE THE VERSION 4
C   USER MANUAL.
C
C***********************************************************************
C
      DOUBLE PRECISION AM
      DOUBLE PRECISION SP
      DOUBLE PRECISION SU
      DOUBLE PRECISION CONV
      DOUBLE PRECISION U
      DOUBLE PRECISION V
      DOUBLE PRECISION W
      DOUBLE PRECISION P
      DOUBLE PRECISION VFRAC
      DOUBLE PRECISION DEN
      DOUBLE PRECISION VIS
      DOUBLE PRECISION TE
      DOUBLE PRECISION ED
      DOUBLE PRECISION RS
      DOUBLE PRECISION T
      DOUBLE PRECISION H
      DOUBLE PRECISION RF
      DOUBLE PRECISION SCAL
      DOUBLE PRECISION XP
      DOUBLE PRECISION YP
      DOUBLE PRECISION ZP
      DOUBLE PRECISION VOL
      DOUBLE PRECISION AREA
      DOUBLE PRECISION VPOR
      DOUBLE PRECISION ARPOR
      DOUBLE PRECISION WFACT
      DOUBLE PRECISION WORK
      DOUBLE PRECISION SMALL
      DOUBLE PRECISION SORMAX
      DOUBLE PRECISION TIME
      DOUBLE PRECISION DT
      DOUBLE PRECISION DTINVF
      DOUBLE PRECISION TPARM
      LOGICAL LDEN,LVIS,LTURB,LTEMP,LBUOY,LSCAL,LCOMP,LRECT,LCYN,LAXIS,
     +        LPOROS,LTRANS
C
      CHARACTER*(*) CWORK
      CHARACTER CNAME*6,CALIAS*24
C
C+++++++++++++++++ USER AREA 1 +++++++++++++++++++++++++++++++++++++++++++
C---- AREA FOR USERS EXPLICITLY DECLARED VARIABLES
      DOUBLE PRECISION THESRC
```

```
C
C+++++++++++++++++ END OF USER AREA 1 ++++++++++++++++++++++++++++++++++
C
      COMMON /ALL/NBLOCK,NCELL,NBDRY,NNODE,NFACE,NVERT,NDIM,
     +       /ALLWRK/NRWS,NIWS,NCWS,IWRFRE,IWIFRE,IWCFRE,/ADDIMS/NPHASE,
     +       NSCAL,NVAR,NPROP,NDVAR,NDPROP,NDXNN,NDGEOM,NDCOEF,NILIST,
     +       NRLIST,NTOPOL,/CHKUSR/IVERS,IUCALL,IUSED,/DEVICE/NREAD,
     +       NWRITE,NRDISK,NWDISK,/IDUM/ILEN,JLEN,/LOGIC/LDEN,LVIS,
     +       LTURB,LTEMP,LBUOY,LSCAL,LCOMP,LRECT,LCYN,LAXIS,LPOROS,
     +       LTRANS,/MLTGRD/MLEVEL,NLEVEL,ILEVEL,/SGLDBL/IFLGPR,ICHKPR,
     +       /SPARM/SMALL,SORMAX,NITER,INDPRI,MAXIT,NODREF,NODMON,
     +       /TRANSI/NSTEP,KSTEP,MF,INCORE,/TRANSR/TIME,DT,DTINVF,TPARM
C
C+++++++++++++++++ USER AREA 2 +++++++++++++++++++++++++++++++++++++++++
C---- AREA FOR USERS TO DECLARE THEIR OWN COMMON BLOCKS
C     THESE SHOULD START WITH THE CHARACTERS 'UC' TO ENSURE
C     NO CONFLICT WITH NON-USER COMMON BLOCKS
C
C+++++++++++++++++ END OF USER AREA 2 ++++++++++++++++++++++++++++++++++
C
      DIMENSION AM(NCELL,6,NPHASE),SP(NCELL,NPHASE),SU(NCELL,NPHASE),
     +          CONV(NFACE,NPHASE)
C
      DIMENSION U(NNODE,NPHASE),V(NNODE,NPHASE),W(NNODE,NPHASE),
     +          P(NNODE,NPHASE),VFRAC(NNODE,NPHASE),DEN(NNODE,NPHASE),
     +          VIS(NNODE,NPHASE),TE(NNODE,NPHASE),ED(NNODE,NPHASE),
     +          RS(NNODE,NPHASE,6),T(NNODE,NPHASE),H(NNODE,NPHASE),
     +          RF(NNODE,NPHASE,4),SCAL(NNODE,NPHASE,NSCAL)
C
      DIMENSION XP(NNODE),YP(NNODE),ZP(NNODE),VOL(NCELL),AREA(NFACE,3),
     +          VPOR(NCELL),ARPOR(NFACE,3),WFACT(NFACE),IPT(*),
     +          IBLK(5,NBLOCK),IPVERT(NCELL,8),IPNODN(NCELL,6),
     +          IPFACN(NCELL,6),IPNODF(NFACE,4),IPNODB(NBDRY,4),
     +          IPFACB(NBDRY),IWORK(*),WORK(*),CWORK(*)
C
C+++++++++++++++++ USER AREA 3 +++++++++++++++++++++++++++++++++++++++++
C---- AREA FOR USERS TO DIMENSION THEIR ARRAYS
C
C---- AREA FOR USERS TO DEFINE DATA STATEMENTS
C
C+++++++++++++++++ END OF USER AREA 3 ++++++++++++++++++++++++++++++++++
C
C---- STATEMENT FUNCTION FOR ADDRESSING
      IP(I,J,K) = IPT((K-1)*ILEN*JLEN+ (J-1)*ILEN+I)
C
C----VERSION NUMBER OF USER ROUTINE AND PRECISION FLAG
C
      IVERS = 4
      ICHKPR = 2
C
C+++++++++++++++++ USER AREA 4 +++++++++++++++++++++++++++++++++++++++++
C---- TO USE THIS USER ROUTINE FIRST SET IUSED=1
C
      IUSED = 1
C
C+++++++++++++++++ END OF USER AREA 4 ++++++++++++++++++++++++++++++++++
C
      IF (IUSED.EQ.0) RETURN
C
C---- FRONTEND CHECKING OF USER ROUTINE
```

```
      IF (IUCALL.EQ.0) RETURN
C
C---- ADD TO SOURCE TERMS
      IF (ICALL.EQ.1) THEN
C
C+++++++++++++++++ USER AREA 5 +++++++++++++++++++++++++++++++++++++++++++++
C
C---- EXAMPLE (HEAT SOURCE) ADD 100W PER UNIT VOLUME IN BLOCK
C       'BLOCK-NUMBER-2'
C
C  USE IPREC TO FIND ADDRESSES
C
C       CALL IPREC('BLOCK-NUMBER-2','BLOCK','CENTRES',IPT,ILEN,JLEN,KLEN,
C      +           CWORK,IWORK)
C
C  FIND VARIABLE NUMBER FOR ENTHALPY
C       CALL GETVAR('USRSRC','H     ',IVAR)
C  IF ENTHALPY EQUATION ADD SOURCE TERMS
C       IF (IVAR.EQ.IEQN) THEN
C  LOOP OVER PATCH
C       DO 103 K = 1, KLEN
C         DO 102 J = 1, JLEN
C           DO 101 I = 1, ILEN
C  USE STATEMENT FUNCTION IP TO GET ADDRESSES
C           INODE = IP(I,J,K)
C  ADD HEAT SOURCE
C           SU(INODE,1)=SU(INODE,1)+100.0*VOL(INODE)
C 101     CONTINUE
C 102   CONTINUE
C 103  CONTINUE
C       ENDIF
C
C---- END OF EXAMPLE
C**************************************************************************
C                                                  HEAT SOURCE
C***************************************************************************

      CALL GETVAR('USRSRC','H     ',IVAR)
       IPHASE = 1

      IF (IVAR.EQ.IEQN) THEN
        CALL IPALL('*','*','BLOCK','CENTRES',IPT,NPT,CWORK,IWORK)

        DO 300 I=1,NPT
         INODE=IPT(I)
         THESRC = 1+((XP(INODE)+3.365D0)/9.9423E-01)**2.0D0
         THESRC = 1.659645E+02/THESRC
         SU(INODE,1)=SU(INODE,1)+VOL(INODE)*THESRC
  300   CONTINUE
      ENDIF


C********************************************************************************
C                                         POTENTIAL SOURCE
C********************************************************************************
      CALL GETVAR('USRSRC','SCAL  ',IVAR)
      IPHASE = 1

      IF (IVAR.EQ.IEQN) THEN
      CALL SETWRK('USRSRC','WORK  ','UGRAD ',3*NCELL,JUGRAD)
      CALL SETWRK('USRSRC','WORK  ','VGRAD ',3*NCELL,JVGRAD)
      CALL SETWRK('USRSRC','WORK  ','WGRAD ',3*NCELL,JWGRAD)

      CALL GRADV('USRSRC',IPHASE,U(1,IPHASE),V(1,IPHASE)
     +         ,W(1,IPHASE),WORK(JUGRAD),WORK(JVGRAD)
     +         ,WORK(JWGRAD),XP,YP,ZP,VOL,AREA
     +         ,IBLK,IPVERT,IPNODN,IPFACN,IPNODF,IPNODB
     +         ,IPFACB,WORK,IWORK,CWORK)
       CALL IPALL('*','*','BLOCK','CENTRES',IPT,NPT,CWORK,IWORK)

       DO 200 I=1,NPT
        INODE=IPT(I)
        THESRC =
     +      (-WORK(JWGRAD+INODE-1))
     +       +WORK(JUGRAD+2*NCELL+INODE-1)
        SU(INODE,1)=SU(INODE,1)-VOL(INODE)*THESRC
  200   CONTINUE
      CALL DELWRK('USRSRC','WORK  ','UGRAD ')

      ENDIF

C*********************************************************************

C+++++++++++++++++ END OF USER AREA 5 +++++++++++++++++++++++++++++++++
      END IF
C
C---- OVERWRITE SOURCE TERMS
      IF (ICALL.EQ.2) THEN
C
C+++++++++++++++++ USER AREA 6 +++++++++++++++++++++++++++++++++++++++++
C
C---- EXAMPLE (HEAT SOURCE) OVERWRITE WITH 100W PER UNIT VOLUME IN
C                       ALL INTERIOR CELLS
C
C     CALL GETVAR('USRSRC','H     ',IVAR)
C
C     IF (IVAR.EQ.IEQN) THEN
C  USE IPALL TO FIND 1D ADDRESSES OF ALL CELL CENTRES
C      CALL IPALL('*','*','BLOCK','CENTRES',IPT,NPT,CWORK,IWORK)
C  LOOP OVER ALL INTERIOR CELLS
C      DO 200 I=1,NPT
C  USE ARRAY IPT TO GET ADDRESS
C       INODE=IPT(I)
C  OVERWRITE SOURCE TERMS
C       SU(INODE,1)=100.0*VOL(INODE)
C 200 CONTINUE
C      ENDIF
C
C---- END OF EXAMPLE
C+++++++++++++++++ END OF USER AREA 6 +++++++++++++++++++++++++++++++++
C
      END IF
C
      RETURN
C
      END

      SUBROUTINE USRTRN(U,V,W,P,VFRAC,DEN,VIS,TE,ED,RS,T,H,RF,SCAL,XP,
     +                 YP,ZP,VOL,AREA,VPOR,ARPOR,WFACT,CONV,IPT,IBLK,
     +               IPVERT,IPNODN,IPFACN,IPNODF,IPNODB,IPFACB,WORK,
     +               IWORK,CWORK)
C
C***********************************************************************
C
C   USER SUBROUTINE TO ALLOW USERS TO MODIFY OR MONITOR THE SOLUTION
AT
C  THE END OF EACH TIME STEP
C  THIS SUBROUTINE IS CALLED BEFORE THE START OF THE RUN AS WELL AS AT
C  THE END OF EACH TIME STEP
C
C  >>> IMPORTANT                                               <<<
C  >>>                                                         <<<
C  >>> USERS MAY ONLY ADD OR ALTER PARTS OF THE SUBROUTINE WITHIN  <<<
C  >>> THE DESIGNATED USER AREAS                               <<<
C
C***********************************************************************
C
C  THIS SUBROUTINE IS CALLED BY THE FOLLOWING SUBROUTINES
C     CUSR  TRNMOD
C
C************************************************************************
C  CREATED
C     27/04/90  ADB
C  MODIFIED
C     05/08/91  IRH  NEW STRUCTURE
C     01/10/91  DSC  REDUCE COMMENT LINE GOING OVER COLUMN 72.
C     29/11/91  PHA  UPDATE CALLED BY COMMENT, ADD RF ARGUMENT,
C                    CHANGE LAST DIMENSION OF RS TO 6 AND IVERS TO 2
C     05/06/92  PHA  ADD PRECISION FLAG AND CHANGE IVERS TO 3
C     03/07/92  DSC  CORRECT COMMON MLTGRD.
C     23/11/93  CSH  EXPLICITLY DIMENSION IPVERT ETC.
C     03/02/94  PHA  CHANGE FLOW3D TO CFDS-FLOW3D
C     22/08/94  NSW  MOVE 'IF(IUSED.EQ.0) RETURN' OUT OF USER AREA
C     19/12/94  NSW  CHANGE FOR CFX-F3D
C     02/07/97  NSW  UPDATE FOR CFX-4
C     02/07/99  NSW  INCLUDE NEW EXAMPLE FOR CALCULATING FLUX OF A
C                    SCALAR AT A PRESSURE BOUNDARY
C
C************************************************************************
C
C  SUBROUTINE ARGUMENTS
C
C     U      - U COMPONENT OF VELOCITY
C     V      - V COMPONENT OF VELOCITY
C     W      - W COMPONENT OF VELOCITY
C     P      - PRESSURE
C     VFRAC  - VOLUME FRACTION
C     DEN    - DENSITY OF FLUID
C     VIS    - VISCOSITY OF FLUID
C     TE     - TURBULENT KINETIC ENERGY
C     ED     - EPSILON
C     RS     - REYNOLD STRESSES
C     T      - TEMPERATURE
C     H      - ENTHALPY
C     RF     - REYNOLD FLUXES
C     SCAL   - SCALARS (THE FIRST 'NCONC' OF THESE ARE MASS FRACTIONS)
C     XP     - X COORDINATES OF CELL CENTRES
C     YP     - Y COORDINATES OF CELL CENTRES
C     ZP     - Z COORDINATES OF CELL CENTRES
```

```
C     VOL    - VOLUME OF CELLS
C     AREA   - AREA OF CELLS
C     VPOR   - POROUS VOLUME
C     ARPOR  - POROUS AREA
C     WFACT  - WEIGHT FACTORS
C     CONV   - CONVECTION COEFFICIENTS
C
C     IPT    - 1D POINTER ARRAY
C     IBLK   - BLOCK SIZE INFORMATION
C     IPVERT - POINTER FROM CELL CENTERS TO 8 NEIGHBOURING VERTICES
C     IPNODN - POINTER FROM CELL CENTERS TO 6 NEIGHBOURING CELLS
C     IPFACN - POINTER FROM CELL CENTERS TO 6 NEIGHBOURING FACES
C     IPNODF - POINTER FROM CELL FACES TO 2 NEIGHBOURING CELL CENTERS
C     IPNODB - POINTER FROM BOUNDARY CENTERS TO CELL CENTERS
C     IPFACB - POINTER FROM BOUNDARY CENTERS TO BOUNDARY FACESS
C
C     WORK   - REAL WORKSPACE ARRAY
C     IWORK  - INTEGER WORKSPACE ARRAY
C     CWORK  - CHARACTER WORKSPACE ARRAY
C
C   SUBROUTINE ARGUMENTS PRECEDED WITH A '*' ARE ARGUMENTS THAT MUST
C   BE SET  BY THE USER IN THIS ROUTINE.
C
C   NOTE THAT OTHER DATA MAY BE OBTAINED FROM CFX-4 USING THE
C   ROUTINE GETADD, FOR FURTHER DETAILS SEE THE VERSION 4
C   USER MANUAL.
C
C***********************************************************************
C
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DOUBLE PRECISION U
      DOUBLE PRECISION V
      DOUBLE PRECISION W
      DOUBLE PRECISION P
      DOUBLE PRECISION VFRAC
      DOUBLE PRECISION DEN
      DOUBLE PRECISION VIS
      DOUBLE PRECISION TE
      DOUBLE PRECISION ED
      DOUBLE PRECISION RS
      DOUBLE PRECISION T
      DOUBLE PRECISION H
      DOUBLE PRECISION RF
      DOUBLE PRECISION SCAL
      DOUBLE PRECISION XP
      DOUBLE PRECISION YP
      DOUBLE PRECISION ZP
      DOUBLE PRECISION VOL
      DOUBLE PRECISION AREA
      DOUBLE PRECISION VPOR
      DOUBLE PRECISION ARPOR
      DOUBLE PRECISION WFACT
      DOUBLE PRECISION CONV
      DOUBLE PRECISION WORK
      DOUBLE PRECISION SMALL
      DOUBLE PRECISION SORMAX
      DOUBLE PRECISION DTUSR
      DOUBLE PRECISION TIME
      DOUBLE PRECISION DT
      DOUBLE PRECISION DTINVF
      DOUBLE PRECISION TPARM
      DOUBLE PRECISION SGNWL
      LOGICAL LDEN,LVIS,LTURB,LTEMP,LBUOY,LSCAL,LCOMP,LRECT,LCYN,LAXIS,
     +        LPOROS,LTRANS
C
      CHARACTER*(*) CWORK
C
C++++++++++++++++ USER AREA 1 +++++++++++++++++++++++++++++++++++++++++
C---- AREA FOR USERS EXPLICITLY DECLARED VARIABLES
      DOUBLE PRECISION SUMU, SUMP, ALFAN, CN, BETAN, GAMMAN, SGN
      DOUBLE PRECISION CS1, CS2, SN1, SN2, CH1, CH2, SH1, SH2
      DOUBLE PRECISION CSG, CHB, DN, EN, KN, DPN
      DOUBLE PRECISION FRAC1, FRAC2, FRAC3, FRAC4
      DOUBLE PRECISION E1, E2, E3, E4, EB, UN, PN
C
C++++++++++++++++ END OF USER AREA 1 ++++++++++++++++++++++++++++++++++++
C
      COMMON /ALL/NBLOCK,NCELL,NBDRY,NNODE,NFACE,NVERT,NDIM,
     +       /ALLWRK/NRWS,NIWS,NCWS,IWRFRE,IWIFRE,IWCFRE,/ADDIMS/NPHASE,
     +       NSCAL,NVAR,NPROP,NDVAR,NDPROP,NDXNN,NDGEOM,NDCOEF,NILIST,
     +       NRLIST,NTOPOL,/CHKUSR/IVERS,IUCALL,IUSED,/CONC/NCONC,
     +       /DEVICE/NREAD,NWRITE,NRDISK,NWDISK,/IDUM/ILEN,JLEN,
     +       /LOGIC/LDEN,LVIS,LTURB,LTEMP,LBUOY,LSCAL,LCOMP,LRECT,LCYN,
     +       LAXIS,LPOROS,LTRANS,/MLTGRD/MLEVEL,NLEVEL,ILEVEL,
     +       /SGLDBL/IFLGPR,ICHKPR,/SPARM/SMALL,SORMAX,NITER,INDPRI,
     +       MAXIT,NODREF,NODMON,/TIMUSR/DTUSR,/TRANSI/NSTEP,KSTEP,MF,
     +       INCORE,/TRANSR/TIME,DT,DTINVF,TPARM
C
C++++++++++++++++ USER AREA 2 +++++++++++++++++++++++++++++++++++++++++
C---- AREA FOR USERS TO DECLARE THEIR OWN COMMON BLOCKS
C     THESE SHOULD START WITH THE CHARACTERS 'UC' TO ENSURE
C     NO CONFLICT WITH NON-USER COMMON BLOCKS
C
C++++++++++++++++ END OF USER AREA 2 ++++++++++++++++++++++++++++++++++++
C
      DIMENSION U(NNODE,NPHASE),V(NNODE,NPHASE),W(NNODE,NPHASE),
     +          P(NNODE,NPHASE),VFRAC(NNODE,NPHASE),DEN(NNODE,NPHASE),
     +          VIS(NNODE,NPHASE),TE(NNODE,NPHASE),ED(NNODE,NPHASE),
     +          RS(NNODE,NPHASE,6),T(NNODE,NPHASE),H(NNODE,NPHASE),
     +          RF(NNODE,NPHASE,4),SCAL(NNODE,NPHASE,NSCAL)
      DIMENSION XP(NNODE),YP(NNODE),ZP(NNODE),VOL(NCELL),AREA(NFACE,3),
     +          VPOR(NCELL),ARPOR(NFACE,3),WFACT(NFACE),
     +          CONV(NFACE,NPHASE),IPT(*),IBLK(5,NBLOCK),
     +          IPVERT(NCELL,8),IPNODN(NCELL,6),IPFACN(NCELL,6),
     +          IPNODF(NFACE,4),IPNODB(NBDRY,4),IPFACB(NBDRY),IWORK(*),
     +          WORK(*),CWORK(*)
      DIMENSION SGNWL(6)
C
C++++++++++++++++ USER AREA 3 +++++++++++++++++++++++++++++++++++++++++
C---- AREA FOR USERS TO DIMENSION THEIR ARRAYS
C
C---- AREA FOR USERS TO DEFINE DATA STATEMENTS
C
C++++++++++++++++ END OF USER AREA 3 ++++++++++++++++++++++++++++++++++++
C
      DATA SGNWL/1.0D0,1.0D0,1.0D0,-1.0D0,-1.0D0,-1.0D0/
C
C---- STATEMENT FUNCTION FOR ADDRESSING
      IP(I,J,K) = IPT((K-1)*ILEN*JLEN+ (J-1)*ILEN+I)
```

```
C
C----VERSION NUMBER OF USER ROUTINE AND PRECISION FLAG
C
      IVERS = 3
      ICHKPR = 2
C
C+++++++++++++++++ USER AREA 4 +++++++++++++++++++++++++++++++++++++++++++
C---- TO USE THIS USER ROUTINE FIRST SET IUSED=1
C
      IUSED = 1
C
C+++++++++++++++++ END OF USER AREA 4 ++++++++++++++++++++++++++++++++++++
C
      IF (IUSED.EQ.0) RETURN
C
C---- FRONTEND CHECKING OF USER ROUTINE
      IF (IUCALL.EQ.0) RETURN
C
C+++++++++++++++++ USER AREA 5 +++++++++++++++++++++++++++++++++++++++++++
C
C---- EXAMPLE (SET TIME INCREMENT FOR NEXT TIME STEP)
C
C     DTUSR = 0.1
C
C---- END OF EXAMPLE
C
C---- EXAMPLE (CALCULATE FLUX OF FIRST SCALAR AT A PRESSURE BOUNDARY)
C
C     IPHASE = 1
C     FLUX = 0.0
C  USE IPALL TO FIND ADDRESSES OF BOUNDARY NODES ON PATCH PRESS1
C     CALL IPALL('PRESS1','PRESS','PATCH','CENTRES'
C    +           ,IPT,NPT,CWORK,IWORK)
C  LOOP OVER ALL BOUNDARY NODES
C     DO 300 I=1,NPT
C  USE ARRAY IPT TO GET ADDRESS
C       INODE = IPT(I)
C       IBDRY = INODE - NCELL
C       IFACE = IPFACB(IBDRY)
C       INDUM = IPNODB(IBDRY,2)
C       NWL   = IPNODB(IBDRY,4)
C       FLUX  = FLUX
C    +        + SGNWL(NWL)*CONV(IFACE,IPHASE)*SCAL(INDUM,IPHASE,1)
C 300 CONTINUE
C
C---- END OF EXAMPLE


      IF (NITER.GT.1) THEN


      IPHASE = 1

      CALL GETVAR('USRTRN','SCAL  ',IT)

      CALL SETWRK('USRTRN','WORK  ','GRADT ',3*NCELL,JGRADT)

      CALL GRADS('USRTRN','SCAL  ',IT,IPHASE,SCAL(1,1,1)
     +           ,WORK(JGRADT),XP,YP,ZP,VOL,AREA
     +           ,IBLK,IPVERT,IPNODN,IPFACN,IPNODF,IPNODB
     +           ,IPFACB,WORK,IWORK,CWORK)

      CALL IPALL('*','*','BLOCK','CENTRES',IPT
     +           ,NPT,CWORK,IWORK)

      DO 203 K = 1, NPT
        INODE=IPT(K)
        SCAL(INODE,1,2)=-WORK(JGRADT+INODE-1)-W(INODE,1)
        SCAL(INODE,1,3)=-WORK(JGRADT+NCELL+INODE-1)
        SCAL(INODE,1,4)=-WORK(JGRADT+2*NCELL+INODE-1)+U(INODE,1)
  203 CONTINUE

      CALL DELWRK('USRTRN','WORK  ','GRADT ')


           ENDIF


C+++++++++++++++++ END OF USER AREA 5 +++++++++++++++++++++++++++++++++++
C
      RETURN

      END
      SUBROUTINE USRBCS(VARBCS,VARAMB,A,B,C,ACND,BCND,CCND,IWGVEL,
     +                  NDVWAL,FLOUT,NLABEL,NSTART,NEND,NCST,NCEN,U,V,W,
     +                  P,VFRAC,DEN,VIS,TE,ED,RS,T,H,RF,SCAL,XP,YP,ZP,
     +                  VOL,AREA,VPOR,ARPOR,WFACT,IPT,IBLK,IPVERT,
     +                  IPNODN,IPFACN,IPNODF,IPNODB,IPFACB,WORK,IWORK,
     +                  CWORK)
C
C***********************************************************************
C
C  USER ROUTINE TO SET REALS AT BOUNDARIES.
C
C   >>> IMPORTANT                                              <<<
C   >>>                                                        <<<
C   >>> USERS MAY ONLY ADD OR ALTER PARTS OF THE SUBROUTINE WITHIN  <<<
C   >>> THE DESIGNATED USER AREAS                              <<<
C
C***********************************************************************
C
C  THIS SUBROUTINE IS CALLED BY THE FOLLOWING SUBROUTINE
C    CUSR  SRLIST
C
C***********************************************************************
C   CREATED
C      30/11/88 ADB
C   MODIFIED
C      08/09/90 ADB  RESTRUCTURED FOR USER-FRIENDLINESS.
C      10/08/91 IRH  FURTHER RESTRUCTURING ADD ACND BCND CCND
C      22/09/91 IRH  CHANGE ICALL TO IUCALL + ADD /SPARM/
C      10/03/92 PHA  UPDATE CALLED BY COMMENT, ADD RF ARGUMENT,
C                    CHANGE LAST DIMENSION OF RS TO 6 AND IVERS TO 2
C      03/06/92 PHA  ADD PRECISION FLAG AND CHANGE IVERS TO 3
C      30/06/92 NSW  INCLUDE FLAG FOR CALLING BY ITERATION
C                    INSERT EXTRA COMMENTS
C      03/08/92 NSW  MODIFY DIMENSION STATEMENTS FOR VAX
C      21/12/92 CSH  INCREASE IVERS TO 4
C      02/08/93 NSW  INCORRECT AND MISLEADING COMMENT REMOVED
C      05/11/93 NSW  INDICATE USE OF FLOUT IN MULTIPHASE FLOWS
```

```
C     23/11/93 CSH  EXPLICITLY DIMENSION IPVERT ETC.
C     01/02/94 NSW  SET VARIABLE POINTERS IN WALL EXAMPLE.
C                   CHANGE FLOW3D TO CFDS-FLOW3D.
C                   MODIFY MULTIPHASE MASS FLOW BOUNDARY TREATMENT.
C     03/03/94 FHW  CORRECTION OF SPELLING MISTAKE
C     02/07/94 BAS  SLIDING GRIDS - ADD NEW ARGUMENT IWGVEL
C                   TO ALLOW VARIANTS OF TRANSIENT-GRID WALL BC
C                   CHANGE VERSION NUMBER TO 5
C     09/08/94 NSW  CORRECT SPELLING
C                   MOVE 'IF(IUSED.EQ.0) RETURN' OUT OF USER AREA
C     19/12/94 NSW  CHANGE FOR CFX-F3D
C     02/02/95 NSW  CHANGE COMMON /IMFBMP/
C     02/06/97 NSW  MAKE EXAMPLE MORE LOGICAL
C     02/07/97 NSW  UPDATE FOR CFX-4
C     08/09/98 NSW  CORRECT SIZE OF WALL ARRAY IN COMMENT
C     22/05/00 NSW  INITIALISE IUBCSF
C
C***********************************************************************
C
C   SUBROUTINE ARGUMENTS
C
C    VARBCS - REAL BOUNDARY CONDITIONS
C    VARAMB - AMBIENT VALUE OF VARIABLES
C    A      - COEFFICIENT IN WALL BOUNDARY CONDITION
C    B      - COEFFICIENT IN WALL BOUNDARY CONDITION
C    C      - COEFFICIENT IN WALL BOUNDARY CONDITION
C    ACND   - COEFFICIENT IN CONDUCTING WALL BOUNDARY CONDITION
C    BCND   - COEFFICIENT IN CONDUCTING WALL BOUNDARY CONDITION
C    CCND   - COEFFICIENT IN CONDUCTING WALL BOUNDARY CONDITION
C    IWGVEL - USAGE OF INPUT VELOCITIES (0 = AS IS,1 = ADD GRID MOTION)
C    NDVWAL - FIRST DIMENSION OF ARRAY IWGVEL
C    FLOUT  - MASS FLOW/FRACTIONAL MASS FLOW
C    NLABEL - NUMBER OF DISTINCT OUTLETS
C    NSTART - ARRAY POINTER
C    NEND   - ARRAY POINTER
C    NCST   - ARRAY POINTER
C    NCEN   - ARRAY POINTER
C    U      - U COMPONENT OF VELOCITY
C    V      - V COMPONENT OF VELOCITY
C    W      - W COMPONENT OF VELOCITY
C    P      - PRESSURE
C    VFRAC  - VOLUME FRACTION
C    DEN    - DENSITY OF FLUID
C    VIS    - VISCOSITY OF FLUID
C    TE     - TURBULENT KINETIC ENERGY
C    ED     - EPSILON
C    RS     - REYNOLD STRESSES
C    T      - TEMPERATURE
C    H      - ENTHALPY
C    RF     - REYNOLD FLUXES
C    SCAL   - SCALARS (THE FIRST 'NCONC' OF THESE ARE MASS FRACTIONS)
C    XP     - X COORDINATES OF CELL CENTRES
C    YP     - Y COORDINATES OF CELL CENTRES
C    ZP     - Z COORDINATES OF CELL CENTRES
C    VOL    - VOLUME OF CELLS
C    AREA   - AREA OF CELLS
C    VPOR   - POROUS VOLUME
C    ARPOR  - POROUS AREA
C    WFACT  - WEIGHT FACTORS
C
```

```
C     IPT    - 1D POINTER ARRAY
C     IBLK   - BLOCK SIZE INFORMATION
C     IPVERT - POINTER FROM CELL CENTERS TO 8 NEIGHBOURING VERTICES
C     IPNODN - POINTER FROM CELL CENTERS TO 6 NEIGHBOURING CELLS
C     IPFACN - POINTER FROM CELL CENTERS TO 6 NEIGHBOURING FACES
C     IPNODF - POINTER FROM CELL FACES TO 2 NEIGHBOURING CELL CENTERS
C     IPNODB - POINTER FROM BOUNDARY CENTERS TO CELL CENTERS
C     IPFACB - POINTER TO NODES FROM BOUNDARY FACES
C
C     WORK   - REAL WORKSPACE ARRAY
C     IWORK  - INTEGER WORKSPACE ARRAY
C     CWORK  - CHARACTER WORKSPACE ARRAY
C
C   SUBROUTINE ARGUMENTS PRECEDED WITH A '*' ARE ARGUMENTS THAT MUST
C   BE SET  BY THE USER IN THIS ROUTINE.
C
C   NOTE THAT OTHER DATA MAY BE OBTAINED FROM CFX-4 USING THE
C   ROUTINE GETADD, FOR FURTHER DETAILS SEE THE VERSION 4
C   USER MANUAL.
C
C***********************************************************************
      DOUBLE PRECISION VARBCS
      DOUBLE PRECISION VARAMB
      DOUBLE PRECISION A
      DOUBLE PRECISION B
      DOUBLE PRECISION C
      DOUBLE PRECISION ACND
      DOUBLE PRECISION BCND
      DOUBLE PRECISION CCND
      DOUBLE PRECISION FLOUT
      DOUBLE PRECISION U
      DOUBLE PRECISION V
      DOUBLE PRECISION W
      DOUBLE PRECISION P
      DOUBLE PRECISION VFRAC
      DOUBLE PRECISION DEN
      DOUBLE PRECISION VIS
      DOUBLE PRECISION TE
      DOUBLE PRECISION ED
      DOUBLE PRECISION RS
      DOUBLE PRECISION T
      DOUBLE PRECISION H
      DOUBLE PRECISION RF
      DOUBLE PRECISION SCAL
      DOUBLE PRECISION XP
      DOUBLE PRECISION YP
      DOUBLE PRECISION ZP
      DOUBLE PRECISION VOL
      DOUBLE PRECISION AREA
      DOUBLE PRECISION VPOR
      DOUBLE PRECISION ARPOR
      DOUBLE PRECISION WFACT
      DOUBLE PRECISION WORK
      DOUBLE PRECISION SMALL
      DOUBLE PRECISION SORMAX
      DOUBLE PRECISION TIME
      DOUBLE PRECISION DT
      DOUBLE PRECISION DTINVF
      DOUBLE PRECISION TPARM
      LOGICAL LDEN,LVIS,LTURB,LTEMP,LBUOY,LSCAL,LCOMP,LRECT,LCYN,LAXIS,
     +       LPOROS,LTRANS
C
      CHARACTER*(*) CWORK
C
C+++++++++++++++++ USER AREA 1 ++++++++++++++++++++++++++++++++++++++++++
C---- AREA FOR USERS EXPLICITLY DECLARED VARIABLES
      DOUBLE PRECISION CON
      DOUBLE PRECISION D1
      DOUBLE PRECISION D2
      DOUBLE PRECISION D3
      DOUBLE PRECISION D4
      DOUBLE PRECISION C1
      DOUBLE PRECISION C2
      DOUBLE PRECISION G0
      DOUBLE PRECISION hx
      DOUBLE PRECISION hy
      DOUBLE PRECISION hz

      DOUBLE PRECISION S1
      DOUBLE PRECISION S2
      DOUBLE PRECISION S3

C
C+++++++++++++++++ END OF USER AREA 1 +++++++++++++++++++++++++++++++++++
C
      COMMON /ALL/NBLOCK,NCELL,NBDRY,NNODE,NFACE,NVERT,NDIM,
     +      /ALLWRK/NRWS,NIWS,NCWS,IWRFRE,IWIFRE,IWCFRE,/ADDIMS/NPHASE,
     +      NSCAL,NVAR,NPROP,NDVAR,NDPROP,NDXNN,NDGEOM,NDCOEF,NILIST,
     +      NRLIST,NTOPOL,/BCSOUT/IFLOUT,/CHKUSR/IVERS,IUCALL,IUSED,
     +      /DEVICE/NREAD,NWRITE,NRDISK,NWDISK,/IDUM/ILEN,JLEN,
     +      /IMFBMP/IMFBMP,JMFBMP,/LOGIC/LDEN,LVIS,LTURB,LTEMP,LBUOY,
     +      LSCAL,LCOMP,LRECT,LCYN,LAXIS,LPOROS,LTRANS,/MLTGRD/MLEVEL,
     +      NLEVEL,ILEVEL,/SGLDBL/IFLGPR,ICHKPR,/SPARM/SMALL,SORMAX,
     +      NITER,INDPRI,MAXIT,NODREF,NODMON,/TRANSI/NSTEP,KSTEP,MF,
     +      INCORE,/TRANSR/TIME,DT,DTINVF,TPARM,/UBCSFL/IUBCSF
C
C+++++++++++++++++ USER AREA 2 ++++++++++++++++++++++++++++++++++++++++++
C---- AREA FOR USERS TO DECLARE THEIR OWN COMMON BLOCKS
C     THESE SHOULD START WITH THE CHARACTERS 'UC' TO ENSURE
C     NO CONFLICT WITH NON-USER COMMON BLOCKS
C
C+++++++++++++++++ END OF USER AREA 2 +++++++++++++++++++++++++++++++++++
C
      DIMENSION VARBCS(NVAR,NPHASE,NCELL+1:NNODE),VARAMB(NVAR,NPHASE),
     +         A(4+NSCAL,NPHASE,NSTART:*),B(4+NSCAL,NPHASE,NSTART:*),
     +         C(4+NSCAL,NPHASE,NSTART:*),FLOUT(*),ACND(NCST:*),
     +         BCND(NCST:*),CCND(NCST:*),IWGVEL(NDVWAL,NPHASE)
C
      DIMENSION U(NNODE,NPHASE),V(NNODE,NPHASE),W(NNODE,NPHASE),
     +         P(NNODE,NPHASE),VFRAC(NNODE,NPHASE),DEN(NNODE,NPHASE),
     +         VIS(NNODE,NPHASE),TE(NNODE,NPHASE),ED(NNODE,NPHASE),
     +         RS(NNODE,NPHASE,6),T(NNODE,NPHASE),H(NNODE,NPHASE),
     +         RF(NNODE,NPHASE,4),SCAL(NNODE,NPHASE,NSCAL)
C
      DIMENSION XP(NNODE),YP(NNODE),ZP(NNODE),VOL(NCELL),AREA(NFACE,3),
     +         VPOR(NCELL),ARPOR(NFACE,3),WFACT(NFACE),IPT(*),
     +         IBLK(5,NBLOCK),IPVERT(NCELL,8),IPNODN(NCELL,6),
     +         IPFACN(NCELL,6),IPNODF(NFACE,4),IPNODB(NBDRY,4),
     +         IPFACB(NBDRY),IWORK(*),WORK(*),CWORK(*)
C
```

```
C++++++++++++++++ USER AREA 3 ++++++++++++++++++++++++++++++++++++++
C---- AREA FOR USERS TO DIMENSION THEIR ARRAYS
C
C---- AREA FOR USERS TO DEFINE DATA STATEMENTS
C
C++++++++++++++++ END OF USER AREA 3 ++++++++++++++++++++++++++++++++
C
C---- STATEMENT FUNCTION FOR ADDRESSING
      IP(I,J,K) = IPT((K-1)*ILEN*JLEN+ (J-1)*ILEN+I)
C
C----VERSION NUMBER OF USER ROUTINE AND PRECISION FLAG
C
      IVERS = 5
      ICHKPR = 2
C
C++++++++++++++++ USER AREA 4 ++++++++++++++++++++++++++++++++++++++
C---- TO USE THIS USER ROUTINE FIRST SET IUSED=1
C     AND SET IUBCSF FLAG:
C     BOUNDARY CONDITIONS NOT CHANGING                        IUBCSF=0
C     BOUNDARY CONDITIONS CHANGING WITH ITERATION             IUBCSF=1
C     BOUNDARY CONDITIONS CHANGING WITH TIME                  IUBCSF=2
C     BOUNDARY CONDITIONS CHANGING WITH TIME AND ITERATION    IUBCSF=3
C
      IUSED = 1
      IUBCSF = 1
C
C++++++++++++++++ END OF USER AREA 4 ++++++++++++++++++++++++++++++++
C
      IF (IUSED.EQ.0) RETURN
C
C---- FRONTEND CHECKING OF USER ROUTINE
      IF (IUCALL.EQ.0) RETURN
C
C++++++++++++++++ USER AREA 5 ++++++++++++++++++++++++++++++++++++++
C
C---- AREA FOR SETTING VALUES AT INLETS, PRESSURE BOUNDARIES
C     AND OUTLETS. (NOTE THAT THE MASS FLOW AT OUTLETS IS
C     SPECIFIED IN USER AREA 7)
C
C     IF USING A REYNOLDS STRESS OR FLUX MODEL, NOTE THAT AT INLETS
C     IT IS IMPORTANT THAT THE USER SETS ALL COMPONENTS OF THE
C     REYNOLDS STRESS AND FLUX AND THE TURBULENT KINETIC ENERGY
C     AS WELL AS THE ENERGY DISSIPATION RATE.
C
C     SET THE VALUES IN VARBCS(NVAR,NPHASE,ILEN,JLEN,KLEN)
C
C---- EXAMPLE: SETTING A LINEAR T PROFILE ON INLET PATCH 'ENTRANCE'
C             LEAVE OTHER VARIABLES AS SET IN COMMAND LANGUAGE
C
C--  INTERROGATE GETVAR FOR VARIABLE NUMBERS.
C
C     CALL GETVAR('USRBCS','T     ',IT)
C
C  SET IPHS = 1 FOR SINGLE PHASE FLOW.
C
C     IPHS = 1
C
C  USE IPREC TO FIND ADDRESSES
C
C     CALL IPREC('ENTRANCE','PATCH','CENTRES',IPT,ILEN,JLEN,KLEN,
C    +          CWORK,IWORK)
C
C     XMAX=2.0
C     XMIN=1.0
C     TMAX=300.0
C     TMIN=250.0
C  LOOP OVER PATCH
C     DO 103 K = 1, KLEN
C       DO 102 J = 1, JLEN
C         DO 101 I = 1, ILEN
C  USE STATEMENT FUNCTION IP TO GET ADDRESSES
C         INODE = IP(I,J,K)
C  SET VARBCS
C         F=(XP(INODE)-XMIN)/(XMAX-XMIN)
C         VARBCS(IT,IPHS,INODE) = F*TMAX + (1.0-F)*TMIN
C 101       CONTINUE
C 102     CONTINUE
C 103   CONTINUE
C
C----END OF EXAMPLE
C
C++++++++++++++++ END OF USER AREA 5 ++++++++++++++++++++++++++++++++++

C
C++++++++++++++++ USER AREA 6 +++++++++++++++++++++++++++++++++++++++++
C
C---- AREA FOR SETTING VALUES AT WALLS
C
C     USE A(4+NSCAL,NPHASE,NNODE)
C     WHERE NSCAL = NO. OF SCALARS, AND NPHASE = NO. OF PHASES.
C
C     THE CONVENTION FOR VARIABLE NUMBERS IS DIFFERENT IN THIS ROUTINE
C     FROM THAT IN THE REST OF THE PROGRAM. IT IS:
C
C     IU = 1, IV = 2 , IW = 3, IT = 4, IS = 5
C
C---- EXAMPLE: SETTING FREE SLIP BOUNDARY CONDITIONS AT ALL WALLS
C           AND SETTING T=300.0 AND SCALAR1 AND SCALAR2 =0.0
C           ON WALL1. SET T=400.0 ON CONDUCTING SOLID BOUNDARY WALL2
C
C-- SET POINTERS
C
      IU = 1
      IV = 2
      IW = 3
      IT = 4
      IS = 5
C
C-- SET IPHS = 1 FOR SINGLE PHASE FLOW.
C
      IPHS = 1
C
C  USE IPALL TO FIND 1D ADDRESSES OF A GROUP OF PATCH CENTRES
C
C     CALL IPALL('*','WALL','PATCH','CENTRES',IPT,NPT,CWORK,IWORK)
C
C  LOOP OVER GROUP OF PATCHES
C     DO 200 I=1,NPT
C  USE ARRAY IPT TO GET ADDRESS
C       INODE=IPT(I)
```

```
C      A(IU,IPHS,INODE) = 0.0
C      B(IU,IPHS,INODE) = 1.0
C      C(IU,IPHS,INODE) = 0.0
C
C      A(IV,IPHS,INODE) = 0.0
C      B(IV,IPHS,INODE) = 1.0
C      C(IV,IPHS,INODE) = 0.0
C
C      A(IW,IPHS,INODE) = 0.0
C      B(IW,IPHS,INODE) = 1.0
C      C(IW,IPHS,INODE) = 0.0
C 200   CONTINUE
C
C  USE IPREC TO FIND ADDRESSES OF SINGLE PATCH
C
C     CALL IPREC('WALL1','PATCH','CENTRES',IPT,ILEN,JLEN,KLEN,
C   +         CWORK,IWORK)
C  LOOP OVER PATCH
C       DO 203 K = 1, KLEN
C         DO 202 J = 1, JLEN
C           DO 201 I = 1, ILEN
C  USE STATEMENT FUNCTION IP TO GET ADDRESSES
C           INODE = IP(I,J,K)
C
C           A(IT,IPHS,INODE) = 1.0
C           B(IT,IPHS,INODE) = 0.0
C           C(IT,IPHS,INODE) = 300.0
C
C           A(IS,IPHS,INODE) = 1.0
C           B(IS,IPHS,INODE) = 0.0
C           C(IS,IPHS,INODE) = 0.0
C
C           A(IS+1,IPHS,INODE) = 1.0
C           B(IS+1,IPHS,INODE) = 0.0
C           C(IS+1,IPHS,INODE) = 0.0
C
C 201         CONTINUE
C 202       CONTINUE
C 203    CONTINUE
C
C  USE IPALL TO FIND 1D ADDRESSES OF A GROUP OF PATCH CENTRES
C
C     CALL IPALL('WALL2','*','PATCH','CENTRES',IPT,NPT,CWORK,IWORK)
C
C  LOOP OVER GROUP OF PATCHES
C     DO 300 I=1,NPT
C  USE ARRAY IPT TO GET ADDRESS
C      INODE=IPT(I)
C      ACND(INODE) = 1.0
C      BCND(INODE) = 0.0
C      CCND(INODE) = 400.0
C 300   CONTINUE
C
C----END OF EXAMPLE
C
C+++++++++++++++++ END OF USER AREA 6 +++++++++++++++++++++++++++++++++++
C
C
C+++++++++++++++++ USER AREA 7 +++++++++++++++++++++++++++++++++++++++++++
C***********************************************************************
```

```
C             THIN WALL BOUNDARY CONDITION
C                             USING Eq 2.16 n
C          AND HARTMANN WALL BC FOR VELOCITIES
C    C2 = conductance ratio of the cooling plates
C   CON = Conductance ratio of the other walls
C    CHa = conductance ratio of the Hartmann layer = 1/Ha
C    AL = under relaxation factor for the boundary condition
C    hx, hy,hz are the maximum extension of the cavity
C*******************************************************************************
        CON =0.065E+00
        C2 = 0.026E+00
        CHa = 1.000E-04
        hx = 3.365D0
        hy = 1.0D0
        hz =  0.207D0
        AL= 2.0E-01


     CALL IPREC('WALLYA','PATCH','CENTRES',IPT,ILEN,JLEN,KLEN,
    +           CWORK,IWORK)
C  LOOP OVER PATCH
        DO 203 K = 1, KLEN
          DO 202 J = 1, JLEN
            DO 201 I = 1, ILEN
C  USE STATEMENT FUNCTION IP TO GET ADDRESSES
            INODE= IP(I,J,K)
              INODE1  = IP(I,J,K+1)
              INODE2 = IP(I,J,K-1)
              INODE3 = IP(I+1,J,K)
              INODE4 = IP(I-1,J,K)
              IBDRY = INODE - NCELL
              INDUM =IPNODB(IBDRY,1)

            D1=DABS(ZP(INODE1)-ZP(INODE))
            D2=DABS(ZP(INODE2)-ZP(INODE))
            D3=DABS(XP(INODE3)-XP(INODE))
            D4=DABS(XP(INODE4)-XP(INODE))



            IF (K.eq.1) THEN

            INODE2 = IPNODN(INDUM,6)
            D2 = hz-DABS(ZP(INODE))+hy-DABS(YP(INODE2))


             ENDIF

            IF (K.eq.KLEN) THEN

            INODE1 = IPNODN(INDUM,3)
            D1 = hy-DABS(YP(INODE1))+hz-DABS(ZP(INODE))
             ENDIF

            IF(I.eq.1) THEN

            INODE4 = IPNODN(INDUM,4)
            D4 =hy-DABS(YP(INODE4))+ hx-DABS(XP(INODE))

             ENDIF
        IF (I.eq.ILEN) THEN

        INODE3 = IPNODN(INDUM,1)
        D3 =hy-DABS(YP(INODE3))+hx-DABS(XP(INODE))
         ENDIF

       S1=(SCAL(INODE1,1,1)/D1+SCAL(INODE2,1,1)/D2)
       S1=S1/(D1+D2)
       S2=(SCAL(INODE3,1,1)/D3+SCAL(INODE4,1,1)/D4)
       S2=S2/(D3+D4)


        A(IS,IPHS,INODE) = (1/(D1*D2)+1/(D3*D4))
         B(IS,IPHS,INODE) =0.5D0/(CHa+CON)
     C(IS,IPHS,INODE) = (1-AL)*C(IS,IPHS,INODE)+AL*(S1+S2)


        DY= DABS(YP(INODE)-YP(INDUM))
        A(IU,IPHS,INODE) = 1.0D0
        B(IU,IPHS,INODE) = -DY
        C(IU,IPHS,INODE) = U(INDUM,1)

        A(IV,IPHS,INODE) = 1.0D0
        B(IV,IPHS,INODE) = 0.0D0
        C(IV,IPHS,INODE) = 0.0D0

        A(IW,IPHS,INODE) =  1.0D0
        B(IW,IPHS,INODE) = -DY
        C(IW,IPHS,INODE) = W(INDUM,1)

 201           CONTINUE
 202        CONTINUE
 203    CONTINUE

    CALL IPREC('WALLYB','PATCH','CENTRES',IPT,ILEN,JLEN,KLEN,
   +          CWORK,IWORK)
C  LOOP OVER PATCH
      DO 303 K = 1, KLEN
        DO 302 J = 1, JLEN
          DO 301 I = 1, ILEN
C  USE STATEMENT FUNCTION IP TO GET ADDRESSES
         INODE= IP(I,J,K)
          INODE1  = IP(I,J,K+1)
          INODE2 = IP(I,J,K-1)
          INODE3 = IP(I+1,J,K)
          INODE4 = IP(I-1,J,K)
          IBDRY = INODE - NCELL
          INDUM =IPNODB(IBDRY,1)

        D1=DABS(ZP(INODE1)-ZP(INODE))
        D2=DABS(ZP(INODE2)-ZP(INODE))
        D3=DABS(XP(INODE3)-XP(INODE))
        D4=DABS(XP(INODE4)-XP(INODE))



        IF (K.eq.1) THEN

        INODE2 = IPNODN(INDUM,6)
        D2 = hz-DABS(ZP(INODE))+hy-DABS(YP(INODE2))


         ENDIF

        IF (K.eq.KLEN) THEN

        INODE1 = IPNODN(INDUM,3)
        D1 = hy-DABS(YP(INODE1))+hz-DABS(ZP(INODE))
         ENDIF

        IF(I.eq.1) THEN

        INODE4 = IPNODN(INDUM,4)
        D4 =hy-DABS(YP(INODE4))+ hx-DABS(XP(INODE))

         ENDIF

        IF (I.eq.ILEN) THEN

        INODE3 = IPNODN(INDUM,1)
        D3 =hy-DABS(YP(INODE3))+hx-DABS(XP(INODE))
         ENDIF

       S1=(SCAL(INODE1,1,1)/D1+SCAL(INODE2,1,1)/D2)
       S1=S1/(D1+D2)
       S2=(SCAL(INODE3,1,1)/D3+SCAL(INODE4,1,1)/D4)
       S2=S2/(D3+D4)


        A(IS,IPHS,INODE) = (1/(D1*D2)+1/(D3*D4))
        B(IS,IPHS,INODE) =0.5D0/(CHa+CON)
      C(IS,IPHS,INODE) = (1-AL)*C(IS,IPHS,INODE)+AL*(S1+S2)

         DY= DABS(YP(INODE)-YP(INDUM))
        A(IU,IPHS,INODE) = 1.0D0
        B(IU,IPHS,INODE) = -DY
        C(IU,IPHS,INODE) = U(INDUM,1)

        A(IV,IPHS,INODE) = 1.0D0
        B(IV,IPHS,INODE) = 0.0D0
        C(IV,IPHS,INODE) = 0.0D0

        A(IW,IPHS,INODE) =  1.0D0
        B(IW,IPHS,INODE) = -DY
        C(IW,IPHS,INODE) = W(INDUM,1)


 301         CONTINUE
 302        CONTINUE
 303    CONTINUE

   CALL IPREC('WALLA','PATCH','CENTRES',IPT,ILEN,JLEN,KLEN,
   +        CWORK,IWORK)
C  LOOP OVER PATCH
      DO 503 K = 1, KLEN
       DO 502 J = 1,JLEN
         DO 501 I = 1,ILEN
C  USE STATEMENT FUNCTION IP TO GET ADDRESSES
         INODE= IP(I,J,K)
```

```
            INODE1   = IP(I,J,K+1)
            INODE2 = IP(I,J,K-1)
            INODE3 = IP(I,J+1,K)
            INODE4 = IP(I,J-1,K)
          D1=DABS(ZP(INODE1)-ZP(INODE))
          D2=DABS(ZP(INODE2)-ZP(INODE))
          D3=DABS(YP(INODE3)-YP(INODE))
          D4=DABS(YP(INODE4)-YP(INODE))



            IF (K.eq.1) THEN
          IBDRY = INODE - NCELL
          INDUM = IPNODB(IBDRY,1)
          INODE2 = IPNODN(INDUM,6)
          DW =hx-DABS(XP(INODE2))+hz-DABS(ZP(INODE))
           ENDIF

          IF (K.eq.KLEN) THEN
          IBDRY = INODE - NCELL
          INDUM = IPNODB(IBDRY,1)
          INODE1 = IPNODN(INDUM,3)
          D1 =hx-DABS(XP(INODE1))+hz-DABS(ZP(INODE))
           ENDIF

          IF(J.eq.1) THEN
           IBDRY = INODE - NCELL
          INDUM = IPNODB(IBDRY,1)
          INODE4 = IPNODN(INDUM,5)
          D4 =hx-DABS(XP(INODE4))+hy-DABS(YP(INODE))

           ENDIF

          IF (J.eq.JLEN) THEN
           IBDRY = INODE - NCELL
          INDUM = IPNODB(IBDRY,1)
          INODE3 = IPNODN(INDUM,2)
          D3 =hx-DABS(XP(INODE3))+hy-DABS(YP(INODE))

           ENDIF

      S1=(SCAL(INODE1,1,1)/D1+SCAL(INODE2,1,1)/D2)
      S1=S1/(D1+D2)
      S2=(SCAL(INODE3,1,1)/D3+SCAL(INODE4,1,1)/D4)
      S2=S2/(D3+D4)


         A(IS,IPHS,INODE) = (1/(D1*D2)+1/(D3*D4))
          B(IS,IPHS,INODE) =0.5D0/CON
     C(IS,IPHS,INODE) = (1-AL)*C(IS,IPHS,INODE)+AL*(S1+S2)




501          CONTINUE
502       CONTINUE
503    CONTINUE
```

```
      CALL IPREC('WALLB','PATCH','CENTRES',IPT,ILEN,JLEN,KLEN,
     +          CWORK,IWORK)
C  LOOP OVER PATCH
        DO 403 K = 1, KLEN
          DO 402 J = 1,JLEN
            DO 401 I = 1,ILEN
C  USE STATEMENT FUNCTION IP TO GET ADDRESSES
              INODE= IP(I,J,K)
             INODE1   = IP(I,J,K+1)
             INODE2 = IP(I,J,K-1)
             INODE3 = IP(I,J+1,K)
             INODE4 = IP(I,J-1,K)
           D1=DABS(ZP(INODE1)-ZP(INODE))
           D2=DABS(ZP(INODE2)-ZP(INODE))
           D3=DABS(YP(INODE3)-YP(INODE))
           D4=DABS(YP(INODE4)-YP(INODE))



             IF (K.eq.1) THEN
           IBDRY = INODE - NCELL
           INDUM = IPNODB(IBDRY,1)
           INODE2 = IPNODN(INDUM,6)
           DW =hx-DABS(XP(INODE2))+hz-DABS(ZP(INODE))
            ENDIF

           IF (K.eq.KLEN) THEN
           IBDRY = INODE - NCELL
           INDUM = IPNODB(IBDRY,1)
           INODE1 = IPNODN(INDUM,3)
           D1 =hx-DABS(XP(INODE1))+hz-DABS(ZP(INODE))
            ENDIF

           IF(J.eq.1) THEN
            IBDRY = INODE - NCELL
           INDUM = IPNODB(IBDRY,1)
           INODE4 = IPNODN(INDUM,5)
           D4 =hx-DABS(XP(INODE4))+hy-DABS(YP(INODE))

            ENDIF

           IF (J.eq.JLEN) THEN
            IBDRY = INODE - NCELL
           INDUM = IPNODB(IBDRY,1)
           INODE3 = IPNODN(INDUM,2)
           D3 =hx-DABS(XP(INODE3))+hy-DABS(YP(INODE))

            ENDIF

        S1=(SCAL(INODE1,1,1)/D1+SCAL(INODE2,1,1)/D2)
        S1=S1/(D1+D2)
        S2=(SCAL(INODE3,1,1)/D3+SCAL(INODE4,1,1)/D4)
        S2=S2/(D3+D4)


          A(IS,IPHS,INODE) = (1/(D1*D2)+1/(D3*D4))
           B(IS,IPHS,INODE) =0.5D0/CON
       C(IS,IPHS,INODE) = (1-AL)*C(IS,IPHS,INODE)+AL*(S1+S2)
```

```
  401          CONTINUE
  402       CONTINUE
  403    CONTINUE


      CALL IPREC('WALLZA','PATCH','CENTRES',IPT,ILEN,JLEN,KLEN,
     +           CWORK,IWORK)
C  LOOP OVER PATCH
         DO 603 K = 1, KLEN
           DO 602 J = 1,JLEN
             DO 601 I = 1,ILEN
C  USE STATEMENT FUNCTION IP TO GET ADDRESSES
            INODE= IP(I,J,K)
              INODE1   = IP(I,J+1,K)
              INODE2 = IP(I,J-1,K)
              INODE3 = IP(I+1,J,K)
              INODE4 = IP(I-1,J,K)
            D1=DABS(YP(INODE1)-YP(INODE))
            D2=DABS(YP(INODE2)-YP(INODE))
            D3=DABS(XP(INODE3)-XP(INODE))
            D4=DABS(XP(INODE4)-XP(INODE))



             IF (J.eq.1) THEN

            IBDRY = INODE - NCELL
            INDUM = IPNODB(IBDRY,1)
            INODE2 = IPNODN(INDUM,5)
            D2 = hy-DABS(YP(INODE))+hz-DABS(ZP(INODE2))

             ENDIF

            IF (J.eq.JLEN) THEN
            IBDRY = INODE - NCELL
            INDUM = IPNODB(IBDRY,1)
            INODE1 = IPNODN(INDUM,2)
            D1 = hy-DABS(YP(INODE))+hz-DABS(ZP(INODE1))

             ENDIF

            IF(I.eq.1) THEN
             IBDRY = INODE - NCELL
            INDUM = IPNODB(IBDRY,1)
            INODE4 = IPNODN(INDUM,4)
            D4 = hx-DABS(XP(INODE))+hz-DABS(ZP(INODE4))

             ENDIF

            IF (I.eq.ILEN) THEN
             IBDRY = INODE - NCELL
            INDUM = IPNODB(IBDRY,1)
            INODE3 = IPNODN(INDUM,1)
            D3 =hx-DABS(XP(INODE))+hz-DABS(ZP(INODE3))

             ENDIF

        S1=(SCAL(INODE1,1,1)/D1+SCAL(INODE2,1,1)/D2)
        S1=S1/(D1+D2)
      S2=(SCAL(INODE3,1,1)/D3+SCAL(INODE4,1,1)/D4)
      S2=S2/(D3+D4)


        A(IS,IPHS,INODE) = (1/(D1*D2)+1/(D3*D4))
         B(IS,IPHS,INODE) =0.5D0/C2
      C(IS,IPHS,INODE) = (1-AL)*C(IS,IPHS,INODE)+AL*(S1+S2)



 601        CONTINUE
 602      CONTINUE
 603   CONTINUE

      CALL IPREC('WALLZB','PATCH','CENTRES',IPT,ILEN,JLEN,KLEN,
    +          CWORK,IWORK)
C  LOOP OVER PATCH
        DO 703 K = 1, KLEN
          DO 702 J = 1,JLEN
            DO 701 I = 1,ILEN
C  USE STATEMENT FUNCTION IP TO GET ADDRESSES
           INODE= IP(I,J,K)
             INODE1   = IP(I,J+1,K)
             INODE2 = IP(I,J-1,K)
             INODE3 = IP(I+1,J,K)
             INODE4 = IP(I-1,J,K)
           D1=DABS(YP(INODE1)-YP(INODE))
           D2=DABS(YP(INODE2)-YP(INODE))
           D3=DABS(XP(INODE3)-XP(INODE))
           D4=DABS(XP(INODE4)-XP(INODE))



            IF (J.eq.1) THEN

           IBDRY = INODE - NCELL
           INDUM = IPNODB(IBDRY,1)
           INODE2 = IPNODN(INDUM,5)
           D2 = hy-DABS(YP(INODE))+hz-DABS(ZP(INODE2))

            ENDIF

           IF (J.eq.JLEN) THEN
           IBDRY = INODE - NCELL
           INDUM = IPNODB(IBDRY,1)
           INODE1 = IPNODN(INDUM,2)
           D1 = hy-DABS(YP(INODE))+hz-DABS(ZP(INODE1))

            ENDIF

           IF(I.eq.1) THEN
            IBDRY = INODE - NCELL
           INDUM = IPNODB(IBDRY,1)
           INODE4 = IPNODN(INDUM,4)
           D4 = hx-DABS(XP(INODE))+hz-DABS(ZP(INODE4))

            ENDIF

           IF (I.eq.ILEN) THEN
            IBDRY = INODE - NCELL
```

```
              INDUM = IPNODB(IBDRY,1)
              INODE3 = IPNODN(INDUM,1)
              D3 =hx-DABS(XP(INODE))+hz-DABS(ZP(INODE3))

               ENDIF

          S1=(SCAL(INODE1,1,1)/D1+SCAL(INODE2,1,1)/D2)
          S1=S1/(D1+D2)
          S2=(SCAL(INODE3,1,1)/D3+SCAL(INODE4,1,1)/D4)
          S2=S2/(D3+D4)



             A(IS,IPHS,INODE) = (1/(D1*D2)+1/(D3*D4))
              B(IS,IPHS,INODE) =0.5D0/C2
         C(IS,IPHS,INODE) = (1-AL)*C(IS,IPHS,INODE)+AL*(S1+S2)


 701           CONTINUE
 702        CONTINUE
 703    CONTINUE





C----- DEFINE FLOW AT OUTLETS (MASS FLOW BOUNDARIES)
C      (TO TEMPERATURES AND SCALARS AT MASS FLOW BOUNDARIES USE
C       USER AREA 5)
C
C     SET PARAMETER IFLOUT:
C     IFLOUT = 1  ==> MASS FLOW SPECIFIED AT LABELLED OUTLETS.
C     IFLOUT = 2  ==> FRACTIONAL MASS FLOW SPECIFIED AT LABELLED OUTLETS
C     IFLOUT = 2
C
C     SET OUTLET FLOW RATES:
C     FLOUT(LABEL) = MASS FLOW OUT OF OUTLETS LABELLED LABEL (IFLOUT=1).
C     FLOUT(LABEL) = FRACTIONAL MASS FLOW OUT OF OUTLETS LABELLED LABEL
C                    (IFLOUT=2).
C     FOR MULTIPHASE FLOWS IT IS NECESSARY TO SET
C     EITHER
C                    FLOUT(LABEL) = TOTAL MASS FLOW
C                    IFLOUT = 1
C                    IMFBMP = 0
C     OR
C                    FLOUT(LABEL + (IPHASE-1)*NLABEL) FOR EACH PHASE
C                    IFLOUT = 1 OR 2
C                    IMFBMP = 1
C
C----   EXAMPLE:   EQUIDISTRIBUTION   OF   FRACTIONAL   MASS   FLOW   AMONGST
OUTLETS
C
C     IFLOUT=2
C     FRAC = 1.0 / MAX( 1.0, FLOAT(NLABEL) )
C     DO 300 ILABEL = 1, NLABEL
C        FLOUT(ILABEL) = FRAC
C300  CONTINUE
C
C----END OF EXAMPLE
C
```

```
C+++++++++++++++++ END OF USER AREA 7 ++++++++++++++++++++++++++++++++++
C
      RETURN
C

       END
```

7. FIGURES

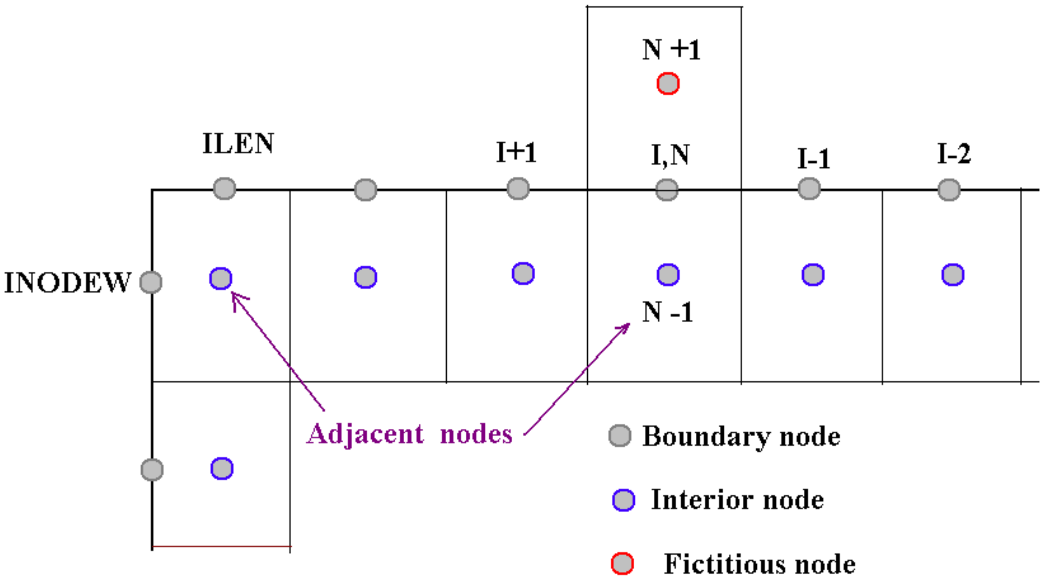


Figure 1: Mesh at a boundary of the domain, at the vicinity of an edge between two walls. Nodes I, I-1, I-2 and I+1 are used for the thin wall boundary condition at the node (I, N). The node N, N-1 and the fictitious node N+1 are used to set the Hartman-core boundary condition for the velocity at the node (I, N).

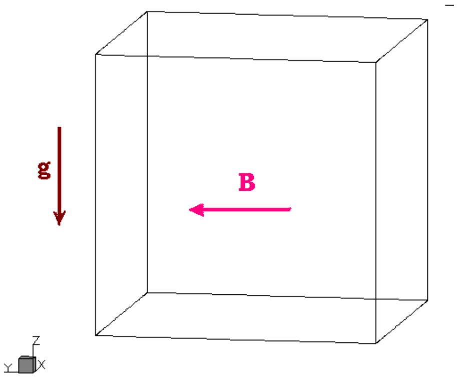


Figure 2: Validation case: Geometry of the fluid domain (Di Piazza (2002)). The Origin of the coordinate system is located in the center of the box

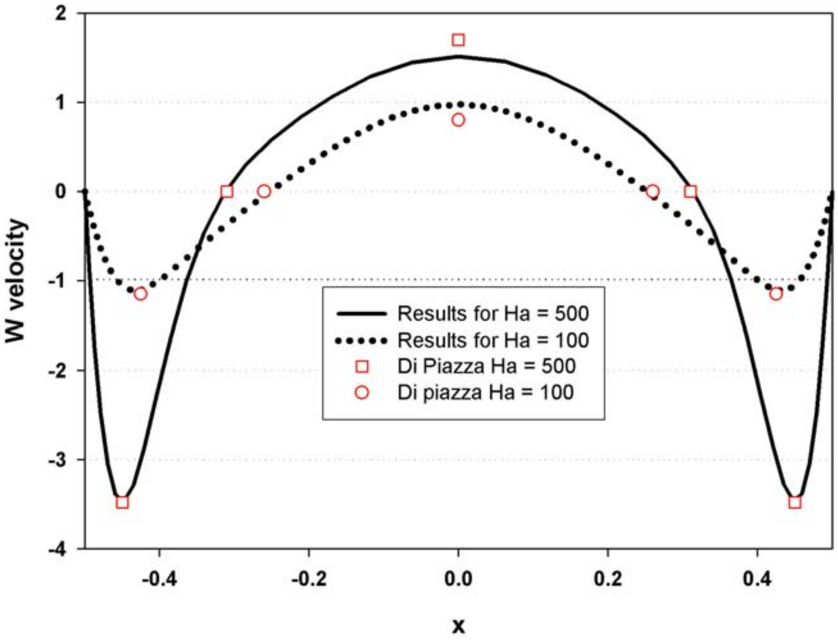


Figure 3: Dimensionless velocity profiles for *c* = 0.01, *Ra* = 10^5.

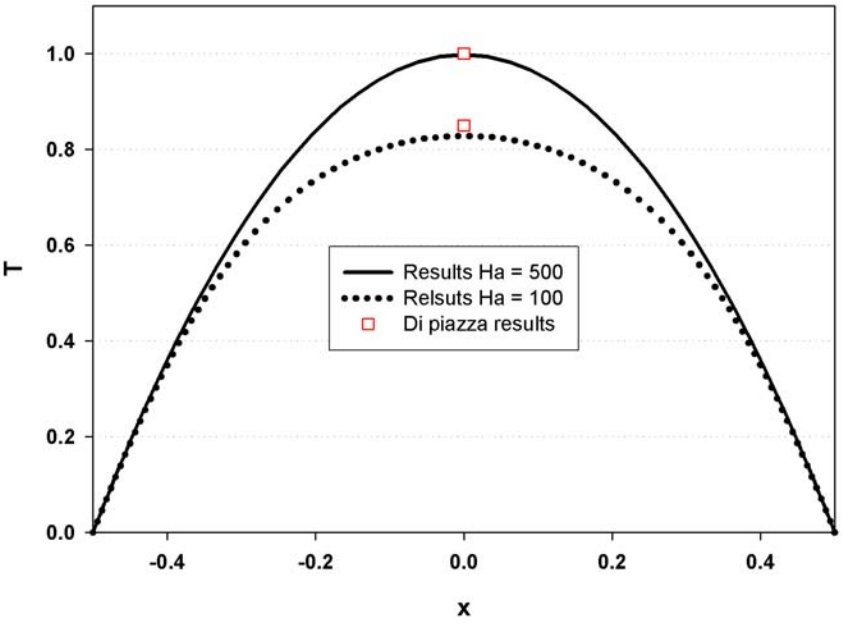
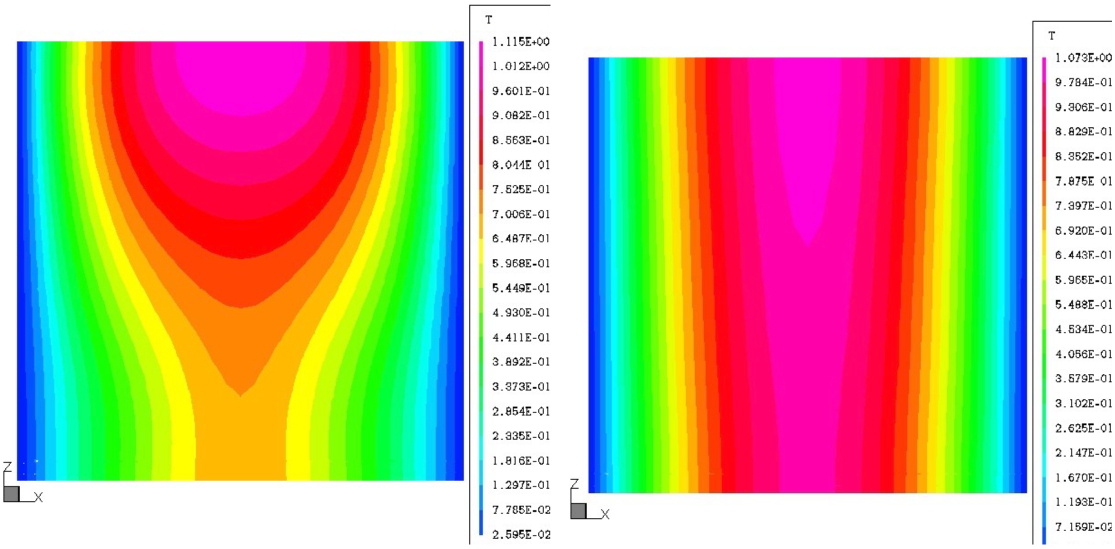


Figure 4: Dimensionless temperature profiles for *c* = 0.01, *Ra* = 10^5.



(a) **(b)**

Figure 5: Dimensionless temperature in the midplane y = 0 for c = 0.01, Ra = 10^5.

(a) Ha = 100, (b) Ha = 500

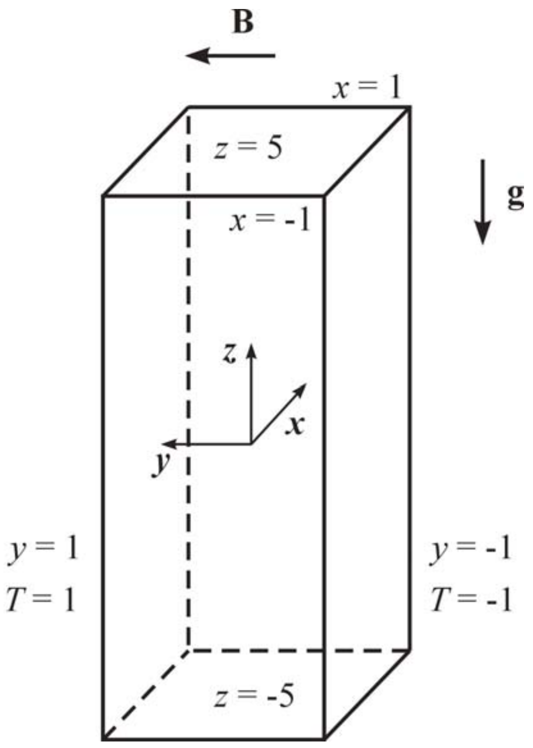


Figure 6: Flow in a vertical rectangular cavity, aspect ratio 1:1:5.

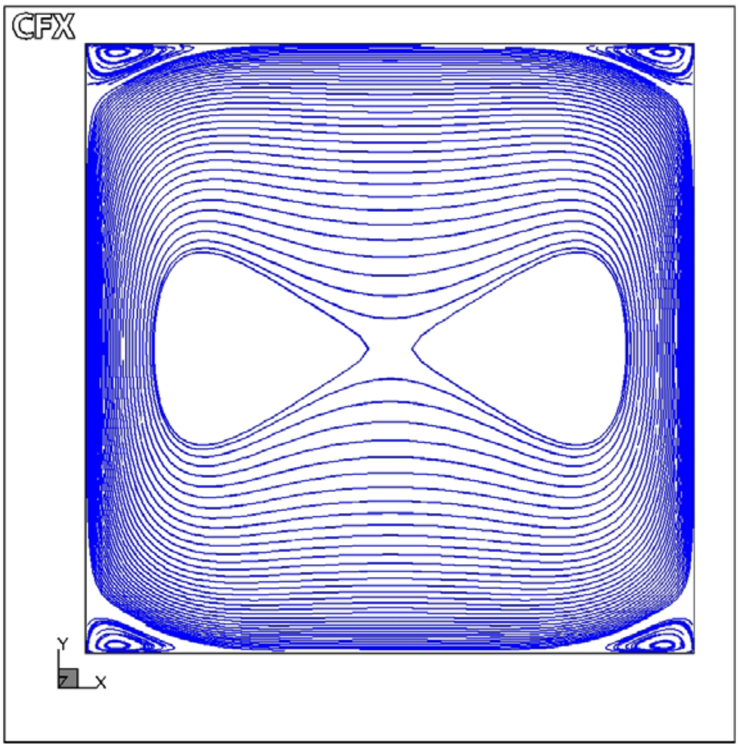


Figure 7: Electric current lines in the plane $z = 0$. $Ha = 100$, $Pe = 0$, $c = 0$, vertical cavity with aspect ratio 1:1:5.

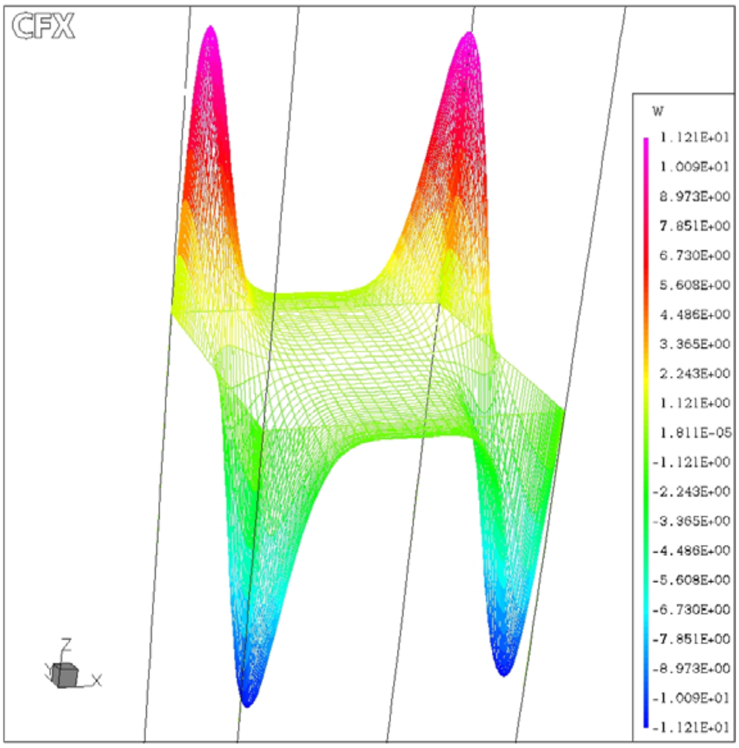


Figure 8: Vertical velocity in plane $z = 0$. $Ha = 100$, $Pe = 0$, $c = 0$, vertical cavity with aspect ratio 1:1:5.

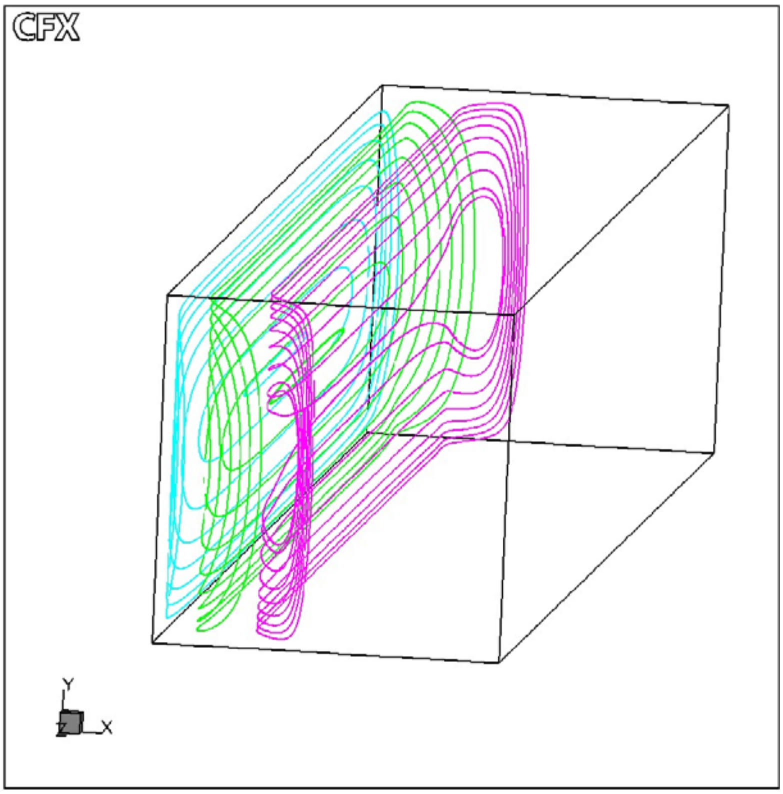


Figure 9: Streamlines. *Ha* = 100, *Pe* = 0, *c* = 0, vertical cavity with aspect ratio 1:1:5.

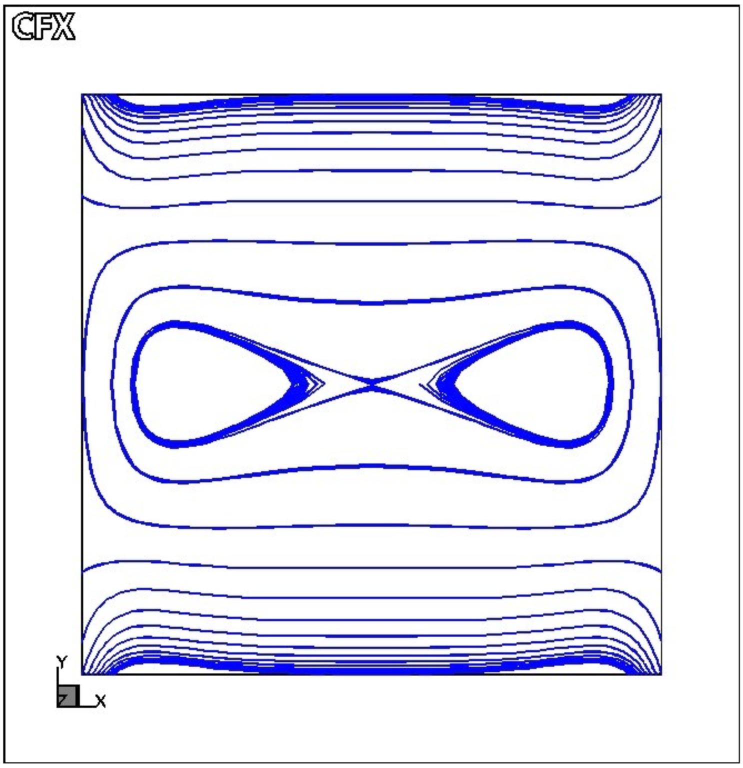


Figure 10: Electric current lines in the plane *z* = 0. *Ha* = 100, *Pe* = 1, *c* = 0.1, vertical cavity with aspect ratio 1:1:5.

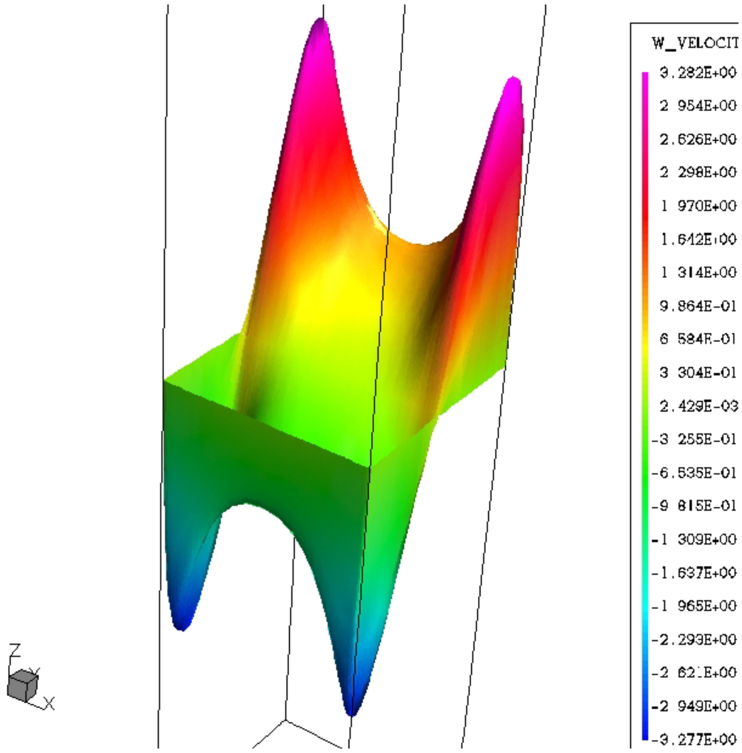


Figure 11: Vertical velocity in plane $z = 0$. $Ha = 100$, $Pe = 0$, $c = 0.1$, vertical cavity with aspect ratio 1:1:5.

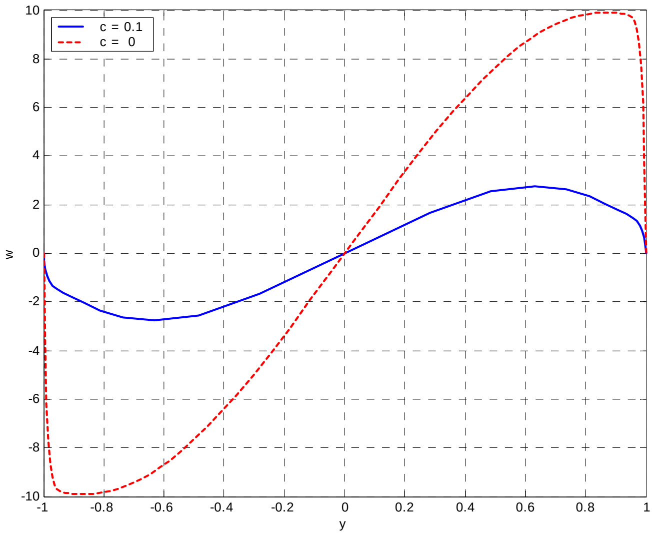
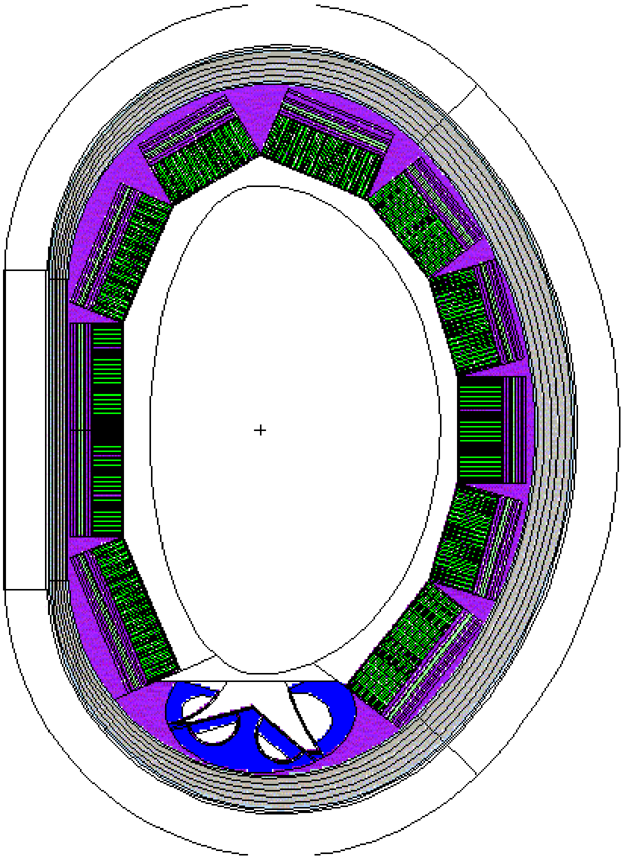


Figure 12: Vertical velocity at $z = 0$, $x = 0.85$ for $c = 0.1$ and $c = 0$. $Ha = 100$, $Pe = 0$, vertical cavity with aspect ratio 1:1:5**.**



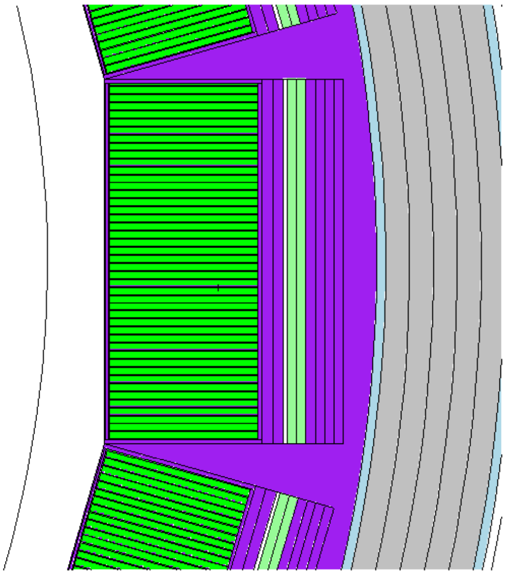


Figure 13 a: Vertical cut of PPCS reactor model B with HCLL blanket modules.

Figure 13b: Vertical cut through central outboard HCLL blanket module.

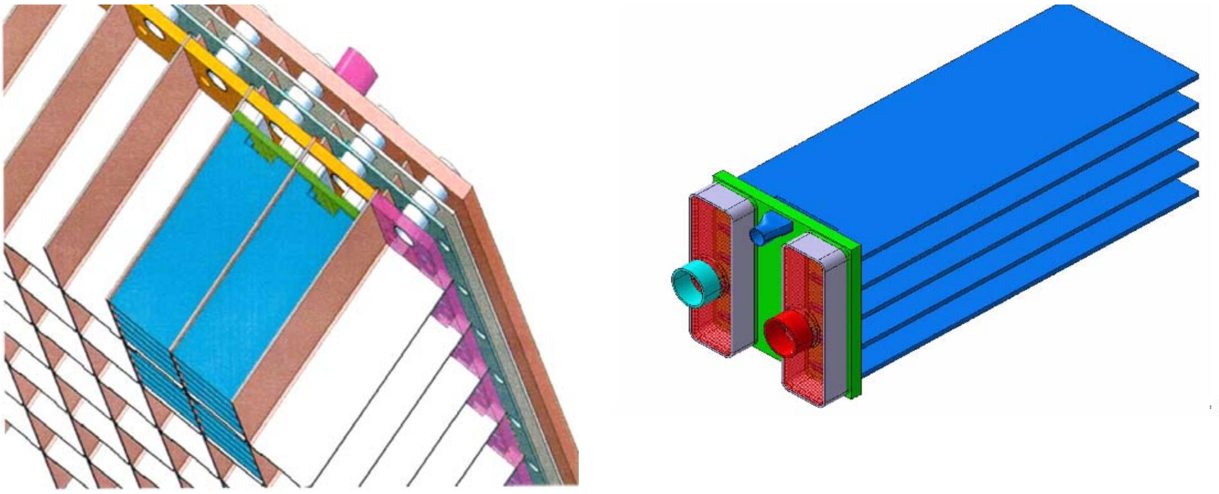


Figure 13c: Details of the HCLL blanket module and breeder unit

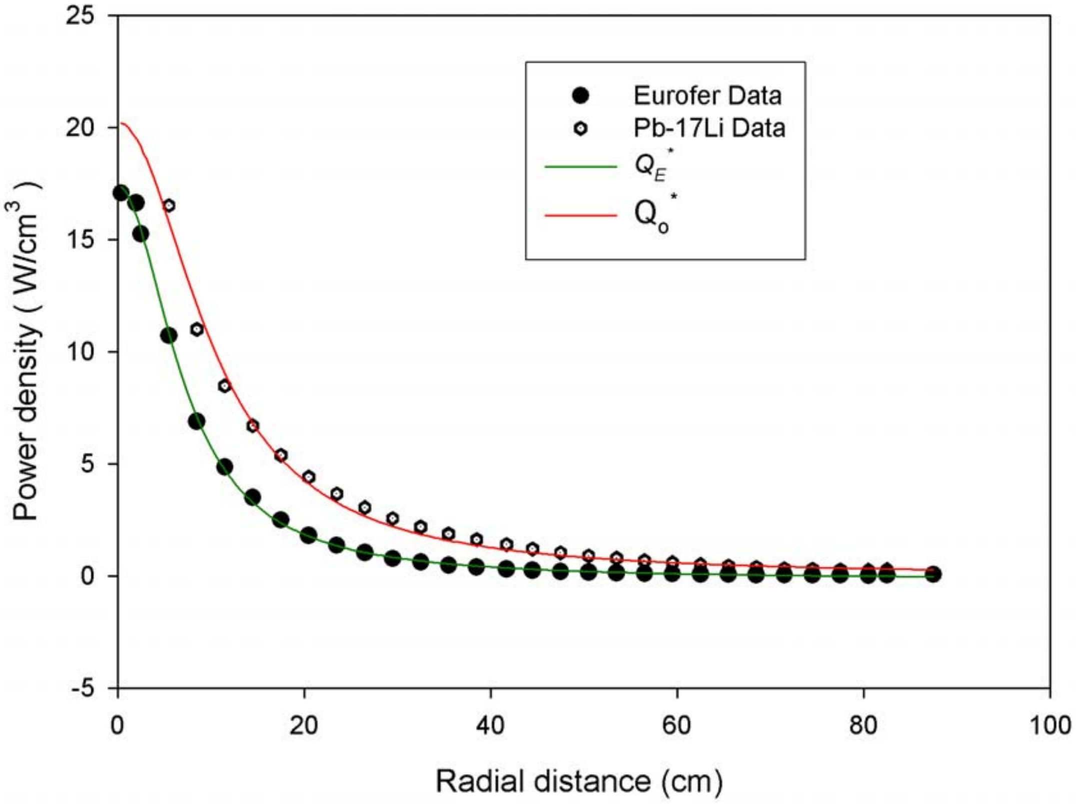


Figure 14 : Radial profile of power density in the central outboard blanket Module (averaged over a poloidal height of 50 cm). $Q_E(x)$ and $Q_O(x)$ are described by Eq 3.5 and Eq 3.6 , x^* is in cm.

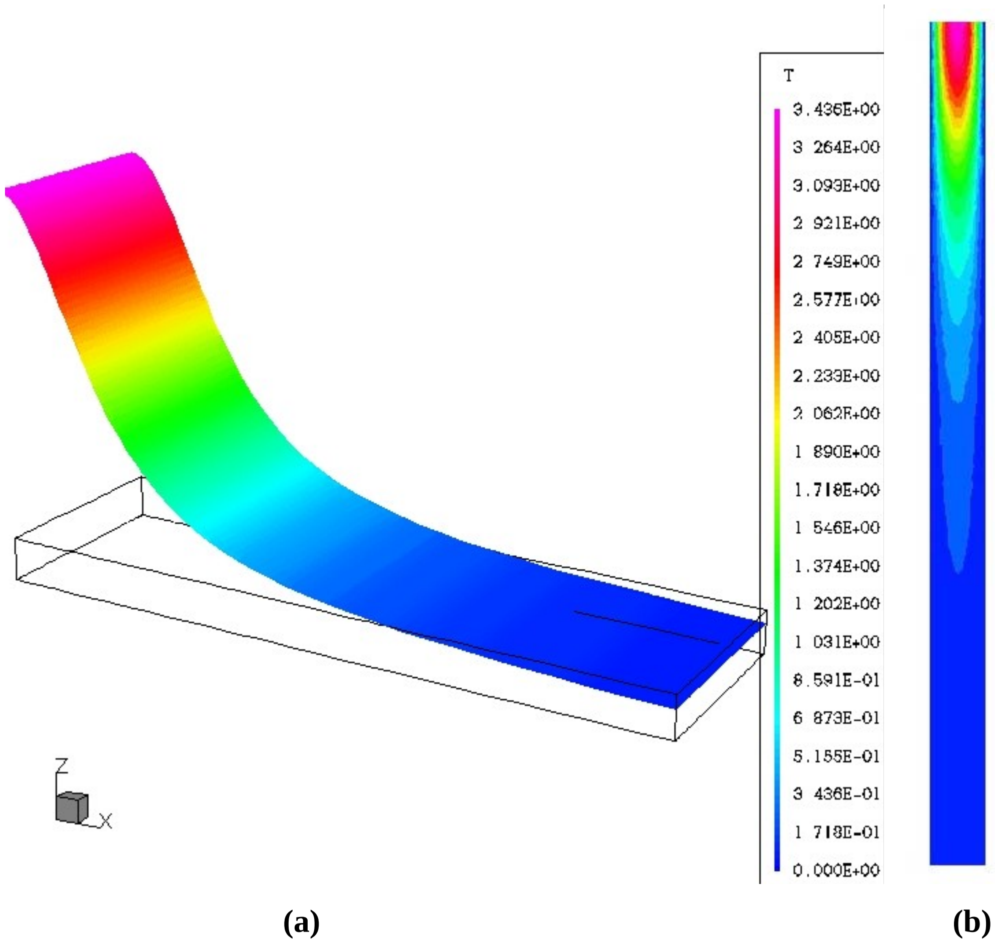


Figure 15: Dimensionless temperature distribution in the midplanes (a) z = 0, and (b) y = 0. (*Pe = 0* and heat source described by Eq 3.7)

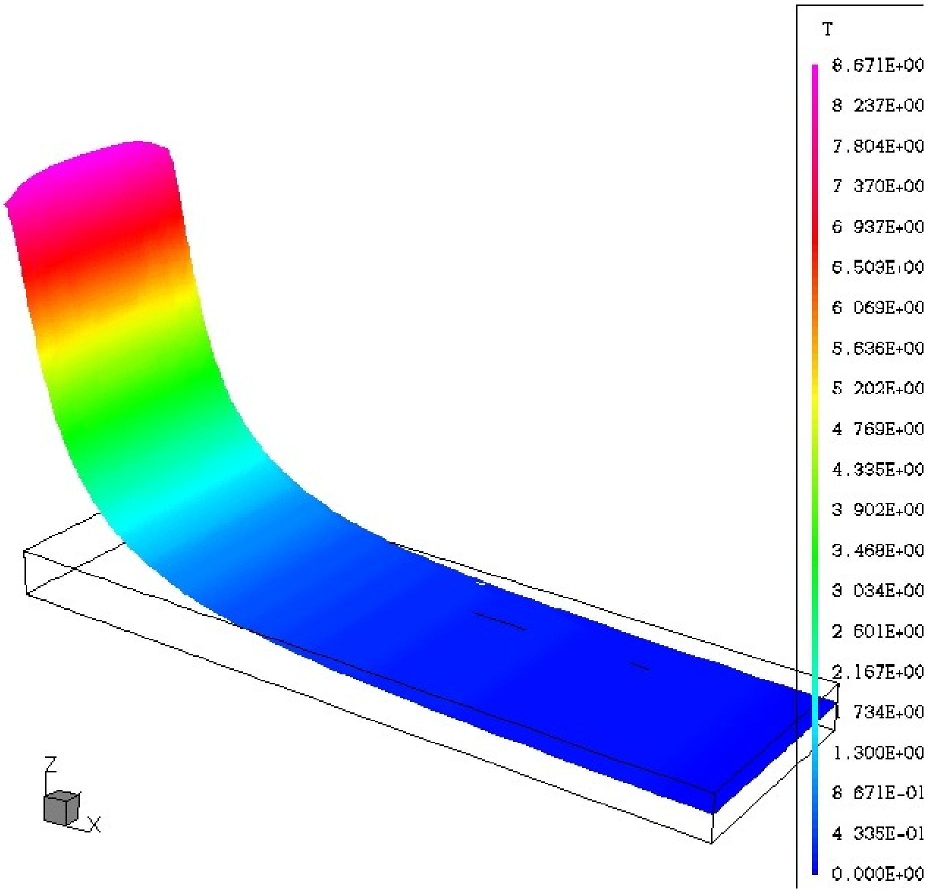


Figure 16: Example of distribution of the dimensionless temperature obtained with a false heat source distribution of Eq. 3.8. The temperature is plotted in the midplane $z = 0$

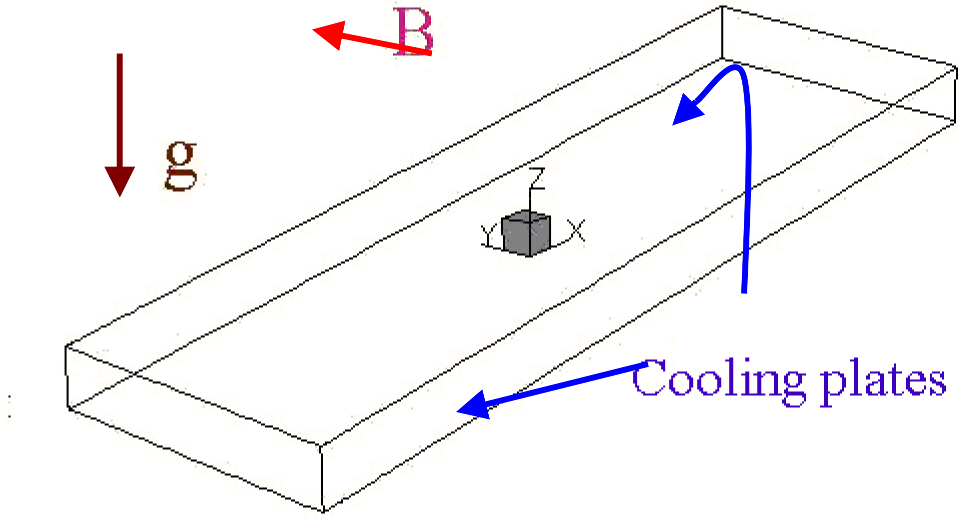


Figure 17: Geometry of the horizontal configuration.

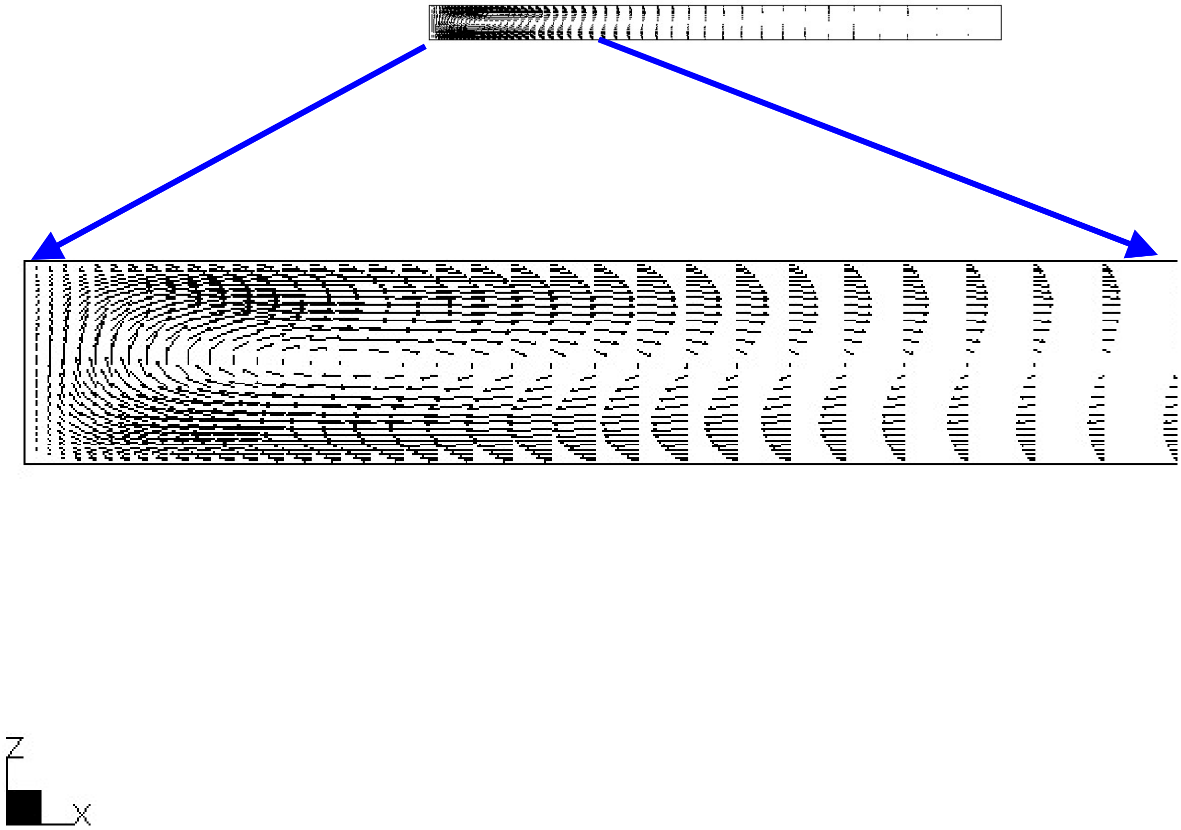


Figure 18: Velocity vector plot in the plane y = 0 for the horizontal cavity. (*Ha*=10000)

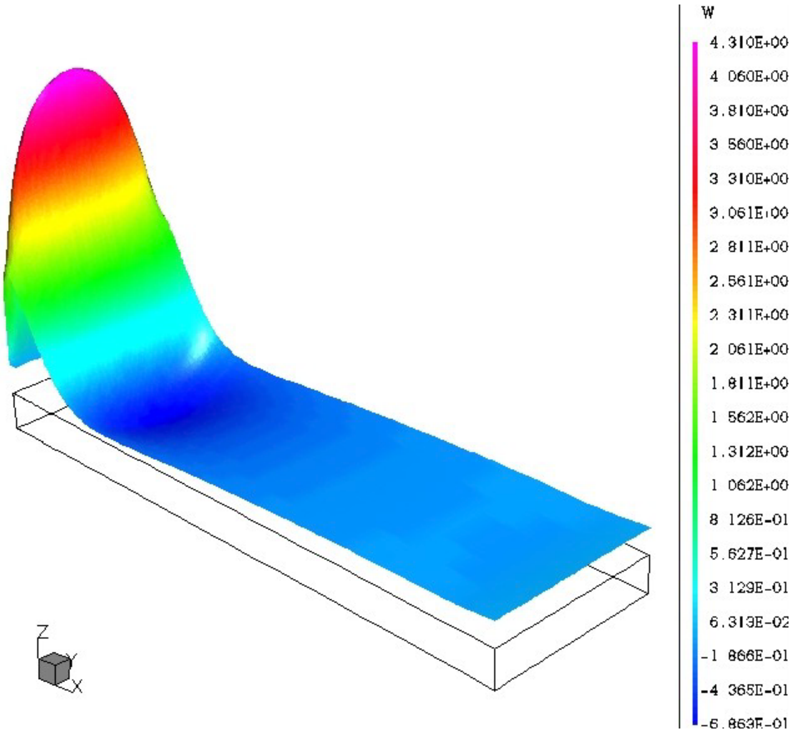


Figure 19: *W* velocity in the plane *y* = 0, *Ha* = 10000 (Horizontal cavity)

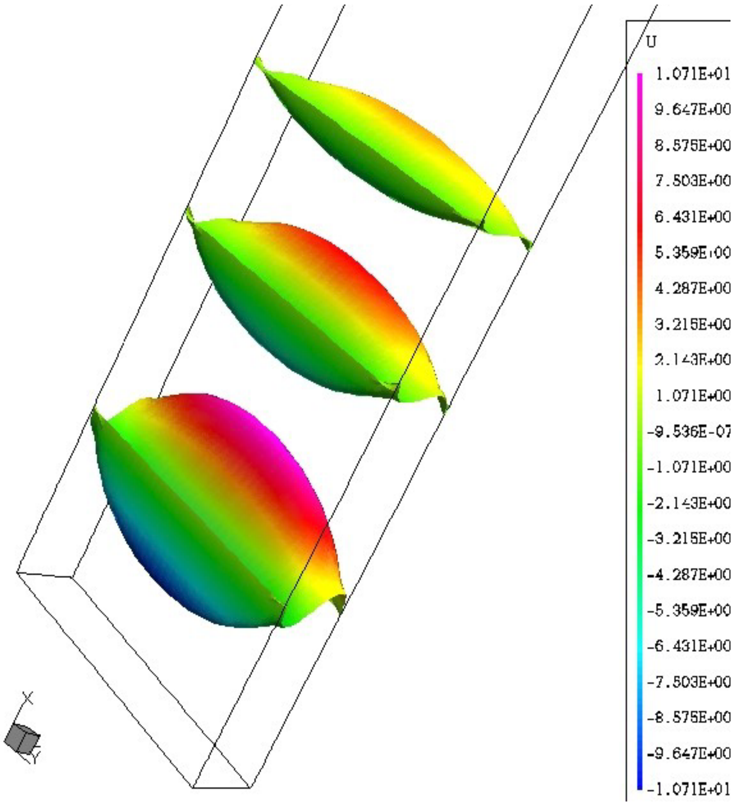
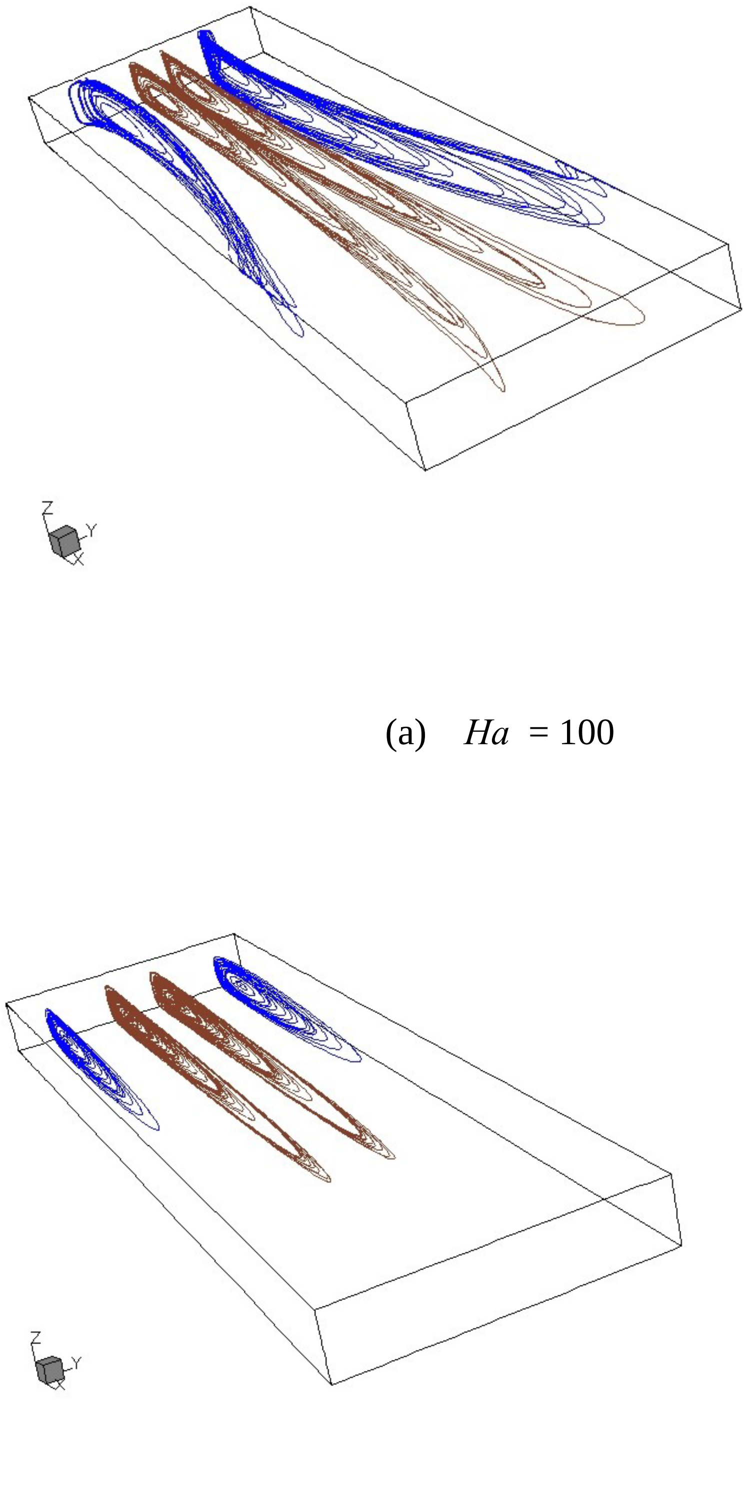


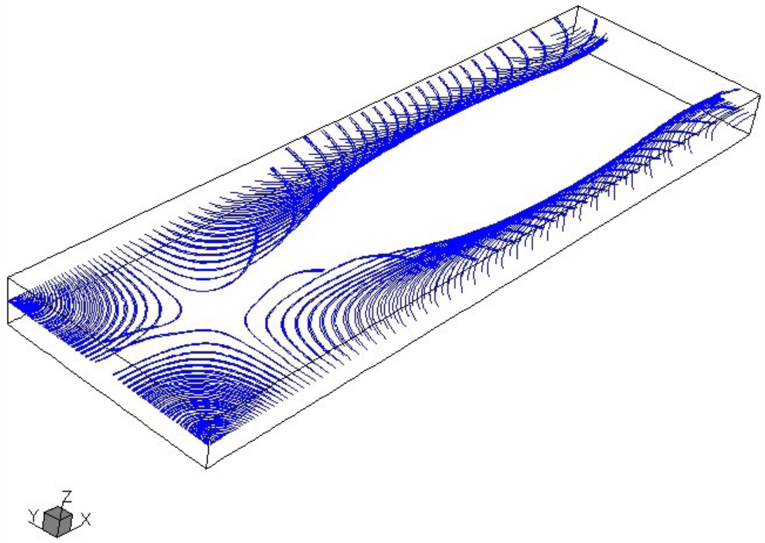
Figure 20: *U* velocity, *Ha* = 10000 (Horizontal cavity)



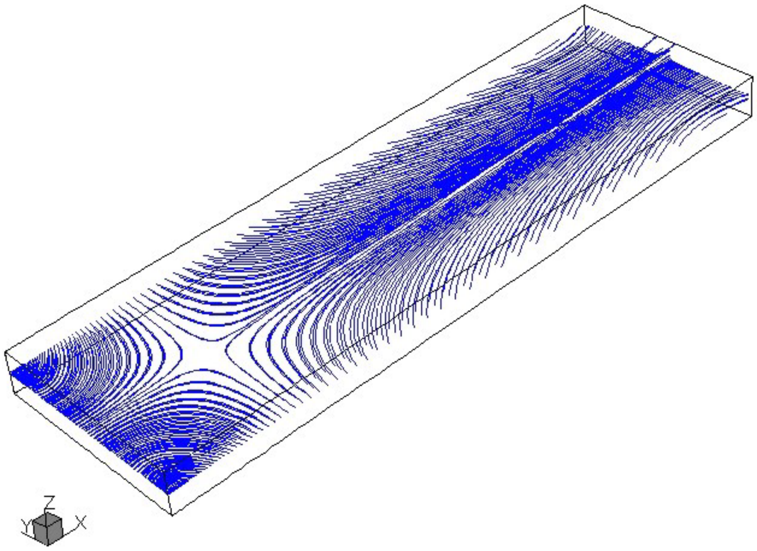
(a) *Ha* = 100

(b) *Ha* = 10000

Figure 21: Velocity streamlines in the horizontal cavity.

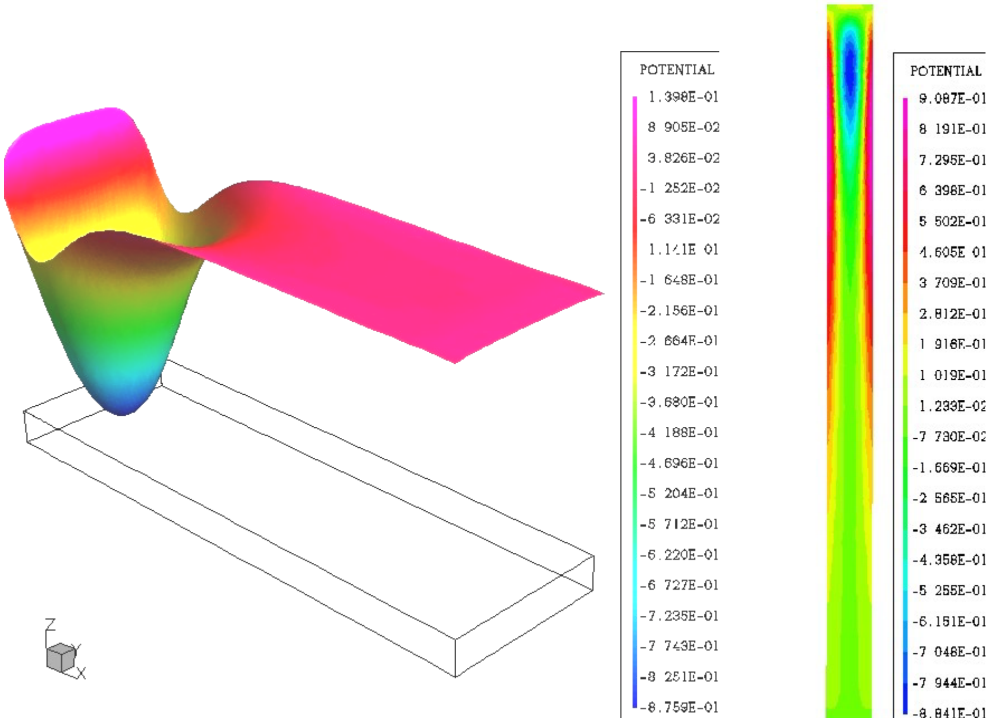


(a)



(b)

Figure 22: Electric current streamlines in the core for the horizontal cavity starting at (a) z = 0.01, (b) z = 0, for *Ha* =10000.



(a) **(b)**

Figure 23: Distribution of the potential in the midplanes z = 0 (a) and y = 0 (b) for Ha = 10000. (Horizontal cavity)

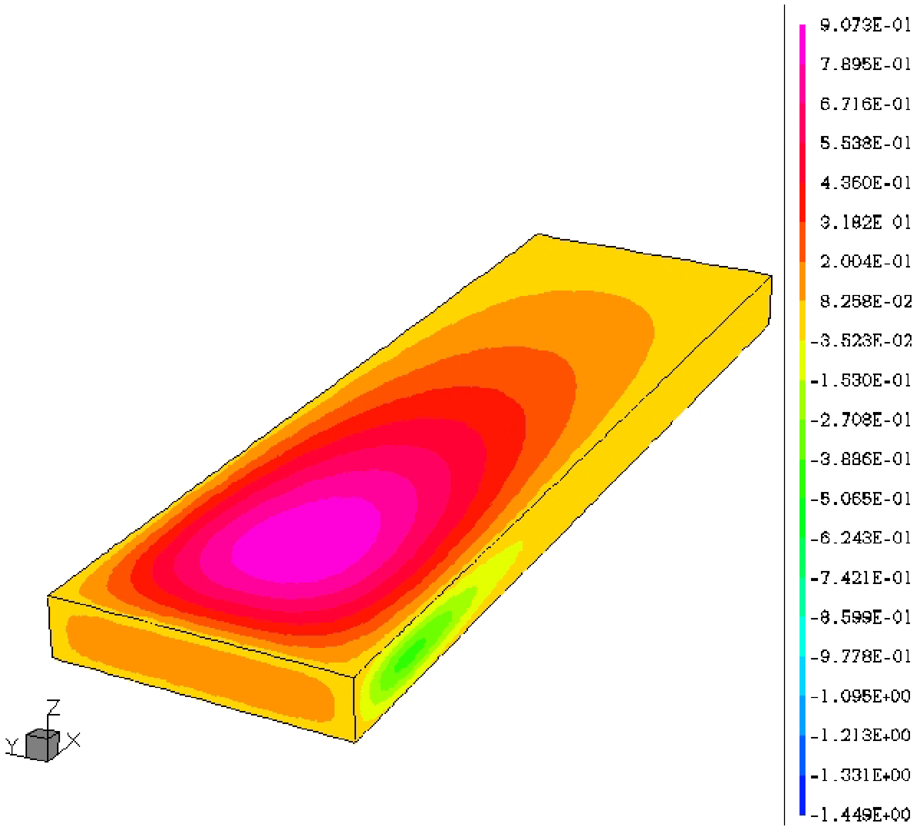


Figure 24: Electric potential of walls in the horizontal cavity. The wall currents are perpendicular to the isovalue of the potential

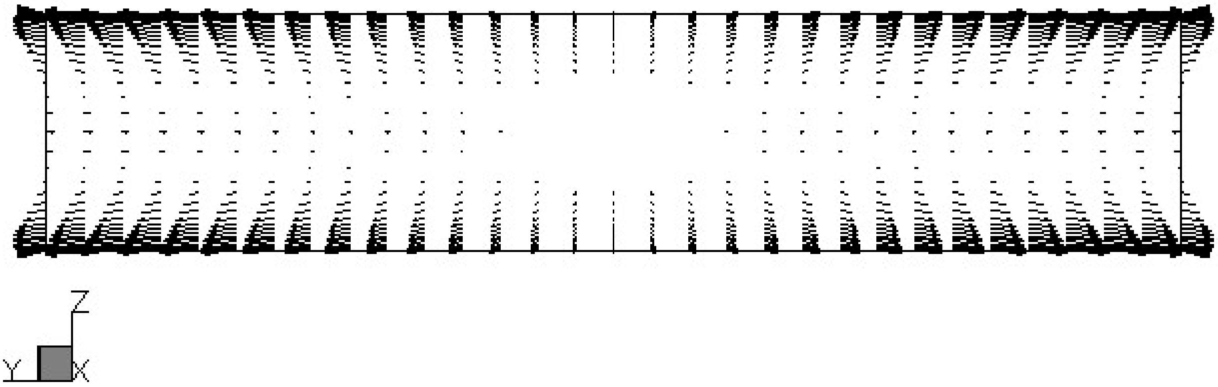


Figure 25: Electric current vectors in the plane *x* = -1.0 (*Ha* = 10000, Horizontal cavity)

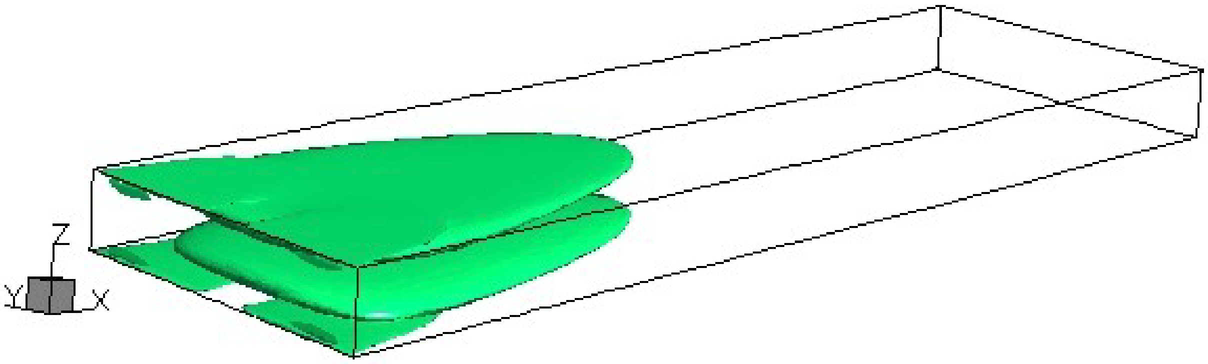


Figure 26: Zone where the velocity is higher than 0.5 mm/s (*Ha*=10000, horizontal cavity)

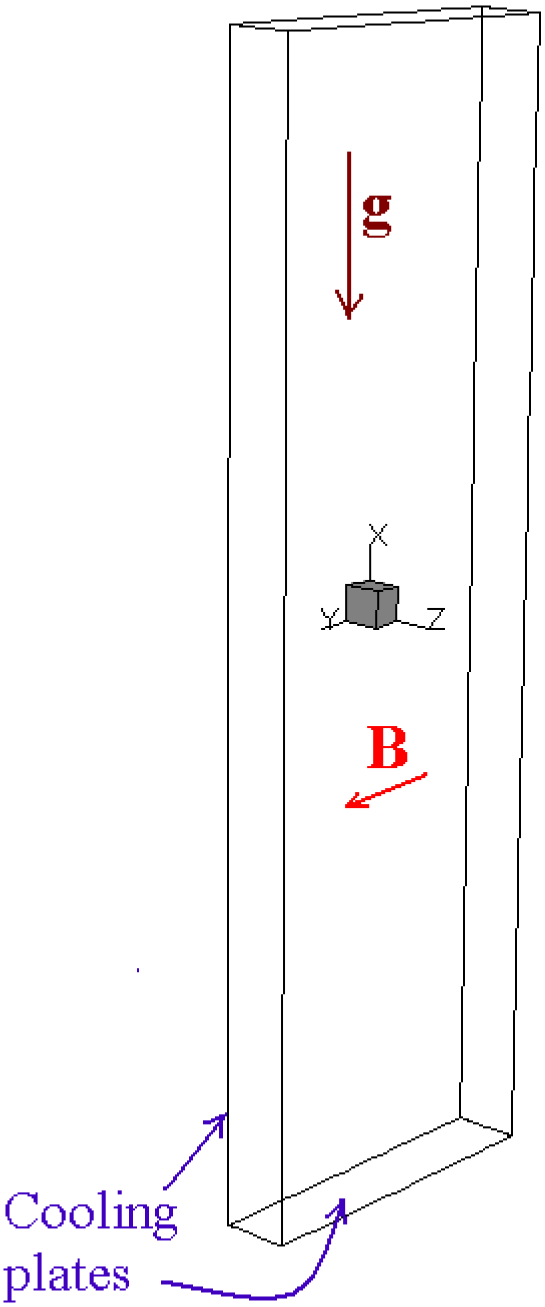


Figure 27: Geometry of the vertical configuration.

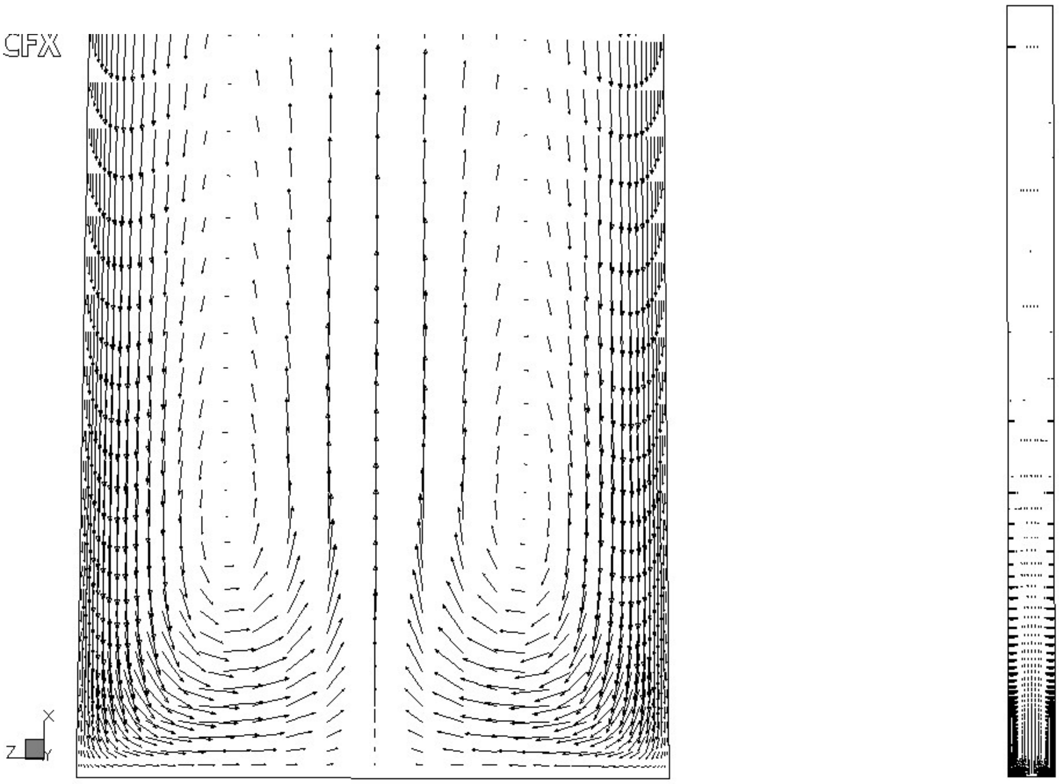


Figure 28: Velocity vector plot in the plane y = 0 for the vertical cavity. (*Ha*=10000)

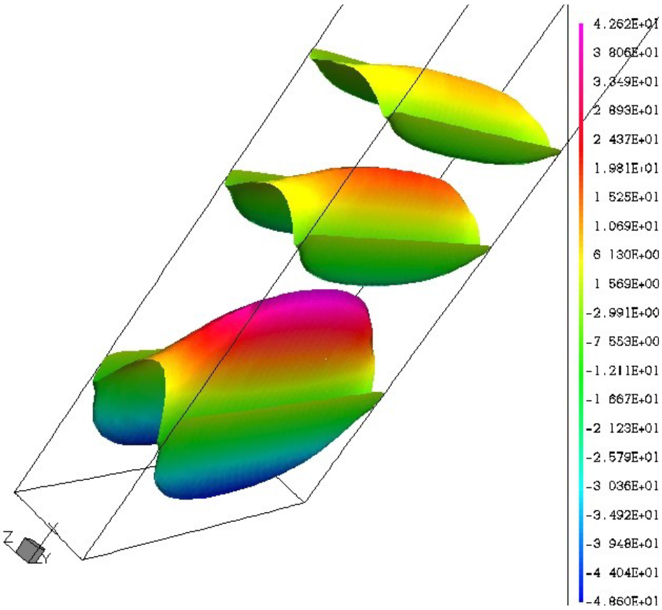
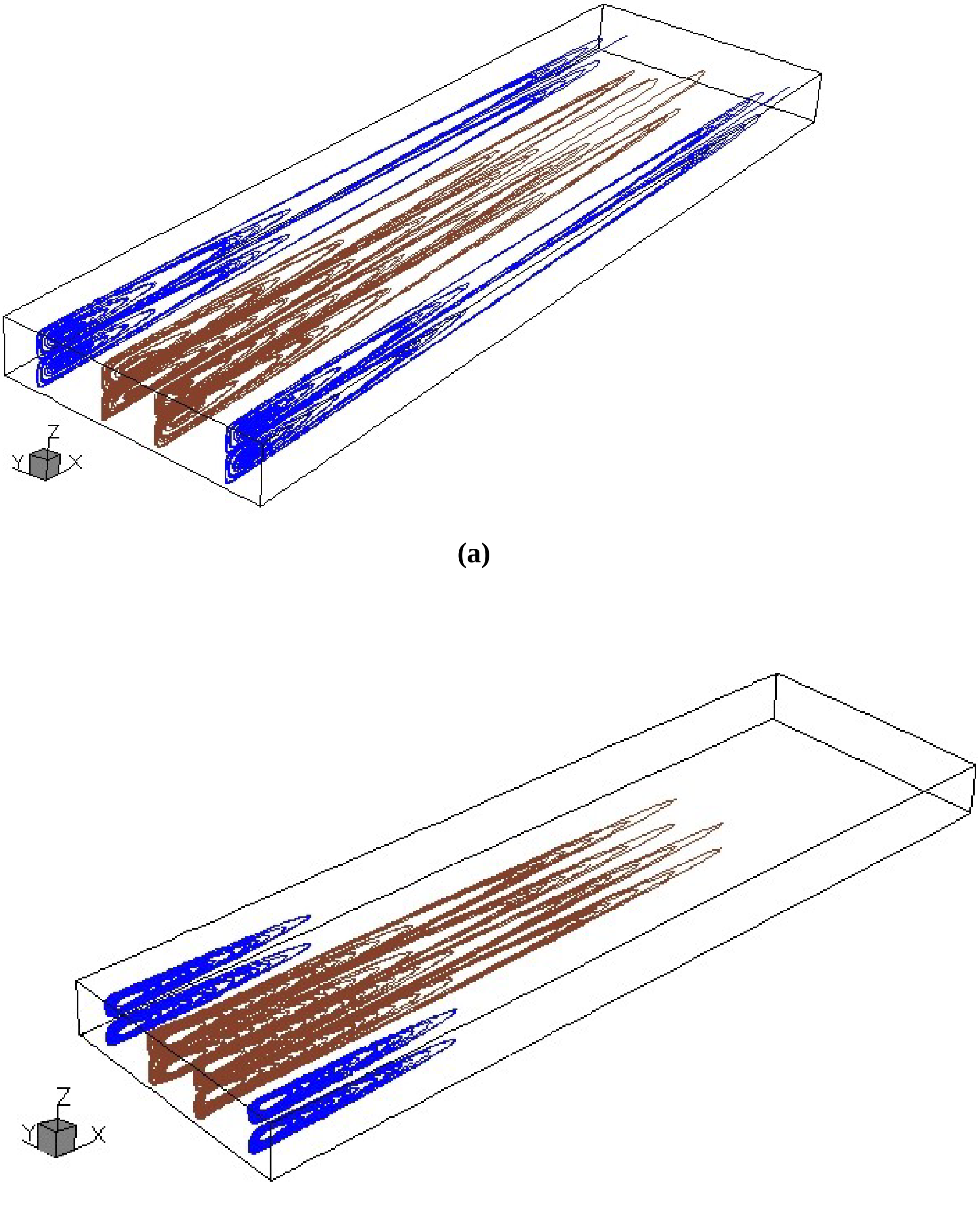


Figure 29: *u* velocity in transverse horizontal planes (x = constant (Vertical cavity, *Ha* =10000)



(a)

(b)

Figure 30: Velocity streamlines for (a) *Ha* = 100, and (b) *Ha* = 10000. (Vertical cavity)

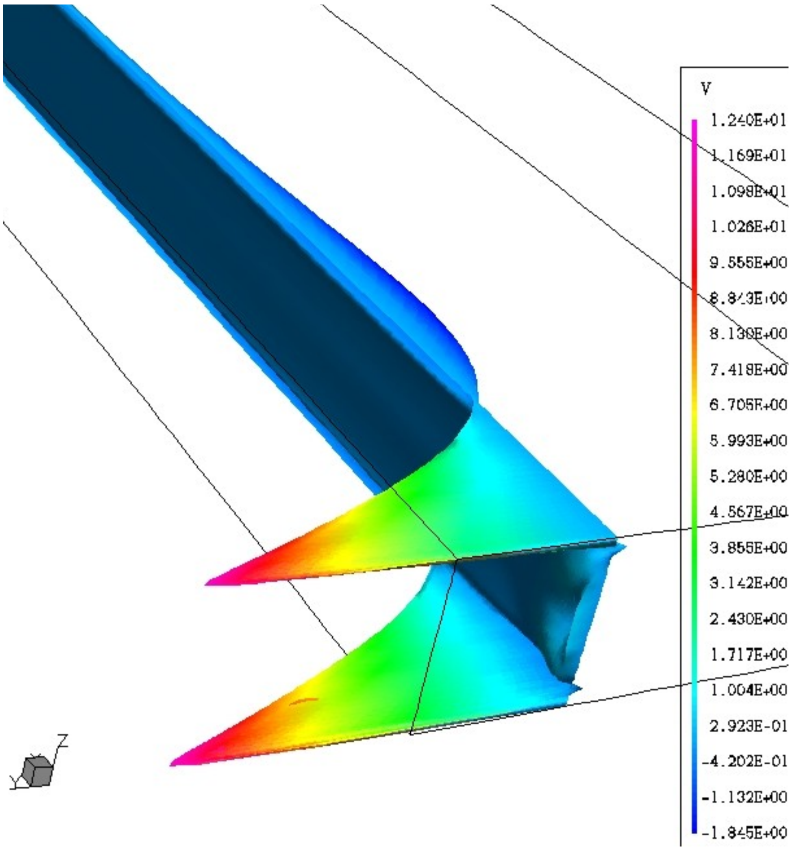
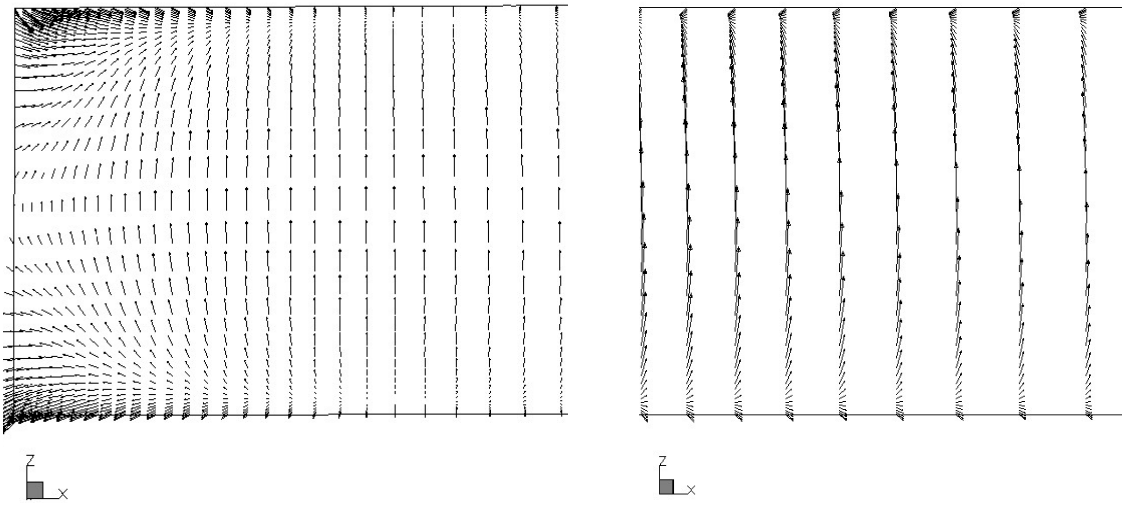


Figure 31: *V* velocity in the plane *y* = 0.5 (*Ha* = 10000, vertical cavity).



(a) **(b)**

Figure 32: Electric current vector plotted in the midplane *y* = 0, (a) at the hot extremity and (b) at the centre of the cavity. (Vertical cavity, *Ha* = 10000)

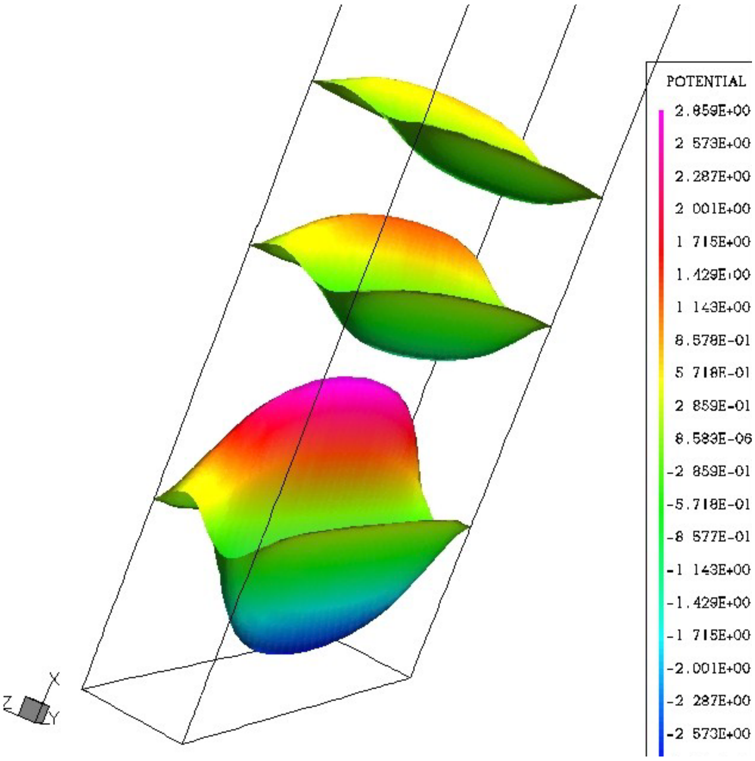


Figure 33: Potential in transverse horizontal planes (*x* = constant). Vertical cavity, *Ha* = 10000

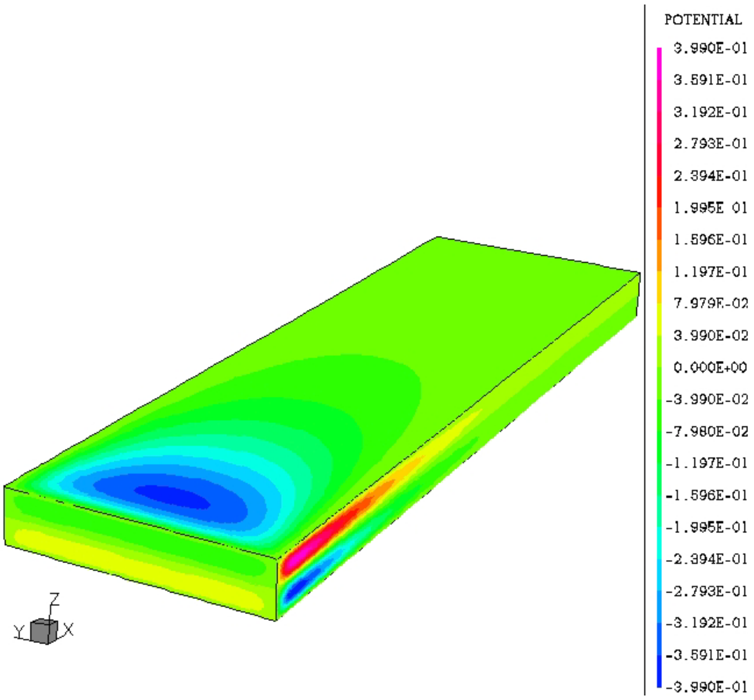


Figure 34: Electric potential of walls in the vertical cavity with *Pe* = 0. The wall currents are perpendicular to the iso-value of the potential.

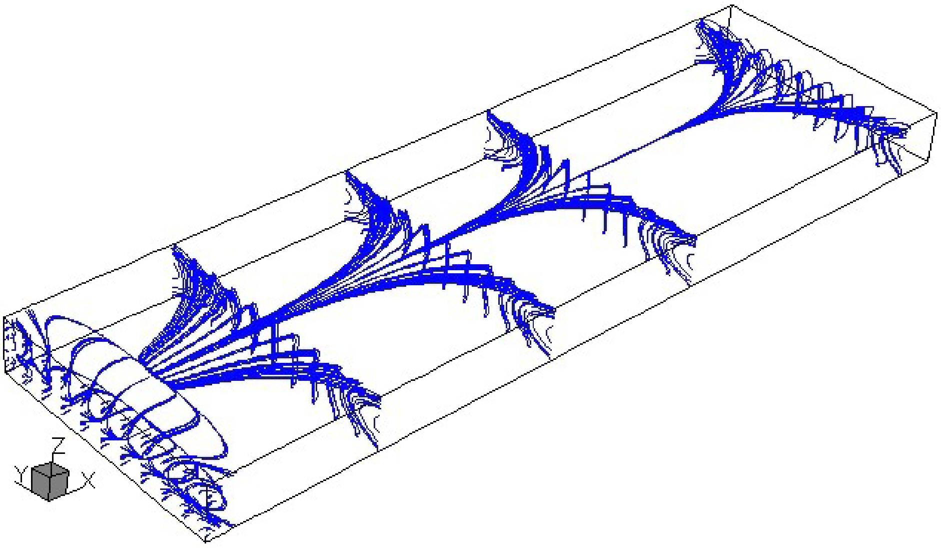


Figure 35: Electric current streamlines in the core and the upper parallel layer (Vertical cavity, *Ha* = 10000)

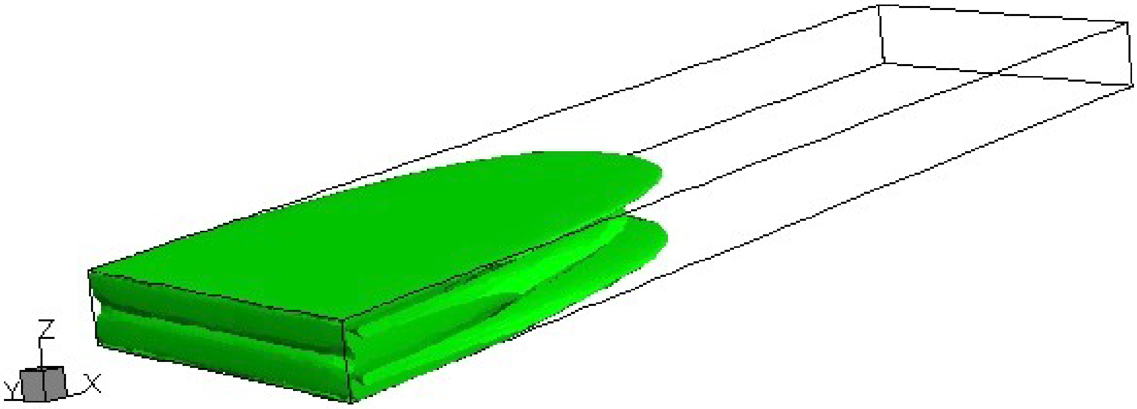


Figure 36: Zone where the velocity is higher than 0.5 mm/s (*Ha*=10000, vertical cavity)

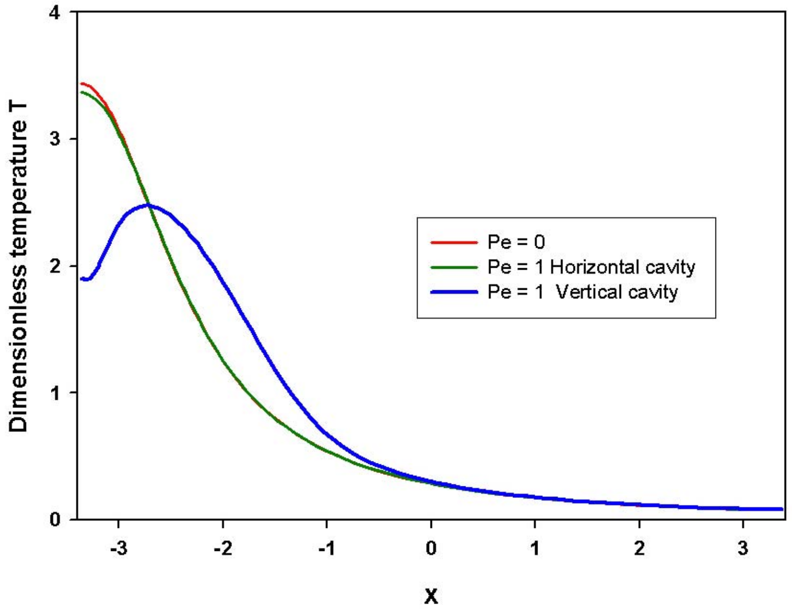
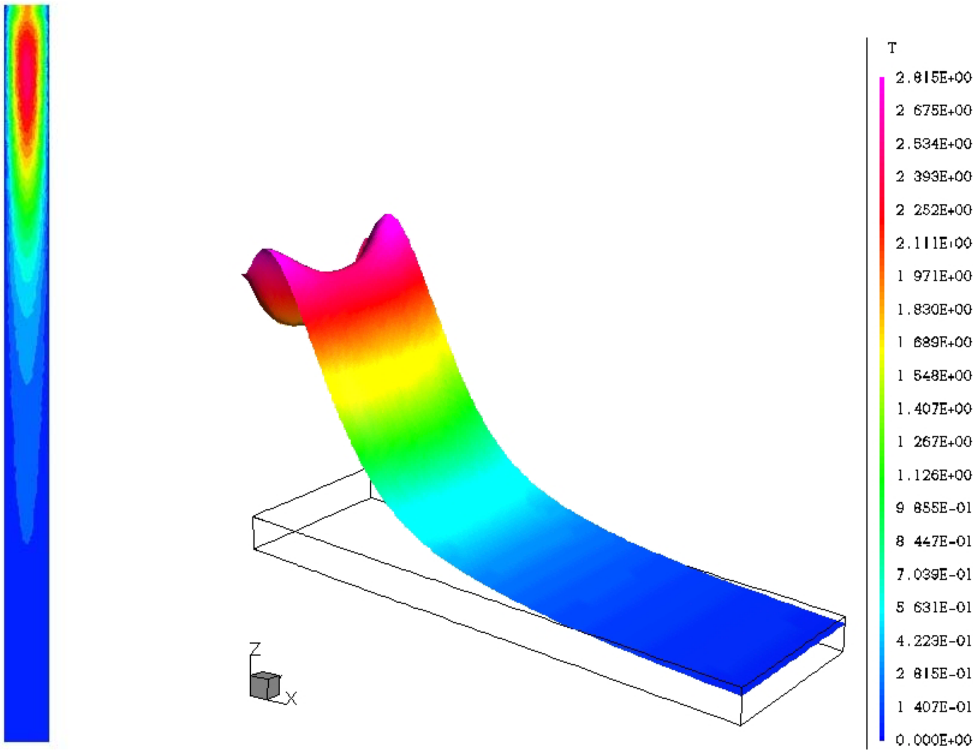


Figure 37 : Temperature distribution along the axis *y* =*z*= 0 for the purely diffusive case and for *Pe* = 1.



(a) **(b)**

Figure 38: Dimensionless temperature distribution in the midplanes (a) *y* = 0, and (b) *z* = 0. (*Pe = 1, Ha= 10000,* Vertical cavity)

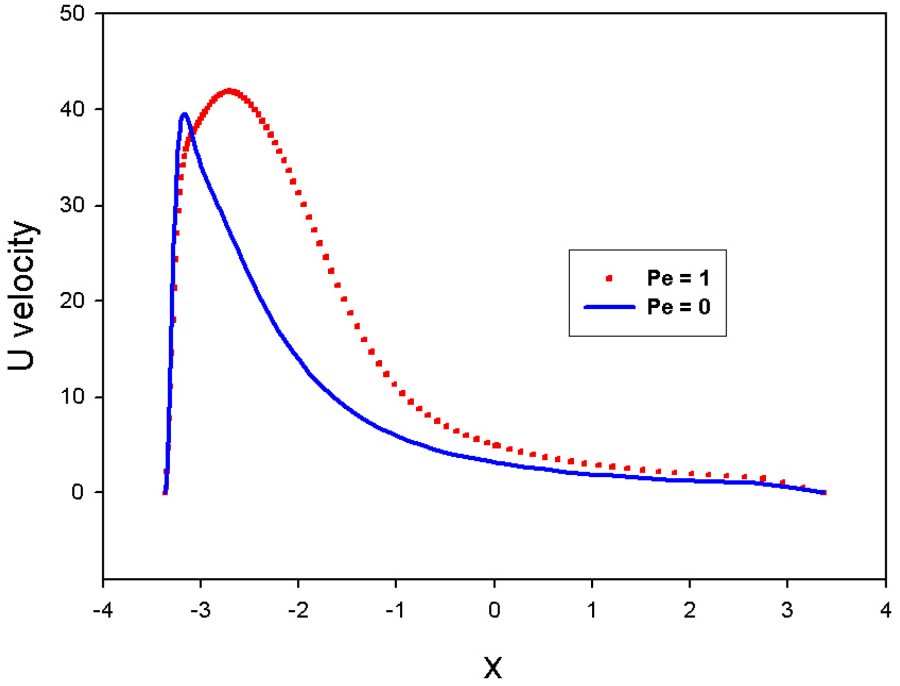


Figure 39: Evolution of the u velocity along the axis y = 0 (vertical cavity).

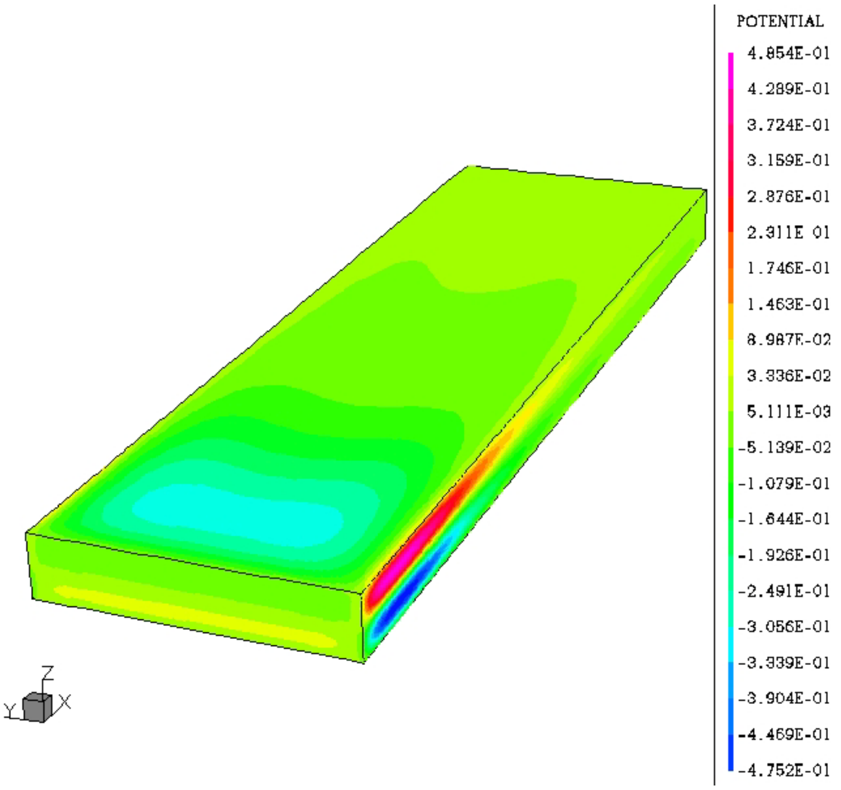


Figure 40: Electric potential of walls in the vertical cavity with *Pe* =1. The wall currents are transverse to the iso-value of the potential.

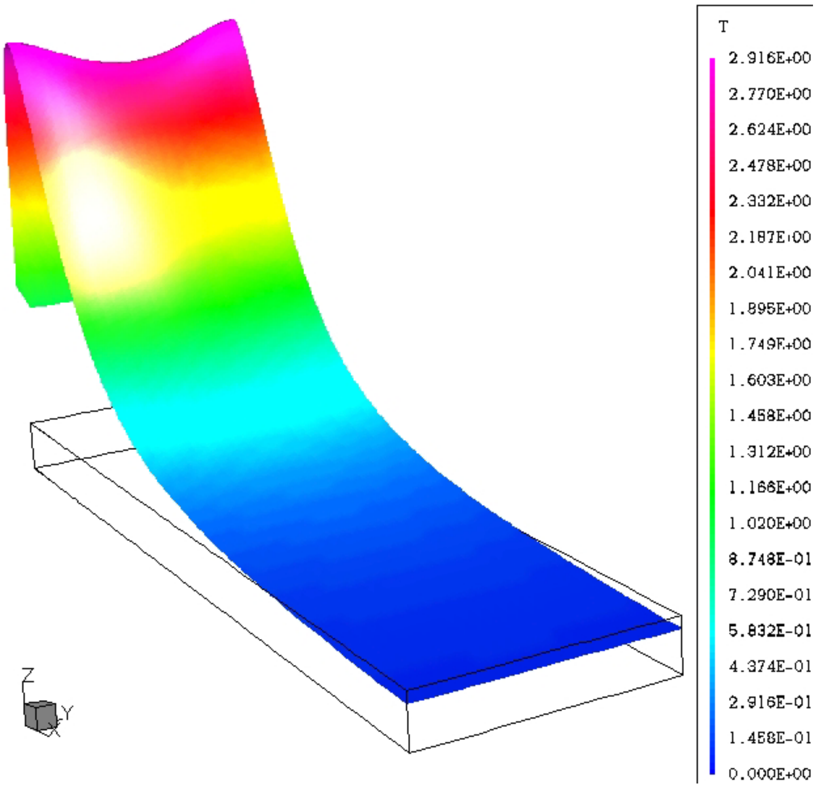


Figure 41: Dimensionless temperature distribution in the mid-plane *z* = 0. (cross flow 1mm/s *Pe = 1, Ha= 10000,* Horizontal duct)

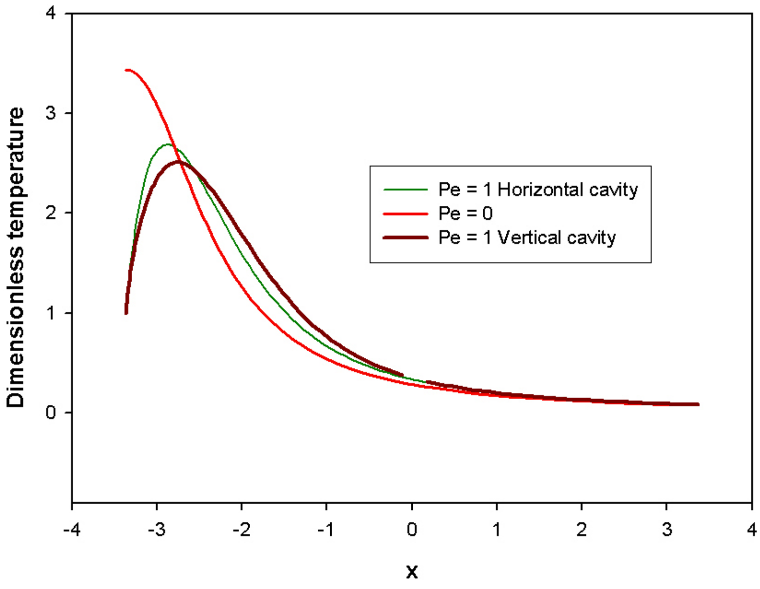


Figure 42: Dimensionless temperature distribution along the *x*-axis (*y* = 0, *z*= 0) (Cross flow 1mm/s, *Pe* = 1, *Ha*= 10000).

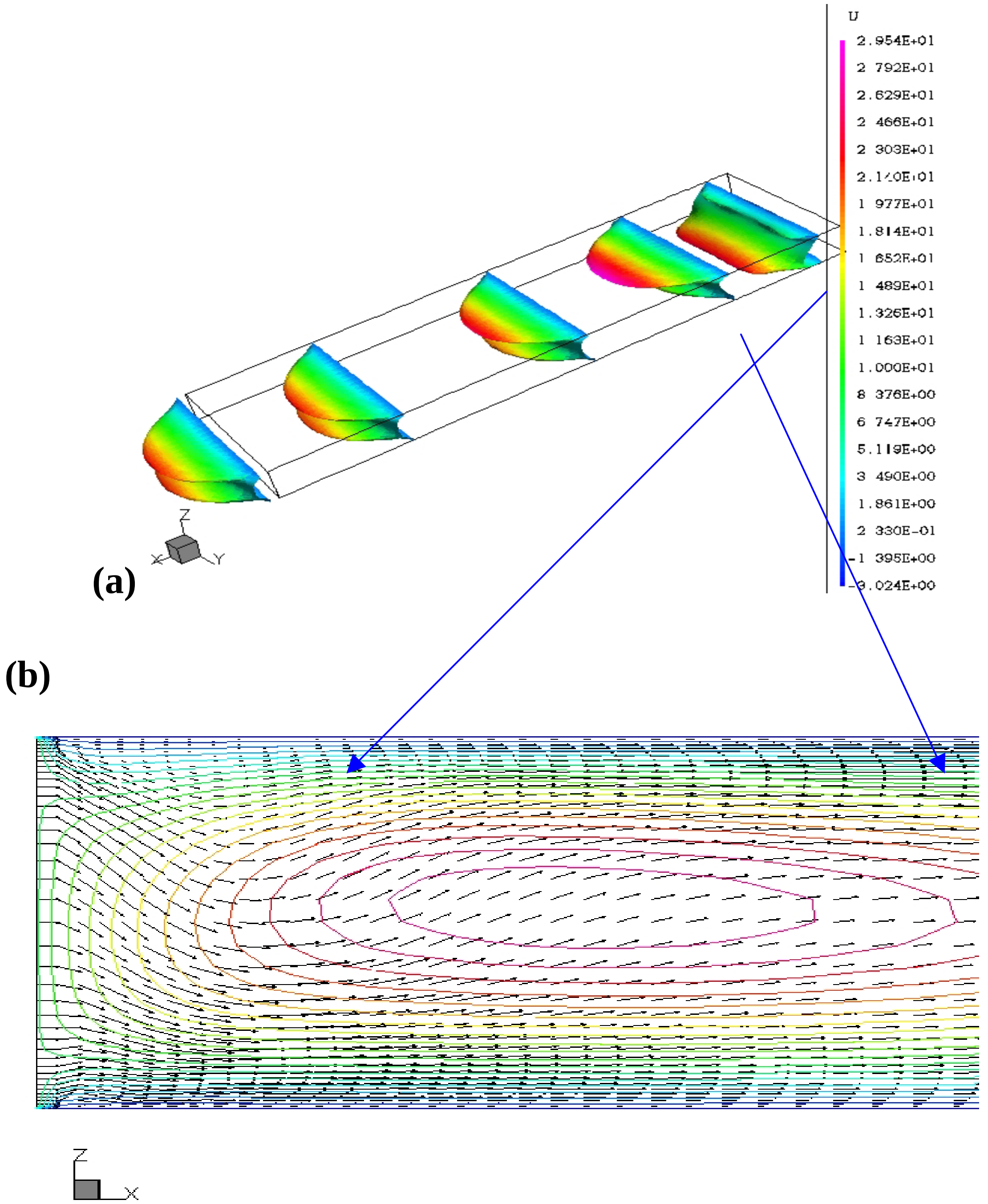


Figure 43: (a) *u* velocity in transverse vertical planes (*x* = constant). The arrows indicate the corresponding position of the two first profiles (from the entrance) on the following figure. **(b)** Velocity vectors and temperature distribution near the inlet in the plane *y* = 0. (*Pe* =1, Horizontal duct, *Ha* = 10000)

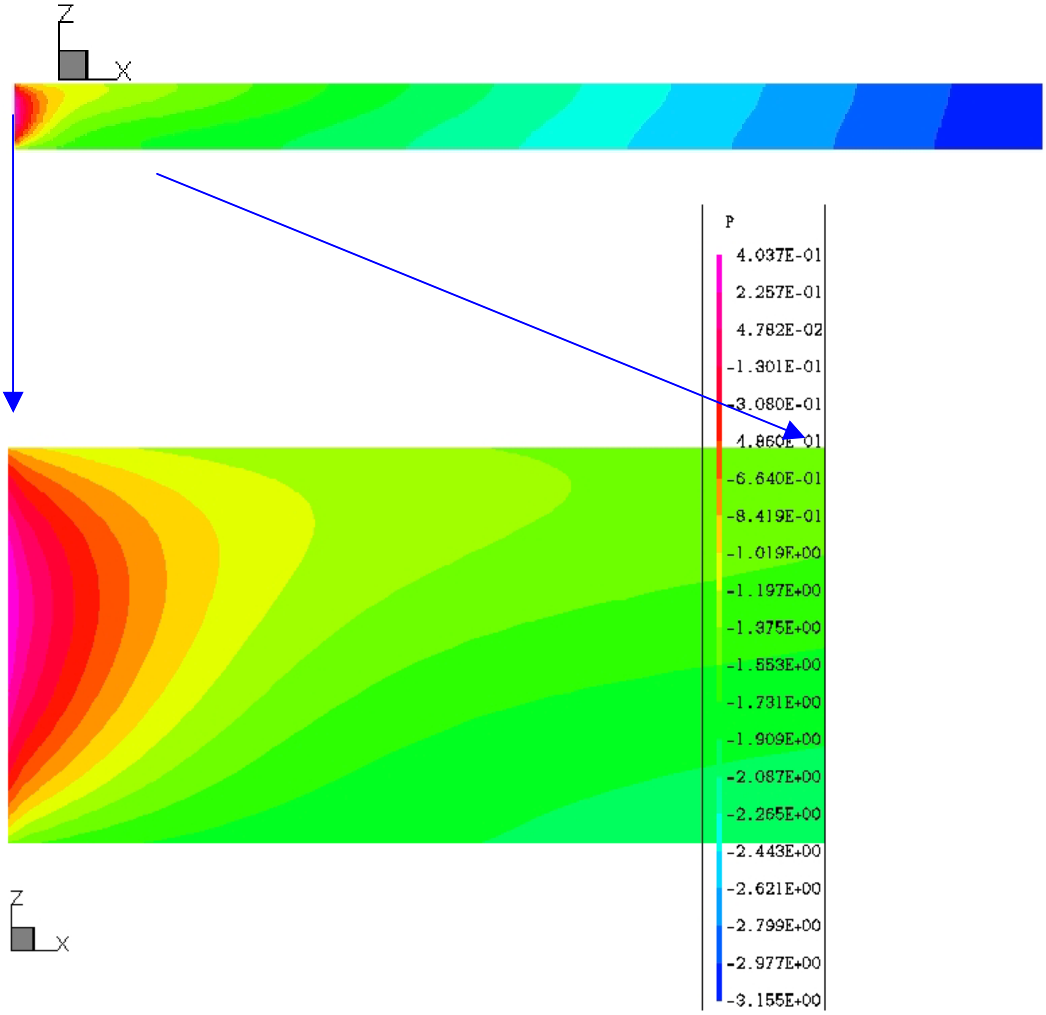


Figure 44: Distribution of the pressure in the horizontal duct. (*Ha* = 10000, *Pe* =1)

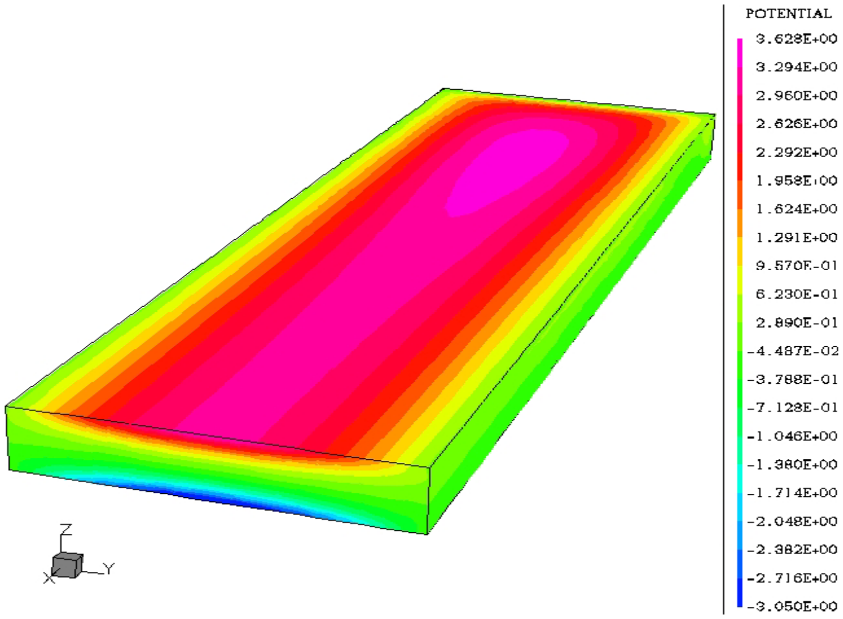


Figure 45: Electric potential on the boundary of the geometry, (the outlet, a cooling plate and a Hartmann wall are represented). (Horizontal duct, *Ha* = 10000, *Pe* = 1)

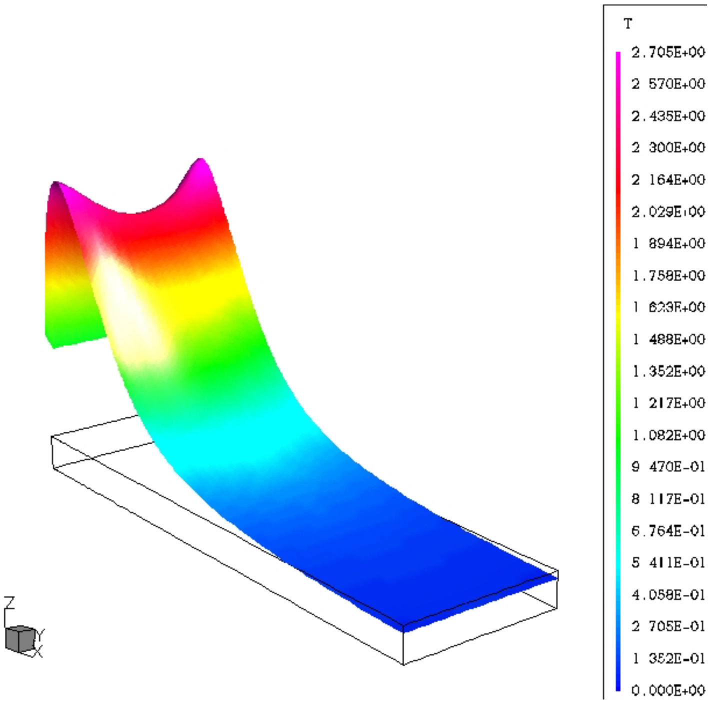


Figure 46: Temperature distribution in the plane *z* = 0. (Vertical duct, *Ha* = 10000, *Pe*=1)

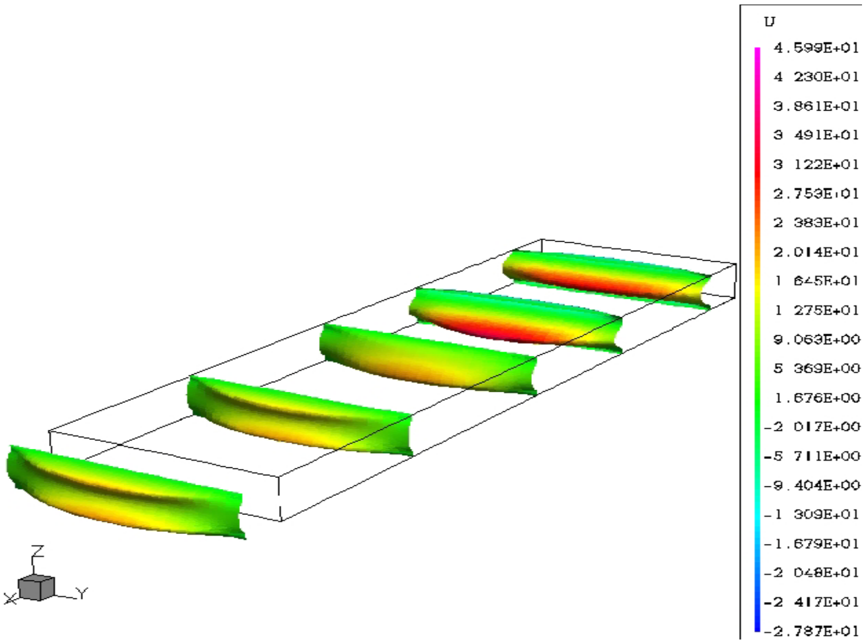


Figure 47 : *u* velocity in transverse vertical planes (*x* = constant) (*Pe* =1, Vertical duct, *Ha* = 10000)

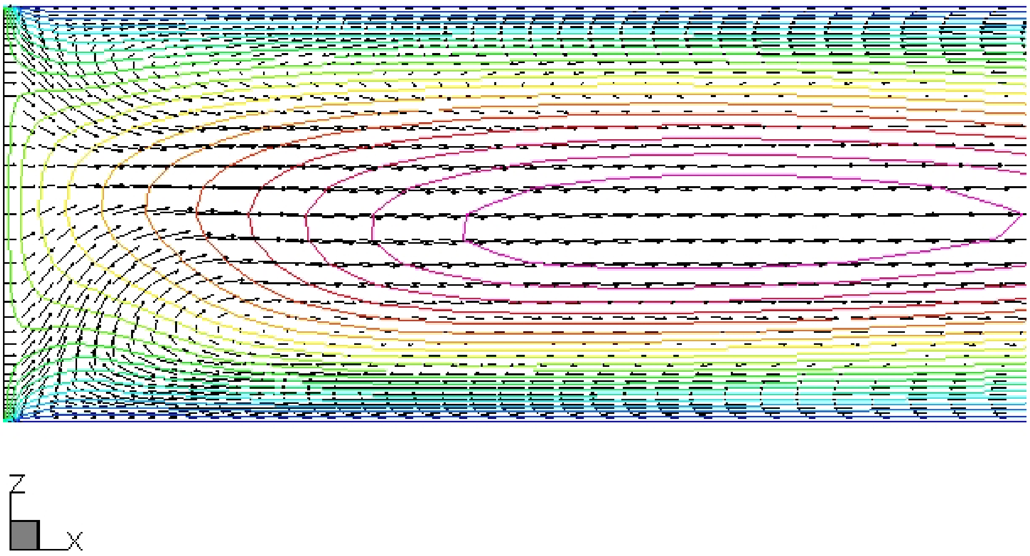


Figure 48: Velocity vectors and temperature distribution at the vicinity of the inlet in the plane *y* = 0. (*Pe* =1, Vertical duct, *Ha* = 10000)

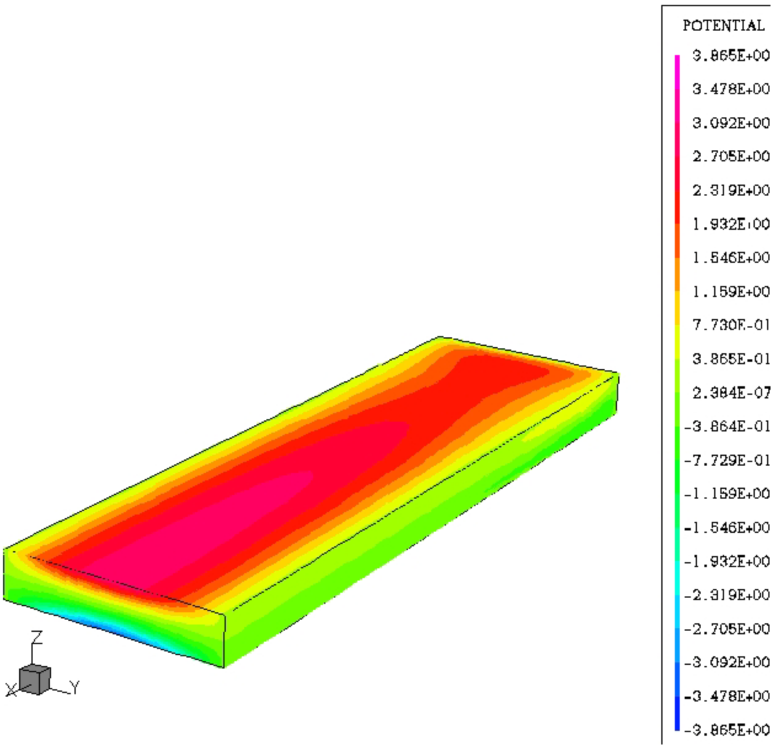


Figure 49: Electric potential on the boundary of the geometry, (the outlet, a cooling plate and a Hartmann wall are represented). (Vertical duct, *Ha* = 10000, *Pe* = 1)