2

EGG-WTD-9845
November 1991

INFORMAL REPORT

A Computational Model for Viscous Fluid Flow, Heat Transfer, and Melting in In Situ Vitrification Melt Pools

P. R. McHugh
J. D. Ramshaw

Idaho National Engineering Laboratory

Managed by the U.S. Department of Energy

EG&G Idaho

Work performed under DOE Contract No. DE-AC07-76ID01570

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

This document contains new concepts or the author(s) interpretation of new calculations and/or measurements; accordingly, EG&G Idaho, Inc. is required by the United States Government to include the following disclaimer:

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product or process disclosed, or represents that its use would not infringe privately owned rights. References herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

EGG-WTD--9845

DE92 012215

A Computational Model for Viscous Fluid Flow, Heat Transfer, and Melting in In Situ Vitrification Melt Pools

P. R. McHugh and J. D. Ramshaw

Published November 1991

Idaho National Engineering Laboratory
EG&G Idaho, Inc.
Idaho Falls, ID 83415

Prepared for the
U.S. Department of Energy
Office of Environmental Restoration & Waste Management
Under DOE Field Office, Idaho
Contract DE-AC07-76IDO1570

A Computational Model for Viscous Fluid Flow, Heat Transfer, and Melting in In Situ Vitrification Melt Pools

EGG-WTD-9845

Prepared by:

Paul McHugh 10/24/91
P. R. McHugh Date

John Ramshaw 10/25/91
J. D. Ramshaw Date

Reviewed by:

Robert J MacKinnon 10/25/91
Robert J. Mackinnon Date

Kenneth Merrill 10/28/91
Ken Merrill Date

ABSTRACT

MAGMA is a FORTRAN computer code designed to simulate viscous flow in in situ vitrification melt pools. It models three-dimensional, incompressible, viscous flow and heat transfer. The momentum equation is coupled to the temperature field through the buoyancy force terms arising from the Boussinesq approximation. All fluid properties, except density, are assumed variable. Density is assumed constant except in the buoyancy force terms in the momentum equation. A simple melting model based on the enthalpy method allows the study of the melt front progression and latent heat effects. An indirect addressing scheme used in the numerical solution of the momentum equation avoids unnecessary calculations in cells devoid of liquid. Two-dimensional calculations can be performed using either rectangular or cylindrical coordinates, while three-dimensional calculations use rectangular coordinates. All derivatives are approximated by finite differences. The incompressible Navier-Stokes equations are solved using a new fully implicit iterative technique, while the energy equation is differenced explicitly in time. Spatial derivatives are written in conservative form using a uniform, rectangular, staggered mesh based on the marker and cell placement of variables. Convective terms are differenced using a weighted average of centered and donor cell differencing to ensure numerical stability. Complete descriptions of MAGMA governing equations, numerics, code structure, and code verification are provided.

CONTENTS

FIGURES

TABLES

NOMENCLATURE

a	Artificial sound speed
$a1$	Polynomial coefficient (user input)
$a2$	Polynomial coefficient (user input)
$a3$	Polynomial coefficient (user input)
$a4$	Polynomial coefficient (user input)
b	Artificial bulk viscosity
c	Specific heat capacity
e	Specific energy
$\mathbf{e_x}$	Unit vector in x-direction
$\mathbf{e_y}$	Unit vector in y-direction
$\mathbf{e_z}$	Unit vector in z-direction
$\mathbf{g}$	Gravity vector
h	Convective heat transfer coefficient
H	Typical height
i	Cell index in the x-direction
j	Cell index in the y-direction
k	Cell index in the z-direction
k	Thermal conductivity
L	Latent heat of fusion
L_C	Typical length
Nu	Nusselt number
p	Pressure (divided by density)
PCM	Phase Change Material
Pr	Prandtl number
q	Heat generation term
q''	Heat flux
Ra	Rayleigh number
Ste	Stefan number
T	Temperature

T_m	Melting temperature.
T_o	Reference temperature
$\mathbf{u}$	Velocity Vector
u	Velocity in x-direction
V	Volume of a computational cell
v	Velocity in y-direction
w	Velocity in z-direction

Greek Symbols:

α	Thermal diffusivity $(\alpha = \frac{k}{\rho c})$ (38)
β	Coefficient of thermal expansion
$\Delta\tau$	Artificial or pseudo-time step
Δx	Spatial increment in x-direction
Δy	Spatial increment in y-direction
Δz	Spatial increment in z-direction
ν	Kinematic viscosity
ρ	Density
σ	Viscous stress tensor
τ	Dimensionless time
ω	Dimensionless damping coefficient of order unity
ξ	Cylindrical coordinates flag
α	Safety factor on the pseudo-time step $(0 < \alpha < 1)$
ε_1	Convergence tolerance on $\nabla \cdot \mathbf{u}$
ε_2	Convergence tolerance on $\frac{\partial u}{\partial \tau}$, $\frac{\partial v}{\partial \tau}$, and $\frac{\partial w}{\partial \tau}$

Subscripts:

B	Bottom side of a computational cell
D	Derriere side of a computational cell
F	Front side of a computational cell

L	Left side of a computational cell
R	Right side of a computational cell
T	Top side of a computational cell
w	Wall value
x	Value in the x-direction
y	Value in the y-direction
z	Value in the z-direction
∞	Value of surroundings at infinity

Superscripts:

(k)	Iteration level
mq	Refers to a particular momentum cell denoted by q with q equal either to u, v, or w
n	Time level

Operators:

∇	Gradient operator
$(\mathbf{A})^{\mathrm{T}}$	Transpose of **A**
$max(A)$, A_{max}	Search over all computational cells for the maximum value of A
$min(A)$, A_{min}	Search over all computational cells for the minimum value of A
$<A>_{ijk}$	Volume average of A over cell (i,j,k)
$(\overline{A})$	Average value of A

A COMPUTATIONAL MODEL FOR VISCOUS FLUID FLOW, HEAT TRANSFER, AND MELTING IN IN SITU VITRIFICATION MELT POOLS

1. INTRODUCTION

MAGMA is a finite difference computer code developed to simulate the three dimensional viscous fluid flow and heat transfer within in situ vitrification (ISV) melt pools. The starting point for the development of MAGMA was an experimental, fully implicit, two-dimensional, incompressible flow, computer code.[a]

The objective of this work was to devise, implement, and test the code modifications needed to permit three-dimensional simulations of melt flow in partially molten domains with curved boundaries. The tasks completed in achieving this objective include:

- The development of boundary condition logic for walls that pass diagonally through cell faces in two dimensions.
- Implementing and testing the two-dimensional diagonal boundary condition logic.
- Converting from direct to indirect, addressing to facilitate limiting calculation of fluid flow to the molten part of the domain.
- The addition of three-dimensional capability.
- A code generalization to allow for variable properties.
- Implementing the Boussinesq buoyancy force terms in the momentum equation.[1]
- Implementing an explicit, finite difference solution of the energy equation.
- A code generalization to allow cell melting, freezing, and the simulation of latent heat effects.

[a] Unpublished research results of J. D. Ramshaw and P.R. McHugh, entitled "Hybrid Iteration Scheme for Implicit Calculations of Incompressible Flow."

The purpose of this report is to document the MAGMA code so that it can serve as a guide for code use, and as a reference source if code modifications are required in the future. Therefore, it includes descriptions of: the basic governing equations and numerics, the finite difference mesh, boundary condition implementation, code structure, code input and output, and code options.

The report is organized into five main sections. The following section describes the basic governing equations, the computational mesh, the basic differencing scheme, and boundary condition implementation. Section 3 is concerned with code structure and organization, code input and output, and code options. Section 4 describes code verification and testing. The final section contains a summary, and recommendations for further code development.

2. GOVERNING EQUATIONS AND NUMERICS

This section describes MAGMA's basic governing partial differential equations and summarizes the numerical techniques used to solve these equations. Included in this discussion are: descriptions of the three-dimensional finite difference mesh, differencing schemes, and boundary condition implementation.

2.1 Governing Equations

The governing equations used by MAGMA are the conservation equations for incompressible viscous flow and heat transfer in three spatial dimensions. The coupl'ng between the momentum and energy equations occurs through the buoyancy force term in the momentum equation, using the Boussinesq approximation.[1]

2.1.1 Incompressible Navier-Stokes Equations

Fluid flow is governed by the incompressible Navier-Stokes equations, namely

$$\nabla \cdot \mathbf{u} = 0 \tag{1}$$

$$\frac{\partial \mathbf{u}}{\partial t} + \nabla \cdot (\mathbf{u}\mathbf{u}) = - \nabla p - \beta \mathbf{g}(T - T_O) + \nabla \cdot \sigma \tag{2}$$

where the viscous stress tensor, σ, is given by,

$$\sigma = \nu\big((\nabla \mathbf{u}) + (\nabla \mathbf{u})^T\big) \tag{3}$$

where

$\mathbf{g}$	=	gravity vector
p	=	pressure (divided by density)
T	=	temperature
T_O	=	reference temperature

$\mathbf{u}$ = velocity vector

β = coefficient of thermal expansion

ν = kinematic viscosity

∇ = vector gradient operator

$(\mathbf{A})^{\mathrm{T}}$ = transpose of **A.**

MAGMA allows the use of two and three-dimensional rectangular coordinates, and two-dimensional cylindrical coordinates. When the two-dimensional option is invoked, all terms associated with the *z*-coordinate vanish. Two-dimensional cylindrical coordinates take the *y*-axis as the axis of symmetry, and the *x*-coordinate as the radial coordinate. Equations (4) through (7) below, are Equations (1) through (3) expressed in terms of coordinates and velocity components. $\xi = 1$ corresponds to two-dimensional cylindrical coordinates, while $\xi = 0$ corresponds to rectangular coordinates.

- Continuity equation:

$$\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z} + \xi \frac{u}{x} = 0 \tag{4}$$

- Momentum equation in x-direction:

$$\frac{\partial v}{\partial t} + \frac{\partial u^2}{\partial x} + \frac{\partial uv}{\partial y} + \frac{\partial uw}{\partial z} + \xi \frac{u^2}{x} = - \frac{\partial p}{\partial x} - \beta \ (\mathbf{g} \cdot \mathbf{e_x})(T - T_o)$$

$$+ \frac{1}{x^{\xi}} \nabla \cdot \left(x^{\xi} \sigma_{\mathbf{x}} \right) - \left(\frac{2 \xi \nu \upsilon}{x^2} \right) \tag{5}$$

- Momentum equation in y-direction

$$\frac{\partial v}{\partial t} + \frac{\partial uv}{\partial x} + \frac{\partial v^2}{\partial y} + \frac{\partial vw}{\partial z} + \xi \frac{uv}{x} = - \frac{\partial p}{\partial y} - \beta\,(\mathbf{g}\cdot\mathbf{e_y})(T - T_o)$$

$$+ \frac{1}{x^\xi}\,\nabla\cdot\left(x^\xi \sigma_\mathbf{y}\right) \tag{6}$$

- Momentum equation in z-direction (not used when $\xi = 1$)

$$\frac{\partial w}{\partial t} + \frac{\partial uw}{\partial x} + \frac{\partial vw}{\partial y} + \frac{\partial w^2}{\partial z} = - \frac{\partial p}{\partial z} - \beta\ (\mathbf{g}\cdot\mathbf{e_z})(T - T_o) + \nabla\cdot\left(\sigma_\mathbf{z}\right) \tag{7}$$

where

$$\sigma_\mathbf{x} = \sigma\cdot\mathbf{e_x} = \nu\ \left(\nabla u\ + \frac{\partial \mathbf{u}}{\partial x}\right) \tag{8}$$

$$\sigma_\mathbf{y} = \sigma\cdot\mathbf{e_y} = \nu\ \left(\nabla v\ + \frac{\partial \mathbf{u}}{\partial y}\right) \tag{9}$$

$$\sigma_\mathbf{z} = \sigma\cdot\mathbf{e_z} = \nu\ \left(\nabla w\ + \frac{\partial \mathbf{u}}{\partial z}\right) \tag{10}$$

where

$\mathbf{e_x}$	=	unit vector in x-direction
$\mathbf{e_y}$	=	unit vector in y-direction
$\mathbf{e_z}$	=	unit vector in z-direction
u	=	x-component of **u**
v	=	y-component of **u**
w	=	z-component of **u**.

Note that the vector gradient operator ∇ in Equations (4) through (10), is defined as

$$\nabla = \mathbf{e_x} \frac{\partial}{\partial x} + \mathbf{e_y} \frac{\partial}{\partial y} + \mathbf{e_z} \frac{\partial}{\partial z} \tag{11}$$

with the understanding that the z-component vanishes in two-dimensions. This convention is used even in the cylindrical case. Thus divergences in Equations (4) through (10) represent the Euclidean divergence, not covariant divergence.

2.1.2 Energy Equation

The energy conservation equation, which governs heat transfer, is given by

$$\rho \left(\frac{\partial e}{\partial t} + \nabla \cdot (e\mathbf{u})\right) = \nabla \cdot (k \nabla T) + q \tag{12}$$

where

e = specific energy

k = thermal conductivity

q = heat generation

ρ = density.

Viscous dissipation due to viscosity has been neglected, as it is negligible in incompressible flow.

In terms of the spatial coordinates, Equation (12) becomes

$$\rho \left(\frac{\partial e}{\partial t} + \frac{\partial (ue)}{\partial x} + \frac{\partial (ve)}{\partial y} + \frac{\partial (we)}{\partial z} + \xi \frac{e\,u}{x}\right) = \frac{\partial}{\partial x}\left(k \frac{\partial T}{\delta x}\right)$$

$$+ \frac{\partial}{\partial y}\left(k \frac{\partial T}{\partial y}\right) + \frac{\partial}{\partial z}\left(k \frac{\partial T}{\partial z}\right) + \xi \frac{k}{x} \frac{\partial T}{\partial x} + q. \tag{13}$$

Terms associated with the z-coordinate vanish in two-dimensions.

2.1.3 State Relation and Melting Model

An equation of state is used to relate cell energies to cell temperatures. This state equation accounts for latent heat effects, and allows user modification for use with different types of fluids. The form of the state relation is illustrated in Figure 1. Note that the zero of energy has arbitrarily been taken to be the energy of the solid phase at the melting point. For $e < 0$ and $e > L$, the curve is assumed to obey a quadratic relation of the form

$$e = \begin{bmatrix} a1\ (T - T_m) + a2\ (T - T_m)^2 & , & e < 0 \\ L + a3\ (T - T_m) + a4\ (T - T_m)^2 & , & e > L \end{bmatrix} \tag{14}$$

where the coefficients *a1*, *a2*, *a3*, and *a4* are user input numbers, and

L = latent heat of fusion

T_m = melting temperature.

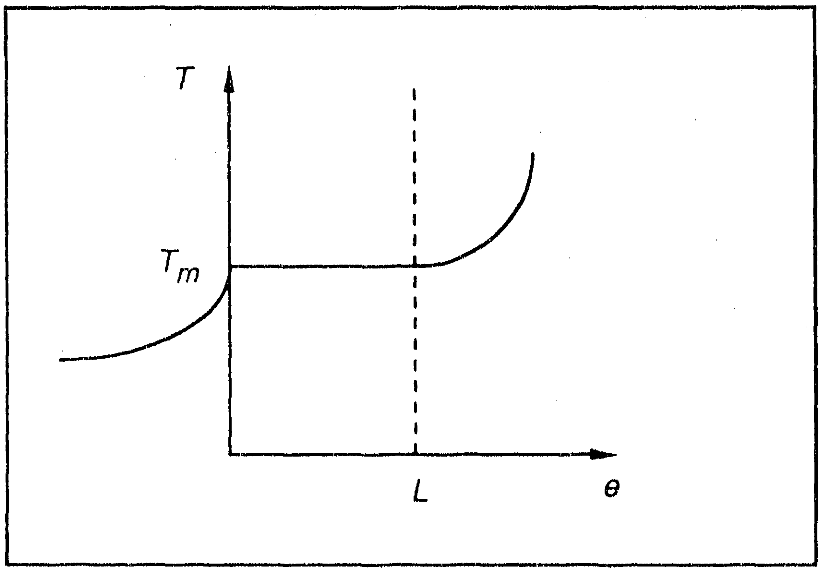


Figure 1. Equation of state cuive.

For $0 < e < L$, the temperature remains constant and is equal to the melt temperature of the fluid, T_m.

If the curves are linear for $e < 0$ and $e > L$ (i.e., $a2 = a4 = 0$), then $a1$ is the specific heat of the solid phase ($e < 0$) and $a3$ is the specific heat of the liquid phase ($e > L$) of the material. If $L = a1 = a2 = a4 = 0$ and $a3$ is the specific heat of the material, Equation (13) reduces to the temperature equation in the absence of phase change.

A cell (i,j,k) is assumed to be solid when $e < 0$. Partial melting occurs when $0 < e < L$, and complete melting when $e > L$. Presently, MAGMA does not allow for partially molten cells. Thus, the condition $e > L/2$ was chosen as the criterion for cell melting. If the cell energy of a solid cell rises above $L/2$, the cell melts. Conversely, if the cell energy of a molten cell falls below $L/2$, it solidifies.

MAGMA assumes that all fluid properties, except density, are variable (density is allowed to vary only in the buoyancy force term in the momentum equation through the Boussinesq approximation).[1] If fluid properties are allowed to vary, relations describing this behavior or specific property data must be supplied by the user.

2.2 Temporal Differencing and Time Integration Procedure

This section describes the time integration techniques and the temporal differencing scheme for the governing equations presented in Section 2.1. The solution procedures of the incompressible Navier-Stokes equations and the energy equation are treated separately in the sections that follow.

First-order forward differences are used to approximate all time derivatives. The time differencing is performed with respect to a set of discrete time values t^n separated by a variable time increment, $\Delta t = t^{n+1} - t^n$. The time level n is used as a superscript to denote the time level of a quantity q. Thus, q^n is the finite difference approximation to the quantity q at time t^n.

In the following sections, the temporal differencing scheme is illustrated while spatial differencing is suppressed for clari.y. Note that all spatial derivatives should be replaced with their finite difference approximations as described in Section 2.3.

2.2.1 Incompressible Navier-Stokes Equations

The code solves a fully implicit time discretization of the incompressible Navier-Stokes equations, namely

$$\nabla \cdot \mathbf{u}^{n+1} = 0 \tag{15}$$

$$\frac{\mathbf{u}^{n+1} - \mathbf{u}^n}{\Delta t} + (\nabla \cdot (\mathbf{uu}))^{n+1} = - \nabla p^{n+1} - \beta \mathbf{g}(T^{n+1} - T_o) + \nabla \cdot \sigma^{n+1}. \tag{16}$$

The scheme of Equations (15) and (16) constitutes a nonlinear algebraic system of equations for the advanced-time quantities p^{n+1} and $\mathbf{u}^{n+1}$. This system is solved by a new hybrid iteration scheme[a] which is essentially a generalization of the damped artificial compressibility method for steady-state incompressible flow[3] to the fully implicit transient equations. This scheme may be written in the form

[a] Unpublished research results of J. D. Ramshaw and P. R. McHugh entitled "Hybrid Iteration Scheme for Implicit Calculations of Incompressible Flow."

$$\frac{p^{(k+1)}-p^{(k)}}{\Delta t} = -\,a^2\,\nabla\cdot\mathbf{u}^{(k)} \tag{17}$$

$$\frac{\mathbf{u}^{(k+1)}-\mathbf{u}^{(k)}}{\Delta t} + \frac{\mathbf{u}^{(k+1)}-\mathbf{u}^{n}}{\Delta t} = -\,(\nabla\cdot(\mathbf{uu}))^{(k)} - \nabla\left(p^{(k+1)} - b\nabla\cdot\mathbf{u}^{(k)}\right)$$

$$-\,\beta\,\mathbf{g}(T^{\,n+1} - T_o) + \nabla\cdot\sigma^{(k)} \tag{18}$$

where $q^{(k)}$ denotes the approximation to q^{n+1} after the k^{th} iteration. The parameter *a* is an artificial sound speed, *b* is an artificial bulk viscosity, and $\Delta\tau$ is the artificial or pseudo-time step. These parameters are constrained by the stability restrictions

$$\frac{2\nu\Delta\tau}{\Delta^2} - \frac{\Delta\tau}{2\Delta\tau} < 1 \tag{19}$$

$$\frac{\Delta\tau(\alpha^2\Delta\tau+2b+2\nu)}{\Delta^2} - \frac{\Delta\tau}{2\Delta t} < 1 \tag{20}$$

$$b + \nu + \frac{\Delta^2}{4\Delta t} > 0 \tag{21}$$

where

$$\Delta^2 = \left\{ \frac{1}{\Delta x^2} + \frac{1}{\Delta y^2} + \frac{1}{\Delta z^2} \right\}^{-1} . \tag{22}$$

Here Δx, Δy, and Δz are the spatial increments in the *x*-, *y*-, and *z*-directions respectively, with the understanding that $\Delta z = \infty$ in two dimensions. Notice that *b* itself need not be positive. These results were derived by a linear stability analysis of the differenced equations neglecting the convection terms. The latter impose an additional material Courant restriction on $\Delta\tau$ of the form

$$\Delta\tau < min\left\{ \frac{\Delta x}{|u|}, \frac{\Delta y}{|v|}, \frac{\Delta z}{|w|} \right\} \tag{23}$$

where *u*, *v*, and *w* are respectively the *x*-, *y*-, and *z*-components of **u**, and the *min* operation includes a search over all cells in the computing mesh. In the derivation of the stability restrictions,

ν was assumed constant in the continuous equations, resulting in considerable simplification of the viscous terms. If ν is assumed variable, then additional terms arise which make the stability conditions more restrictive by replacing ν with 2ν in Equations (20) and (21).

The proper selection of the parameters is somewhat more difficult than in the analogous steady-state method,[4] because the character and behavior of the system now depend on the additional parameter Δt. In particular, care must be taken to ensure that the hyperbolic character of the system is not lost due to the additional damping introduced by the time derivative terms. This consideration, together with an analysis of the net damping rate, leads to the expressions[b]

$$a^2 = \frac{1.1}{4}\left(\frac{L_c^2}{\pi^2 \Delta t^2} + \frac{\pi^2 \nu^2}{L_c^2} + \frac{2\nu}{\Delta t}\right) \tag{24}$$

$$\Delta\tau = \alpha \left\{ \begin{array}{ll} min\left\{ \left[\frac{2\nu}{\Delta^2} - \frac{1}{2\Delta t}\right]^{-1}, \Delta\tau^* \right\}, & \Delta t > \Delta^2/(4\nu) \\ \Delta\tau^* & , \quad \Delta t < \Delta^2/(4\nu) \end{array} \right\} \tag{25}$$

$$b = \frac{\Delta^2}{4\Delta t}\left[2\omega\left(1 + \frac{\Delta t}{\Delta\tau}\right) - 1\right] - a^2\Delta\tau - \nu \tag{26}$$

where

$$\Delta\tau^* = \frac{\omega\Delta^2}{4a^2\Delta t}\left[1 + \left(1 + \frac{8a^2\Delta t^2}{\omega\Delta^2}\right)^{\frac{1}{2}}\right]. \tag{27}$$

L_C is a length typical of the overall dimensions of the flow region, α is a safety factor on the pseudo-time step ($0 < \alpha < 1$), and ω is a dimensionless damping coefficient of order unity. For Reynolds numbers near 0.1 the values $\alpha = 0.9$ and $\omega = 0.124$ appear to be nearly optimal. At higher Reynolds numbers smaller values of α are needed to ensure compliance with the convective stability limit of Equation (23).

[b] Unpublished research results of J. D. Ramshaw and P. R. McHugh entitled "Hybrid Iteration Scheme for Implicit Calculations of Incompressible Flow.

The scheme must also be supplemented by convergence criteria to determine when the iteration has converged to satisfactory accuracy. At present these criteria are taken to be

$$\max |\nabla \cdot \mathbf{u}^{(k+1)}| < \varepsilon_1 \frac{|\mathbf{u}|_{max}}{L_C} \tag{28}$$

$$\frac{1}{\Delta\tau} \max\{|u^{(k+1)}-u^{(k)}|, |v^{(k+1)}-v^{(k)}|, |w^{(k+1)}-w^{(k)}|\} < \varepsilon_2 \frac{|\mathbf{u}|^2_{max}}{L_C} \tag{29}$$

where the *max* operations include a search over all cells in the computing mesh. We currently take $\varepsilon_1 = 10^{-5}$ and $\varepsilon_2 = 10^{-3}$.

MAGMA also allows a linearly implicit option in the temporal differencing of the convection terms. This option consists of linearizing the convective terms by replacing $\nabla \cdot (\mathbf{uu})^{(k)}$ in Equation (18) with $\nabla \cdot (\mathbf{u}^n \mathbf{u}^{(k)})$, where n is the time level and k is the iterate level. In addition, a semi-implicit solution algorithm can be invoked by evaluating all terms on the right hand side of Equation (18) except pressure and temperature at the old time value, n.

2.2.2 Energy Equation

The code solves an explicit time discretization of the energy conservation equation,

$$\rho\left(\frac{e^{n+1} - e^n}{\Delta t} + \nabla\bullet(\mathbf{u}e)^n\right) = \nabla\cdot(k\ \nabla\ T)^n + q \tag{30}$$

Cell energies are marched forward explicitly in time to the new time level using only variables from the previous time level. The time step is therefore limited by the diffusive and Courant stability restrictions,

$$\Delta t < \frac{\Delta^2}{2\alpha} \tag{31}$$

$$\Delta t < min\left\{\frac{\Delta x}{|u|}, \frac{\Delta y}{|v|}, \frac{\Delta z}{|w|}\right\} \tag{32}$$

where

$$\alpha = \frac{k}{\rho c} \quad \text{(thermal diffusivity)} \tag{33}$$

and c is the specific heat capacity. If necessary, the code resets Δt after each time step in order to satisfy the changing Courant stability restriction.

It would of course be preferable to use a fully implicit scheme for the energy equation as well, thereby removing the stability restrictions of Equations (31) and (32). This scheme should be replaced by a fully implicit scheme if the model is developed further.

2.3 Spatial Differencing

The governing equations are spatially differenced using the control volume or finite volume approach, whereby the temporally differenced conservation equations are averaged over the volume of the appropriate computational cell.[4] The divergence theorem is used to convert the volume averages of terms of divergence form to surface integrals over cell faces. This procedure leads to conservative difference equations for mass, momentum, and energy.[4] The spatial differencing is conventional in all respects. All terms other than convection are approximated by centered differences, while convection terms are approximated by weighted averages of centered and upwind (donor cell) differences (see Appendix B).

2.3.1 Finite Difference Mesh

MAGMA numerically solves differenced forms of the coupled governing equations on a two- or three-dimensional, staggered grid using two- or three-dimensional rectangular coordinates or two-dimensional cylindrical coordinates. The computational mesh consists of rectangular cells of width Δx, length Δy, and height Δz. The mesh contains *imax* cells in the x-direction, *jmax* cells in the y-direction, and *kmax* cells in the z-direction.

Variables in a computational cell are placed according to the MAC solution procedure.[2] Velocity components are located on cell faces, while pressures, temperatures, energies, and fluid properties are located at cell centers. Figure 2 is a schematic illustration of a computational cell. Note the terminology used in this figure to refer to cell faces. For example, the velocity component in the x-direction (u) is located on the *right* cell face, the y-direction velocity component (v) on the *front* cell face, and the z-direction velocity component (w) on the *top* cell face. The cell faces opposing these three sides are the *left*, *derriere*, and *bottom* cell faces, respectively. The mesh contains both active cells containing fluid and inactive solid cells, as shown in Figure 3. Cells not containing fluid (inactive cells) are represented by shaded cells. Boundary cells are used to impose the velocity and temperature boundary conditions. Velocity boundary cells, which are marked with the letters *VBC* in Figure 3, are always adjacent to cells that contain fluid. Temperature boundary cells, which are marked with the letters *TBC* in Figure 3, are always located at the edges, or perimeter, of the computational region. Three-dimensional rectangular coordinates is the code default. Two-dimensional rectangular coordinates and two-dimensional cylindrical coordinates are invoked by setting the spatial increment in the z-direction to a negative value ($\Delta z < 0$). This option forces all terms associated with the z-coordinate to vanish in the discretized form of the governing equations (formally $\Delta z \rightarrow \infty$). Two-dimensional cylindrical coordinates assume that the

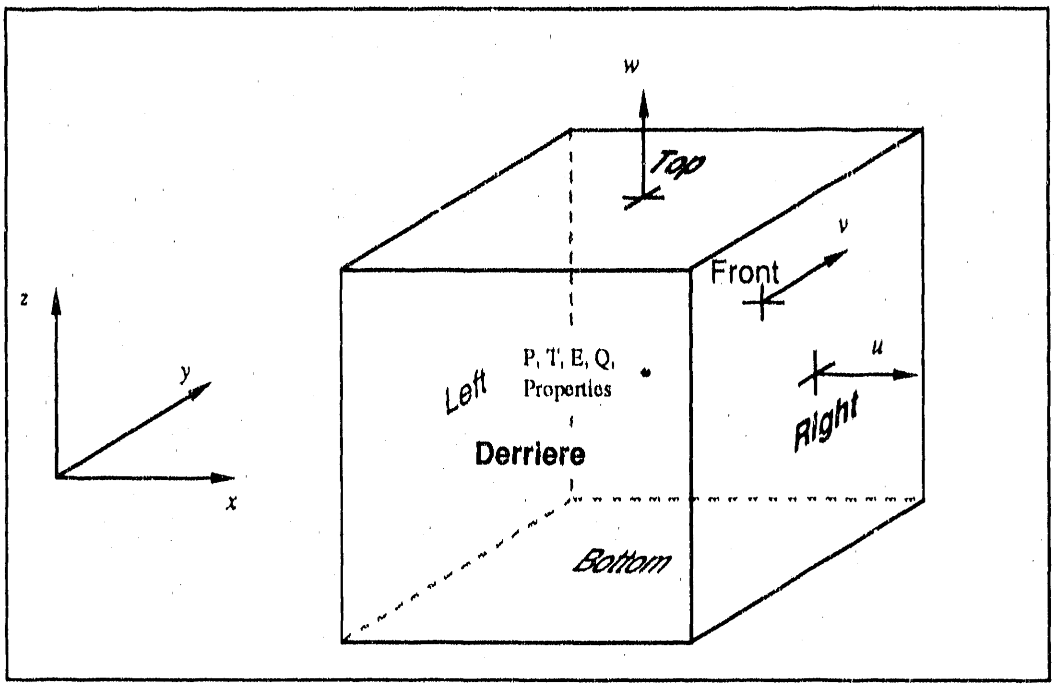
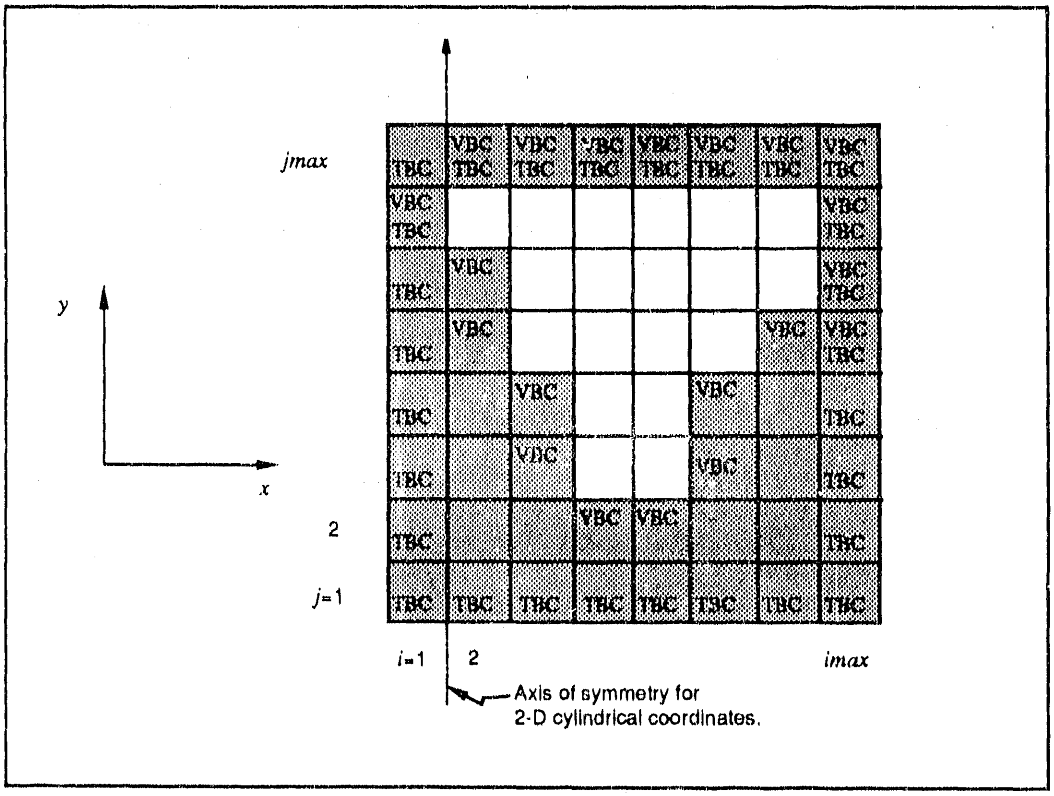


Figure 2. Schematic of computational cell (*i,j,k*).



jmax	TBC	VBC TBC	VBC TBC	'JBC TBC	VBC TBC	VBC TBC	VBC TBC	VBC TBC
	VBC TBC							VBC TBC
	TBC	VBC						VBC TBC
	TBC	VBC					VBC	VBC TBC
	TBC		VBC			VBC		TBC
	TBC		VDC			VBC		TBC
2	TBC			VBC	VBC			TBC
j=1	TBC	TBC	TBC	TBC	TBC	TBC	TBC	TBC
	i=1	2						*imax*

Figure 3. *X*-*y* cross section of computational region.

y-coordinate axis is the axis of symmetry, x is the radial coordinate, and all terms associated with the z-coordinate vanish. These cylindrical terms in the governing equations are included in the formulation when the cylindrical coordinates flag, ξ (CYL in the code), is set to one.

Cell spacing in all coordinate directions is assumed uniform. Simple averaging is used to obtain quantities at points on the mesh where they are not explicitly defined. The averaging procedure is described in Appendix A.

As shown in Figure 2, velocities are located on cell faces, i.e.,

$$u_{ijk} = u(i\Delta x,(j - \frac{1}{2})\Delta y,(k - \frac{1}{2})\Delta z) \qquad (34)$$

$$v_{ijk} = v[(i - \frac{1}{2})\Delta x, j\Delta y,(k - \frac{1}{2})\Delta z)] \qquad (35)$$

$$w_{ijk} = w[(i - \frac{1}{2})\Delta x,(j - \frac{1}{2})\Delta y, k\Delta z)] \qquad (36)$$

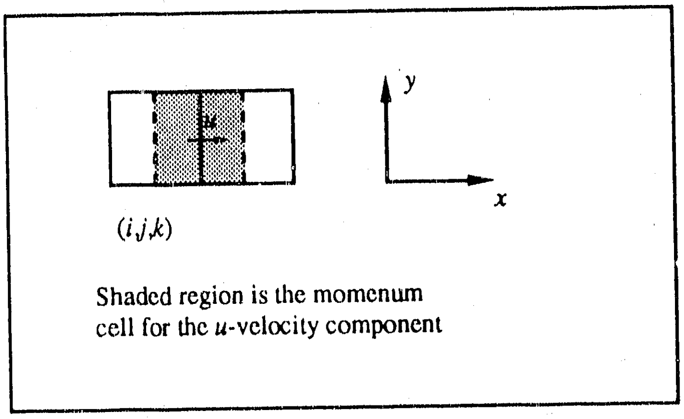
All other quantities are located at cell centers. The location of the cell centered quantity Q_{ijk}, is given by

$$Q_{ijk} = Q[(i - \frac{1}{2})\Delta x,(j - \frac{1}{2})\Delta y,(k - \frac{1}{2})\Delta z)] \qquad (37)$$

The volume, V, of a computational cell (i,j,k) is simply,

$$V = \Delta x \Delta y \Delta z. \qquad (38)$$

It is convenient to define three additional types of cells, called *momentum cells*, which are centered about the locations of the velocity components. Figure 4 illustrates the momentum cell corresponding to the u-velocity component. Analogous momentum cells exist for the v and w velocity components. In the notation of subsequent sections, the momentum cells corresponding to the u-, v-, and w-velocity components are referred to by superscripts m_u, m_v, and m_w respectively.



Shaded region is the momenum cell for the u-velocity component

Figure 4. Schematic illustration of a *momentum cell* corresponding to the u-velocity component.

2.3.2 Differencing Notation

Following the notation of Reference 4, volume averages are denoted by angle brackets. The average of a quantity, Q, over a typical cell (i,j,k) is given by

$$< Q >_{ijk} = \frac{1}{V} \int \int \int Q(x,y,z)\ dxdydz \tag{39}$$

where the volume integral is taken over that cell. Similarly, the average of a quantity over each of the three momentum cells is denoted by

$$< Q >_{ijk}^{m_u} = \frac{1}{V} \int \int \int Q(x,y,z)\ dxdydz \tag{40}$$

$$< Q >_{ijk}^{m_v} = \frac{1}{V} \int \int \int Q(x,y,z)\ dxdydz \tag{41}$$

$$< Q >_{ijk}^{m_w} = \frac{1}{V} \int \int \int Q(x,y,z)\ dxdydz \tag{42}$$

where the volume integrals are taken over the indicated momentum cell. If the quantity Q is not of divergence form, these averages are approximated by the values at cell centers; i.e.,

$$< Q >_{ijk} = Q((i - \frac{1}{2})\Delta x,(j - \frac{1}{2})\Delta y,(k - \frac{1}{2})\Delta z) \tag{43}$$

$$< Q >_{ijk}^{m_u} = Q(i\Delta x,(j - \frac{1}{2})\Delta y,(k - \frac{1}{2})\Delta z) \tag{44}$$

$$< Q >_{ijk}^{m_v} = Q((i - \frac{1}{2})\Delta x,j\Delta y,(k - \frac{1}{2})\Delta z)) \tag{45}$$

$$< Q >_{ijk}^{m_w} = Q((i - \frac{1}{2})\Delta x,(j - \frac{1}{2})\Delta y,k\Delta z)) \tag{46}$$

where the quantity $Q(x,y,z)$ is evaluated at the point (x,y,z) using whatever averaging is necessary. The averaging procedure used by MAGMA is described in Appendix A.

If the quantity Q is of divergence form, i.e., $Q = \nabla\bullet\mathbf{W}$, where $\mathbf{W}$ is a vector or tensor quantity, then the volume integrals in Equations (39) through (42) are converted to surface integrals using the divergence theorem,

$$< \nabla\bullet\mathbf{W} >_{ijk} = \frac{1}{V} \oint dA\ \mathbf{n}\bullet\mathbf{W} \tag{47}$$

where $\mathbf{n}$ is the outward unit normal to the surface and $\oint dA$ is the surface integral over that surface. Equation (47) can be approximated by

$$< \nabla\bullet\mathbf{W} >_{ijk} = \frac{1}{V} \sum_{\alpha}(\Delta A\ \mathbf{n})_{\alpha}\bullet\mathbf{W}_{\alpha} \tag{48}$$

where the subscript a refers to the faces of the cell (i,j,k). Equation (48) can be expanded as

$$< \nabla \bullet \mathbf{w} >_{ijk} = \frac{1}{V} [\ (\Delta A\ \mathbf{n})_B \bullet \mathbf{w}_B + (\Delta A\ \mathbf{n})_D \bullet \mathbf{w}_D$$

$$+ (\Delta A\ \mathbf{n})_F \bullet \mathbf{w}_F + (\Delta A\ \mathbf{n})_L \bullet \mathbf{w}_L$$

$$+ (\Delta A\ \mathbf{n})_R \bullet \mathbf{w}_R + (\Delta A\ \mathbf{n})_T \bullet \mathbf{w}_T \ ] \tag{49}$$

where the subscripts B, D, F, L, R, and T refer to the bottom, derriere, front, left, right, and top sides, respectively, of the computational cell (i,j,k). The quantities in parentheses in Equation (49) are defined as follows:

$$(\Delta A\ \mathbf{n})_B = -\mathbf{e}_z\ \Delta x \Delta y \tag{50}$$

$$(\Delta A\ \mathbf{n})_D = -\mathbf{e}_y\ \Delta x \Delta z \tag{51}$$

$$(\Delta A\ \mathbf{n})_F = \ \mathbf{e}_y\ \Delta x \Delta z \tag{52}$$

$$(\Delta A\ \mathbf{n})_L = -\mathbf{e}_x\ \Delta y \Delta z \tag{53}$$

$$(\Delta A\ \mathbf{n})_R = \ \mathbf{e}_x\ \Delta y \Delta z \tag{54}$$

$$(\Delta A\ \mathbf{n})_T = \ \mathbf{e}_z\ \Delta x \Delta y \tag{55}$$

where $\mathbf{e}_x$, $\mathbf{e}_y$, and $\mathbf{e}_z$ are unit vectors in the x, y, and z coordinate directions, respectively. Similar expressions hold for the volume averaged quantities integrated over the three momentum cells. In summation form these expressions are given by

$$< \nabla \bullet \mathbf{w} >_{ijk}^{mq} = \frac{1}{V} \sum_{\alpha_q} (\Delta A\ \mathbf{n})_{\alpha_q} \bullet \mathbf{w}_{\alpha_q} \tag{56}$$

where the subscript α_q (α_q = B, D, F, L, R, or T) refers to the faces of the momentum cell denoted by mq, with q equal to u, v, or w.

A special case arises when Q represents convective terms in either the momentum equation or the energy equation (i.e., $Q = \nabla\bullet[\mathbf{uu}]$ or $Q = \nabla\bullet[\mathbf{u}T]$). As an example, let $Q = \nabla\bullet[\mathbf{u}S]$ where S is the quantity being convected by the velocity, $\mathbf{u}$. The volume averaged expression for Q is given by

$$< \nabla\bullet[\mathbf{u}S] >_{ijk} = \frac{1}{V}\sum_{\alpha_q}(\Delta A\ \mathbf{n})_{\alpha_q}\bullet S_a. \tag{57}$$

Note that if $S = T$, the summation is performed over the regular cell (i,j,k); however, if $S = u$, v, or w, the summation is performed over the corresponding momentum cell m_q with $q = u$, v, or w. A differencing technique based on a weighted average of centered and donor cell differencing is used in conjunction with the convection terms to ensure numerical stability. The value of $(\Delta A\ \mathbf{n}\bullet\mathbf{u})_{\alpha_q}$ is computed as a centered average on side α, while the quantity being convected, S_α, is computed as a weighted average of its centered and its "upwinded" values as described in Appendix B.

2.3.3 Incompressible Navier-Stokes Equations

The spatial differencing for Equations (17) and (18) is given below, using the expanded forms of the equations in the x-, y-, and z-coordinate directions. The additional terms required for two-dimensional cylindrical coordinates are explicitly included in this discussion. Recall that terms associated with the z-coordinate direction vanish in two dimensions.

- Continuity equation

$$\frac{p^{(k+1)}_{ijk} - p^{(k)}_{ijk}}{\Delta\tau} = -a^2\left[<\nabla\bullet\mathbf{u}^{(k)}>_{ijk} + \xi\frac{<u^{(k)}>_{ijk}}{(i - 1/2)\Delta x}\right] \tag{58}$$

- Momentum equation in x-direction

$$\frac{u_{ijk}^{(k+1)} - u_{ijk}^{(k)}}{\Delta\tau} = -\frac{u_{ijk}^{(k+1)} - u_{ijk}^{n}}{\Delta t} - <\nabla\cdot[\mathbf{u}u]^{(k)}>_{ijk}^{m_u} - \xi\frac{<u^2>_{ijk}^{m_u}}{i\Delta x}$$

$$- <\nabla\cdot\left[\mathbf{e}_x\left(p^{(k)} - b<\nabla\cdot\mathbf{u}^{(k)}>_{ijk}\right)\right]>_{ijk}^{m_u}$$

$$- \beta\,(\mathbf{g}\cdot\mathbf{e}_x)<T^{n+1} - T_o>_{ijk}^{m_u} + \frac{<\nabla\cdot\left((x)^{\xi}\sigma_x\right)>_{ijk}^{m_u}}{(i\Delta x)^{\xi}}$$

$$- \frac{2\,\nu\xi<u^2>_{ijk}^{m_u}}{(i\Delta x)^2} \tag{59}$$

- Momentum equation in y-direction

$$\frac{v_{ijk}^{(k+1)} - v_{ijk}^{(k)}}{\Delta\tau} = -\frac{v_{ijk}^{(k+1)} - v_{ijk}^{n}}{\Delta t} - <\nabla\cdot[\mathbf{u}v]^{(k)}>_{ijk}^{m_v} - \xi\frac{<uv>_{ijk}^{m_v}}{(i - 1/2)\Delta x}$$

$$- <\nabla\cdot\left[\mathbf{e}_y\left(p^{(k)} - b<\nabla\cdot\mathbf{u}^{(k)}>_{ijk}\right)\right]>_{ijk}^{m_v}$$

$$- \beta\,(\mathbf{g}\cdot\mathbf{e}_y)<T^{n+1} - T_o>_{ijk}^{m_v}$$

$$+ \frac{<\nabla\cdot\left((x)^{\xi}\sigma_y\right)>_{ijk}^{m_v}}{\left((i - 1/2)\Delta x\right)^{\xi}} \tag{60}$$

- Momentum equation in z-direction

$$\frac{w^{(k+1)}_{ijk} - w^{(k)}_{ijk}}{\Delta \tau} = -\frac{w^{(k+1)}_{ijk} - w^{n}_{ijk}}{\Delta t} - < \nabla \cdot [\mathbf{u} w]^{(k)} >^{m_w}_{ijk}$$

$$- < \nabla \cdot \left[\mathbf{e}_z \left(p^{(k)} - b < \nabla \cdot \mathbf{u}^{(k)} >_{ijk} \right) \right] >^{m_w}_{ijk}$$

$$- \beta \, (\mathbf{g} \cdot \mathbf{e}_z) < T^{n+1} - T_o >^{m_w}_{ijk} + < \nabla \cdot \sigma_z >^{m_w}_{ijk} \tag{61}$$

where k is the iteration index, and the gradient operator, ∇, is expressed in rectangular coordinates.

Recall from Section 2.2 that if the linearly implicit option is specified, the convection terms, $< \nabla \cdot [\mathbf{u} q]^{(k)} >^{m_q}_{ijk}$, are replaced by $< \nabla \cdot [\mathbf{u}^{n} q^{(k)}] >^{m_q}_{ijk}$ ($q = u$, v, or w), where n is the time level.

2.3.4 Energy Equation

Equation (30) is the temporally differenced energy equation. The spatial differencing for this equation is given in Equation (62) below.

$$\rho \left(\frac{e^{n+1}_{ijk} - e^{n}_{ijk}}{\Delta t} + < \nabla \cdot (\mathbf{u} e)^{n} >_{ijk} + \frac{\xi < (e u)^{n} >_{ijk}}{(i - 1/2) \Delta x} \right) =$$

$$< \nabla \cdot (k \, \nabla T)^{n} >_{ijk} + \frac{\xi k}{(i - 1/2) \Delta x} < \nabla \cdot (\mathbf{e}_x T^{n}) >_{ijk} + q_{ijk} \tag{62}$$

where the gradient operator, ∇, is expressed in rectangular coordinates (Equation (11)).

2.4 Boundary Conditions

Mesh cells containing fluid are surrounded by solid or fictitious boundary cells that impose the problem boundary conditions. Boundaries may be positioned along cell faces or they may pass diagonally through the centers of cell faces. The following sections describe how MAGMA sets velocities and temperatures in boundary cells in order to impose the physical boundary conditions.

2.4.1 Velocity Boundary Conditions

Boundary conditions associated with velocity are set in inactive cells adjacent to at least one active cell. Cell position and orientation determine how boundary velocities are set. There are two possible orientations with regard to setting velocity boundary conditions. The first orientation is when a wall or boundary lies along a cell face, and the second orientation is when a wall cuts diagonally through cell faces. These orientations are discussed separately in the sections that follow.

2.4.1.1 Walls Along Cell Faces. Consider Figure 5 as an example. Note that the positive *z*-direction is out of the paper. An analogous situation exists if *y* and *v* are replaced by *z* and *w*, respectively, and the positive *y* direction is into the paper.

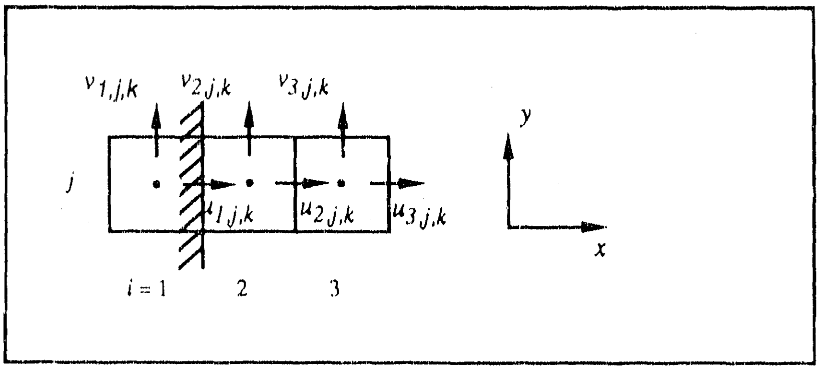


Figure 5. Wall boundary orientated along a cell face.

Physically, there can be no flow across a wall. Therefore, the normal velocity component on a wall is identically set to zero regardless of the wall type (free slip, no slip, etc...) i.e.,

$$u_{1\,j,k} = 0 \tag{63}$$

Physical characteristics of a free slip wall, a no-slip wall, or a wall with a known velocity determine how tangential velocity components are set in a velocity boundary cell. At a free slip wall, there is a zero shear stress, so the normal derivative of the tangential velocity must vanish. Thus, the tangential velocity component in the boundary cell is set equal to the tangential velocity component in the adjacent active cell. In Figure 5 this condition is satisfied by

$$v_{1\,j,k} = v_{2\,j,k}\,. \tag{64}$$

No-slip boundary conditions require that tangential velocity components vanish at the wall. The tangential velocity in the boundary cell is set in order to satisfy this condition and in order to guarantee the finite difference approximation to the second normal derivative at the wall is consistent with a second order Taylor series approximation. For example, at cell (2,*j*,*k*) in Figure 5 the code approximates the second derivative of *v* in the *x*-direction as,

$$\left(\frac{\partial^2 v}{\partial x^2}\right)_{\text{code}} = \frac{(v_{3\,j,k} - 2v_{2\,j,k} + v_{1\,j,k})}{\Delta x^2} \tag{65}$$

The Taylor series approximation to this quantity is found by expanding $v_{3\,j,k}$ and the velocity at the wall about the point (2,*j*,*k*)

$$v_{3,j,k} = v_{2\,j,k} + \Delta x\,\frac{\partial v}{\partial x} + \frac{\Delta x^2}{2}\,\frac{\partial^2 v}{\partial x^2} + \cdots \tag{66}$$

$$v_{wall} = 0 = v_{2\,j,k} - \frac{\Delta x}{2}\,\frac{\partial v}{\partial x} + \frac{\Delta x^2}{8}\,\frac{\partial^2 v}{\partial x^2} + \cdots \tag{67}$$

Neglecting higher order terms and solving for the second derivative gives

$$\left(\frac{\partial^2 v}{\partial x^2}\right)_{\text{Taylor series}} = \frac{4}{3}\,\frac{(v_{3\,j,k} - 3v_{2\,j,k})}{\Delta x^2} \tag{68}$$

Equating Equations (65) and (68) and solving for $v_{1j,k}$ gives,

$$v_{1j,k} = \frac{1}{3} v_{3j,k} - 2\, v_{2j,k} \tag{69}$$

This expression satisfies the condition of zero tangential velocity at the wall as well as guaranting a consistent second order approximation to the second normal derivative. Analogous expressions arise for boundaries located on the other cell faces.

Specified velocity boundary conditions are easily imposed if the velocity components are located on the boundary. However, when this is not the case the boundary velocity component is once again set in a consistent, second order manner. For example, if the tangential velocity on the boundary in Figure 5 is v_L, the tangential velocity component in the boundary cell is calculated from,

$$v_{1j,k} = \frac{1}{3} v_{3j,k} - 2\, v_{2j,k} + \frac{8}{3} v_L \tag{70}$$

Analogous expressions exist for other specified velocity boundary conditions of this type.

2.4.1.2 Walls Cutting Diagonally Through Cell Faces. No-slip walls passing diagonally through cell faces allow the simulation of non-rectangular geometries. Diagonal boundaries can cut through two or three cell faces. Boundary cell velocities located on the wall are identically set to zero. Diagonal boundaries passing through only two cell faces require setting a tangential velocity component not located on the wall. This situation is illustrated in Figure 6, in which the tangential velocity at point (1) (out of the paper) must be set in order to satisfy the no-slip requirement on the boundary. Note that the horizontal and vertical velocity components in cell (i,j,k) are identically set to zero because they are directly located on the boundary. The third velocity component at point (1) coming out of the paper is set in the following manner:

1. The velocity in this direction is computed at point (2) by averaging the corresponding velocity components in cells $(i,j+1,k)$ and $(i+1,j,k)$.

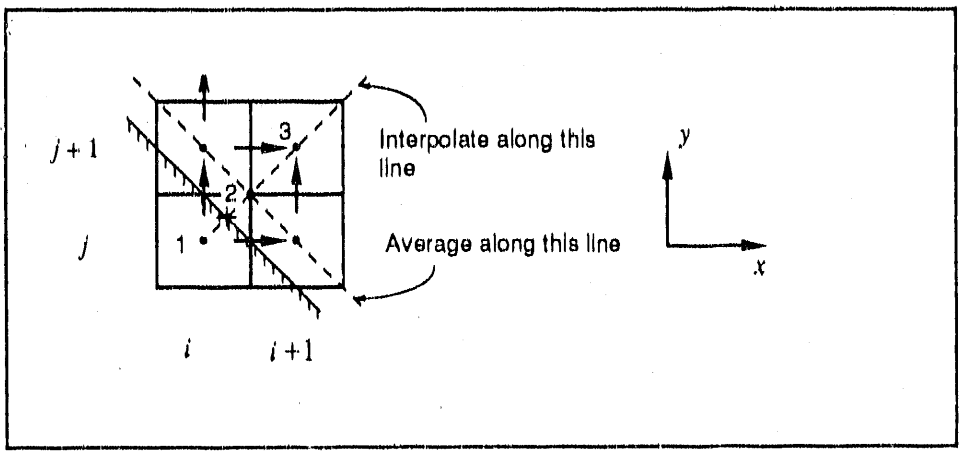


Figure 6. Diagonal wall boundary.

2. A second order polynomial is fitted through points (2) and (3), and the point indicated on the boundary in Figure 2.5.2. The velocity on the boundary is zero for this interpolation.

3. The velocity at point (1) is calculated from this second order polynomial equation.

This procedure results in the following expression for the tangential velocity component, *w*, in cell (*i*,*j*,*k*)

$$w_{i,j,k} = \frac{1}{3} w_{i+1,j+1,k} - w_{i+1,j,k} - w_{i,j+1,k} \qquad (71)$$

Analogous expressions arise when other diagonal boundary orientations are encountered.

2.4.2 Temperature Boundary Conditions

Temperature boundary conditions are imposed on the perimeter of the computational region. Four types of temperature boundary conditions are allowed: convective boundary conditions, constant heat flux, constant temperature, and adiabatic (zero heat flux) boundary conditions.[5]

For convective/mixed type boundary conditions, the boundary temperature is determined from an energy balance at the wall, where the heat transfer by convection equals the heat transfer by conduction at the wall. On the front boundary this condition is expressed as,[5]

$$q'' = h(T_w - T_\infty) = -k\left(\frac{dT}{dy}\right)_w = -k\frac{(T_{i,jmax,k} - T_{i,jmax-1,k})}{\Delta x} \tag{72}$$

where

h = convective heat transfer coefficient

k = thermal conductivity

q'' = heat flux

subscripts:

w = wall value

∞ = value of surroundings.

T_w is approximated by

$$T_w = \frac{1}{2}(T_{i,jmax,k} + T_{i,jmax-1,k}) \tag{73}$$

and the value of the thermal conductivity on the wall is taken to be the value at the center of the cell adjacent to the wall. The convective-mixed type boundary condition on the front wall is then satisfied by

$$T_{i,jmax,k} = \frac{\dfrac{2h\Delta y}{k_{i,jmax-1,k}}T_\infty + \left(2 - \dfrac{h\Delta y}{k_{i,jmax-1,k}}\right)T_{i,jmax-1,k}}{2 + \dfrac{h\Delta y}{k_{i,jmax-1,k}}} \tag{74}$$

Expressions satisfying convective/mixed boundary conditions on the bottom, derriere, left, right, and top sides, respectively, are completely analogous to Equation (74).

For a constant heat flux type boundary condition, the boundary temperature is once again determined from an energy balance at the wall. As an example, consider the case where a constant heat flux, q_L'', exists across the left boundary of the computational region.[5] The energy balance at the wall requires

$$q_L'' = -k\left(\frac{dT}{dx}\right)_{x=0} = -k_{2\,j,k}\frac{(T_{2\,j,k} - T_{1\,j,k})}{\Delta x} \tag{75}$$

or

$$T_{1\,j,k} = T_{2\,j,k} + \frac{q_L''\Delta x}{k_{2\,j,k}} \tag{76}$$

Analogous expressions can be derived for constant heat flux conditions along the bottom, derriere, front, right, and top walls, respectively. The adiabatic boundary condition is imposed by simply setting the heat flux across that wall to zero.

Constant temperature boundary conditions are easily imposed if the temperature is located on the boundary. However, when the boundary coincides with a cell face, the boundary temperature is set in a consistent second order manner similar to the method described for specified velocity boundary conditions. A constant temperature (T_B) along the bottom wall requires the boundary cell temperature be calculated from the expression given in Equation (77). Expressions for the constant temperature boundary conditions along the remaining walls are completely analogous to Equation (77).

$$T_{i,j,1} = \frac{1}{3}T_{i,j,3} - 2\,T_{i,j,2} + \frac{8}{3}T_B \tag{77}$$

2.4.3 Boundary Conditions Summary

Velocity boundary conditions are imposed after each iteration in the numerical solution to the momentum equation. Velocity boundaries are not restricted to the perimeter of the computational region. Velocity boundaries can lie along or pass diagonally through cell faces. Code logic identifies the type of boundary orientation and sets the boundary velocities accordingly.

Types of allowable velocity boundary conditions include:

- No-slip
- Free slip
- Specified velocity.

Temperature boundary conditions are imposed after each time step on the perimeter of the computational region. Types of allowable temperature boundary conditions include:

- Constant temperature
- Constant heat flux
- Symmetric
- Convective/mixed.

3. CODE DESCRIPTION

MAGMA is a FORTRAN computer code. Important aspects of the MAGMA code are discussed below including:

- Indirect addressing scheme
- General logic flow
- Subroutines and variables used by the code
- Code input and options
- Code output.

A variable dictionary is given in Appendix C, and a listing of a sample input data file is given in Appendix D.

3.1 Indirect Addressing Scheme

The indirect addressing scheme used by MAGMA is designed to avoid costly and unnecessary computation over inactive cells in the computational domain. By defining indirect addressing arrays the solution algorithm can sweep only over active cells in the mesh. This scheme is used in conjunction with the solution of the incompressible Navier-Stokes equations, but not the solution of the energy equation. Thus, heat flow is computed throughout the entire region, while fluid flow is computed only in molten cells.

IMAP is a four dimensional integer array. It is used to specify whether computational cells are active or inactive. And in the case of velocity boundary cells, the type of velocity boundary condition to be used. The first three indices of *IMAP* correspond to the indices of the computational grid. If the fourth index is 1, the information stored in that array element determines whether cell (*i,j,k*) is active or inactive. For example, if cell (*i,j,k*) is an active cell then *IMAP(I,J,K,1)* = 1. Conversely, if cell (*i,j,k*) is an inactive cell then *IMAP(I,J,K,1)* = 0. If the fourth index is 2, then the information stored in that array element determines the type of velocity boundary condition to be used in cell (*i,j,k*). For example, if *IMAP(I,J,K,2)* = 0, then the no-slip boundary condition is enforced in cell (*i,j,k*). Similarly, *IMAP(I,J,K,2)* = 1 indicates that the free slip boundary condition is applied in cell (*i,j,k*), while *IMAP(I,J,K,2)* = 2 indicates that the specified type velocity boundary condition is applied in cell (*i,j,k*). The use of *IMAP* in the boundary condition logic requires: *I* to range between 0 and *IMAX* + 1, *J* to range between 0 and *JMAX* +1, and *K* to range between 0 and *KMAX* + 1.

A two-dimensional integer array (*ID*) is used to keep track of all the active cells in the region. The active cells are numbered consecutively using the *IMAP* array, sweeping through the mesh from left to right, front to derriere, and bottom to top, respectively. The *i*, *j*, and *k* indices of each active cell are stored along with its cell number in order to keep track of cell position and relation to neighboring cells. The number of rows in this array is three, while the number of columns is the number of active cells (*NCELLS*). The column number is also the number of an active cell. Therefore, for each active cell (each column) the first row contains the *i* coordinate index, the second row the *j* coordinate index, and the third row the *k* coordinate index.

A third integer array (*IDBC*) is used to keep track of the velocity boundary cells and to set the velocity boundary conditions. The format of this array is the same as that described above. The column number corresponds to the number of the velocity boundary cell (*NBC* columns). The three rows contain the *i*, *j*, and *k* coordinate indices corresponding to that velocity boundary cell.

3.2 Overall Program Flow Chart

The general logic flow of the MAGMA code is shown in Figure 7. Initially the code reads input data from either an input file or a binary restart file. The restart file is used to restart a previous calculation from the point of the last binary data dump. Program variables and constants are initialized following input.

Following initialization, the code tests an input flag to determine if the energy equation is to be solved. If this energy flag tests true, the physical time step and iteration parameters are reset based on energy equation stability restrictions. Cell energies are updated to the new time level and cell temperatures are computed using the equation of state. Temperature boundary conditions are set and temperature dependent fluid properties such as thermal conductivity, viscosity, and specific heat capacity are specified.

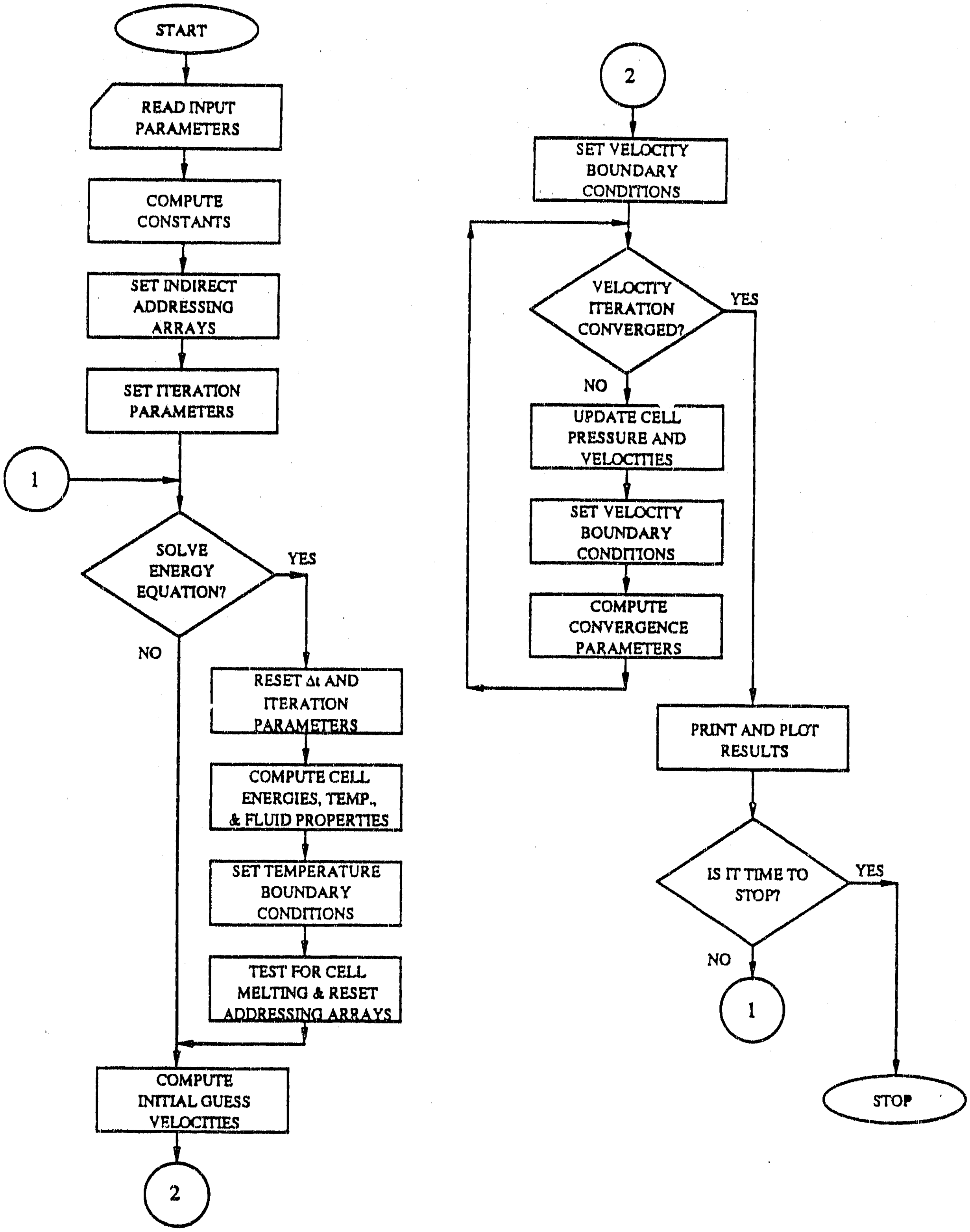

START
READ INPUT PARAMETERS
COMPUTE CONSTANTS
SET INDIRECT ADDRESSING ARRAYS
SET ITERATION PARAMETERS
1
SOLVE ENERGY EQUATION?
YES
NO
RESET Δt AND ITERATION PARAMETERS
COMPUTE CELL ENERGIES, TEMP., & FLUID PROPERTIES
SET TEMPERATURE BOUNDARY CONDITIONS
TEST FOR CELL MELTING & RESET ADDRESSING ARRAYS
COMPUTE INITIAL GUESS VELOCITIES
2
2
SET VELOCITY BOUNDARY CONDITIONS
VELOCITY ITERATION CONVERGED?
YES
NO
UPDATE CELL PRESSURE AND VELOCITIES
SET VELOCITY BOUNDARY CONDITIONS
COMPUTE CONVERGENCE PARAMETERS
PRINT AND PLOT RESULTS
IS IT TIME TO STOP?
YES
NO
1
STOP

Figure 7. Top level flow chart for the MAGMA code.

Next, the code computes initial guess velocities and proceeds to the iterative solution of the incompressible Navier-Stokes equations. Velocity boundary conditions are imposed following each iteration. The iteration continues until velocities and pressures have converged to the solution at the new time level. Data is written to data files and the code compares the new time level with the user specified ending time to determine whether to continue or terminate the run.

3.3 Subroutine Organization and Description

The following list defines the subroutines used in the MAGMA code. Primary functions of the subroutines are briefly described. Figure 8 presents a Hierarchical Table of Contents (HTOC), which demonstrates relationships existing between subroutines and the main code.

1. ADDRSS (ADDReSS)

   This subroutine defines the indirect addressing arrays which relate the cell numbering scheme to the physical coordinates of the interior and boundary cells of the computational region. It is called by MAIN.

2. ADVNCE (ADVaNCE)

   This subroutine updates all time dependent variables to the new time level once convergence has been obtained. It is called by MAIN.

3. BNDRY (BounDaRY)

   This subroutine sets the velocity boundary conditions. It is called by MAIN.

4. CONVRG (CONVeRGe)

   This subroutine checks if the iteration scheme solution has satisfied the convergence criteria. It is called by MAIN.

5. DUMP (DUMP)

   Performs a binary dump of program variables to a restart file. It is called by PRNT.

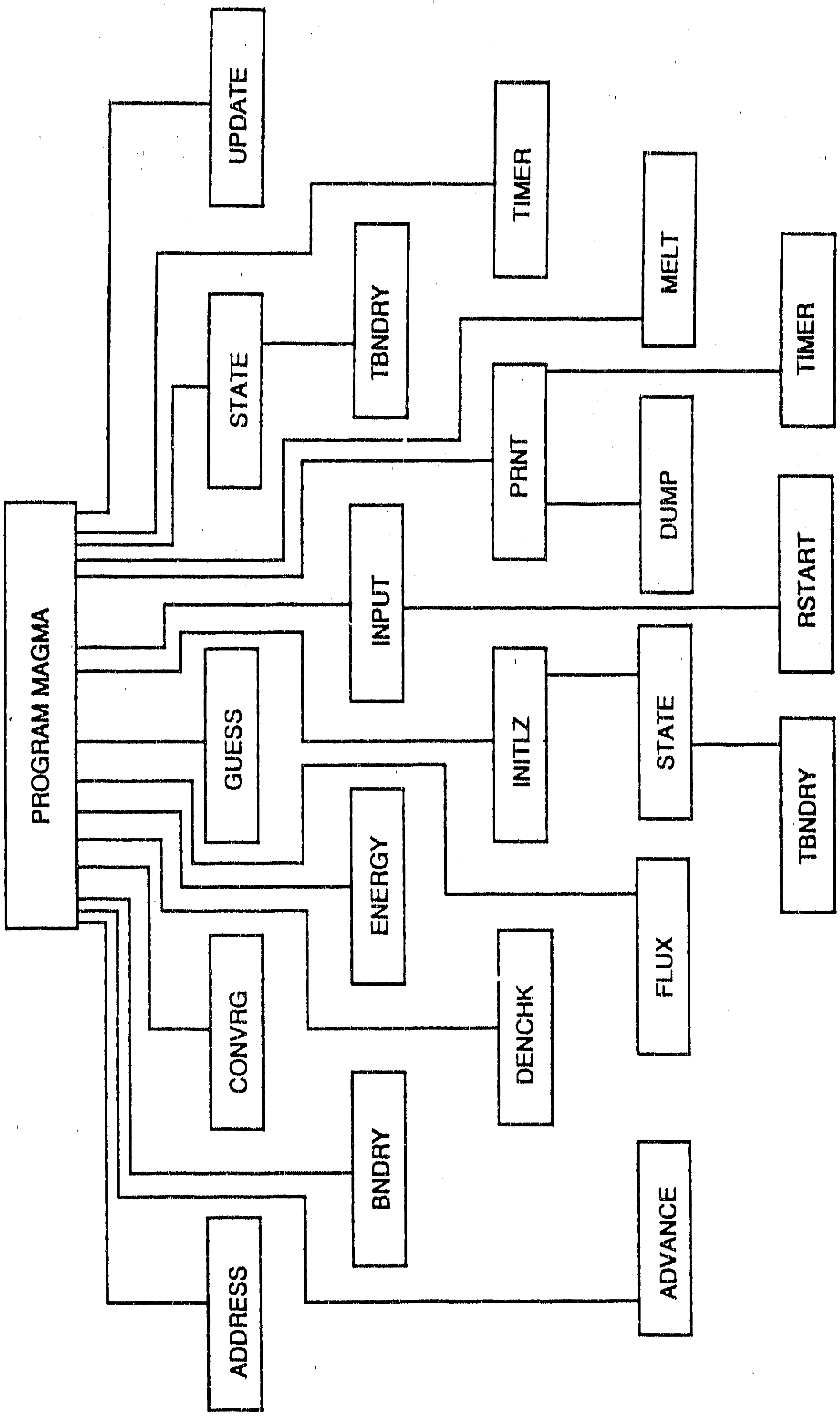


Figure 8. Subroutine hierarchical table of contents (HTOC).

6. ENERGY (ENERGY)

This subroutine solves the energy equation. The governing partial differential equation is solved explicitly in time as described in the main text.

7. FLUX (FLUX)

This subroutine computes the velocity changes due to the convective and viscous terms in the momentum equation. It is called by MAIN.

8. GUESS (GUESS)

This subroutine computes the initial guess for the velocity field required for the iteration scheme. It is called by MAIN.

9. INITLZ (INITiaLiZe)

This subroutine initializes constant terms which are used throughout the rest of the program. It also calls STATE to set the initial energy field from the initial temperature and to compute the initial temperature dependent properties. It is called by MAIN.

10. INPUT (INPUT)

This subroutine reads in the necessary input data. It is called by MAIN.

11. MELT (MELT)

This subroutine tests cell energy values against the material latent heat of fusion value to determine if cell melting or freezing should occur.

12. PRNT (PRiNT)

This subroutine outputs the results of the calculations. It is called by MAIN.

13. RSTART (Re START)

Reads program variables from a binary restart file. It is called by INPUT.

14. STATE (STATE)

This subroutine uses an equation of state to relate the temperature to the energy or vice versa. It also calculates temperature dependent properties and calls TBNDRY to reset the temperature boundary conditions. It is called by MAIN and by INITLZ.

15. TBNDRY (Temperature BouNDaRY)

This subroutine sets the temperature boundary conditions. It is called by MAIN and by STATE.

16. TIMER (TIMER)

This subroutine keeps track of program times. It is called by MAIN and PRNT.

17. UPDATE (UPDATE)

This subroutine updates the pressure and velocity fields to the new iterate level according to the fully/linearly implicit artificial compressibility iteration scheme with damping. It is called by MAIN.

3.4 Code Options and Input

A sample input file for the MAGMA code is located in Appendix D. Most of the variables in Appendix D are self-explanatory. Those requiring additional information are described below. Note that additional variable definitions can be found in the variable dictionary in Appendix C.

The first input variable (*IRSTRT*) determines if a previous calculation is to be restarted (*IRSTRT*=1) or if a new calculation is to be performed (*IRSTRT*=0). If *IRSTRT*=1, the program reads all variables from a binary file called RINPUT. The remaining variables in the input file are read after the binary read from RINPUT. This allows additional program control and flexibility, however the user must be certain that input variables are consistent with the variables of the previous calculation. If *IRSTRT*=0, the program starts a new calculation.

ALPHA is a weighting parameter used in the differencing of the convective terms of the Navier-Stokes equations. It represents the percentage of upwinded or donor cell differencing. *ALPHA* = 1 represents pure donor cell differencing, while *ALPHA* = 0 represents pure centered differencing.

CYL is the two-dimensional cylindrical coordinate system flag. *CYL* = 1 indicates that two-dimensional cylindrical coordinates are to be used. Setting *CYL* to one also requires that *DELZ* be set to a negative number. A negative *DELZ* is interpreted by the code logic as an input signal for a purely two-dimensional calculation. The *y*-axis becomes the axis of symmetry for two-dimensional calculations and the *x*-coordinate becomes the radial coordinate. *CYL* = 0 indicates that either a two or three dimensional simulation is to be performed using rectangular coordinates. Three dimensional calculations are performed if *DELZ* is greater than zero, and two-dimensional calculations are performed if *DELZ* is less than zero. Note that the *x*-*y* plane is the default for two-dimensional calculations.

ECOEF1, *ECOEF2*, *ECOEF2*, and *ECOEF3* are temperature coefficients in the quadratic equation of state (Equation (14)) where *ECOEF1* = *a1*, *ECOEF2* = *a2*, *ECOEF3* = *a3*, and *ECOEF4* = *a4*.

EPSI is the tolerance value for the velocity divergence condition given by Equation (28), i.e., *EPSI* = ε_1. *TOL* is the tolerance value for the convergence condition given by Equation (29), i.e., *TOL* = ε_2.

TQB, *TQD*, *TQF*, *TQL*, *TQR*, *TQT* are temperature boundary condition parameters associated with the bottom, derriere, front, left, right, and top boundaries, respectively. The interpretation of these parameters depends upon the temperature boundary condition flags *ITMPB*, *ITMPD*, *ITMPF*, *ITMPL*, *ITMPR*, and *ITMPT*. The letters B, D, F, L, R, and T refer to the bottom, derriere, front, left, right, and top boundaries of the computational region. Table 1 indicates the interpretation of TQn as a a function of *ITMPn*, where *n* represents B, D, F, L, R, or T. As indicated in Table 1, TQH is the convective heat transfer coefficient when the convective wall boundary condition (*ITMPn* = 3) is selected.

Table 1. Interpretation of temperature boundary condition parameters

ITMPn	Interpretation of *TQn*
1	Wall temperature (constant temperature boundary condition)
2	Wall heat flux (constant heat flux boundary condition)
3	Temperature of surroundings TQH is the heat transfer coefficient (convective boundary condition)

TWFIN and *IFREQ* are program control variables. *TWFIN* sets the ending time for the calculations and *IFREQ* determines the frequency of data outputs. Data is written to a restart file and several output data files (i.e., *mltdat, outdat, prsdat, tmpdat, and veldat, and pltdat)* every *IFREQ* time cycles.

IENERGY, *IFIMP*, *IGUESS*, *IMELT*, and *IVISC* are program option flags. *IENERGY* = 1 indicates that the energy equation is to be solved, whereas *IENERGY* = 0 turns off the energy solution subroutine. *IMELT* controls the code melting and freezing options. *IMELT* = 1 allows cell melting and freezing, while *IMELT* = 0 turns off this option. *IVISC* controls the variable viscosity option in the MAGMA code. *IVISC* = 1 indicates that the variable viscosity formulation is to be used, while *IVISC* = 0 indicates that the constant viscosity formulation is to be used. Tables 2 and 3 indicate the various options associated with the *IFIMP* and *IGUESS* flags.

Additional input options may be directly hard wired into the Magma code. For example, expressions relating the fluid properties to temperature must be directly hard wired into the STATE subroutine. Initial non-uniform temperature or velocity distributions may be coded directly into the INPUT subroutine. The initial distribution of active cells in the computational region may also be coded directly into the INPUT subroutine. This task is accomplished by simply setting the necessary elements of the *IMAP* array to 1. Special boundary conditions not included in the MAGMA code can also be hard wired into the code. The BNDRY subroutine contains velocity

Table 2. Interpretation of the *IFIMP* variable flag

IFIMP	Interpretation
0	Indicates the fully implicit solution algorithm is to be used in conjunction with the momentum equation.
1	Indicates the linearly implicit solution algorithm is to be used in conjunction with the momentum equation
2	Indicates the semi-implicit solution algorithm is to be used in conjunction with the momentum equation

Table 3. Interpretation of the *IGUESS* variable flag

IGUESS	Interpretation
0	Initial velocity guess is to be computed from an explicit approximation to the momentum equation.
1	Initial velocity guess is the velocity solution from the previous time step.
2	Initial velocity guess is to be computed from an extrapolation from the two previous time step velocity solution.

boundary condition logic and the TBNDRY subroutine contains the temperature boundary condition logic. MAGMA allows no-slip and free slip wall velocity boundary conditions; and constant temperature, constant heat flux, and convective temperature boundary conditions.

Array dimensions and common blocks can be changed in a single INCLUDE file named 'comd.h', which is included in the mainline and all subroutines. All important PARAMETER statements, DIMENSION statements, and COMMON blocks, are also located in this file.

3.5 Code Output

Data is output to four ASCI data files and one binary data file after every IFREQ time cycles. The binary data file is a restart file named ROUTPT. This file contains a binary dump of all variables necessary for a successful restart of a previous calculation. In order to restart a calculation, the flag IRSTRT is set to one and ROUTPT is renamed RINPUT before the program is executed.

The four ASCI data files are output in a similar fashion. After IFREQ time cycles the velocity, temperature, pressure are written to the files named: 'veldat', 'tmpdat', and 'prsdat', respectively. Other variable arrays such as energy and temperature dependent properties are written to the file named 'outdat'. If cell melting is allowed, the melt front profile is written to the file named 'mltdat'. The form of a program output is similar for each of these files. At each output, header information is written to the data file before any data. This header information includes a brief description of the type of data (i.e., VELOCITY DATA), the output time, the number of cycles, the mesh spacing, etc... In the veldat data file, following the header information the x, y, and z coordinates are written with the corresponding u, v, and w velocity components on the same output card (i.e., x, y, z, u, v, w). Other data files write the x, y, and z coordinates with the corresponding variable value on the same output card (i.e., x, y, z, p).

After IFREQ time cycles, the time, the cycle count, and the iteration count are written to the standard output device so that the user can monitor the progress of the code calculations. Note that this output should be tailored to meet the user's own monitoring requirements. If the code is executed in batch mode, this output should be redirected to a data file or suppressed.

A post-processing routine may be necessary to convert MAGMA data into the correct form for use with an external plotting package. The MAGMA code does not have any internal plotting capability.

4. CODE TESTING AND VERIFICATION

MAGMA has been assessed against numerous numerical and analytical test problems found in the literature. Table 4 presents a summary of these test problems. They have been divided into five main categories: 1) Two-dimensional advection, 2) Two-dimensional conduction, 3) Two-dimensional natural convection, 4) Three-dimensional natural convection, and 5) Melting.

During two-dimensional tests, each of the three two-dimensional planes were tested (i.e., x-y, x-z, and y-z). The results obtained were identical, verifying that symmetry existed in each of the three coordinate directions. Although tests were performed on each of the two-dimensional planes, two dimensional calculations are presently restricted to the x-y plane.

Quantitative data comparisons were possible in assessing MAGMA against tests 1, 2a, 2b, 3b, and 3c. MAGMA results compared well with analytical solutions for tests 1, 2a, and 2b.[6,7,8,9] Good agreement was obtained with numerical solutions for tests 3b and 3c.[12] Qualitative comparisons were made with numerical solutions to tests 3a, 4a, 4b, and 5. MAGMA results also compared favorably with these numerical solutions, especially considering differences in the numerical techniques and computational meshes.[10,13,14]

The results from MAGMA simulations of tests 3b, 4a, and 5 are presented below in order to demonstrate the capabilities and versatility of the code. These simulations were chosen because they involve two and three dimensional natural convection, rectangular and cylindrical coordinates, and heat transfer and fluid flow during melting.

Test 3b is a benchmark numerical solution for steady state, two-dimensional, natural convection.[12] The problem geometry and thermal boundary condition are illustrated in Figure 9. All boundaries are rigid, no-slip walls. The left wall is maintained at the hot temperature, T_h. The right wall is maintained at the cold temperature, T_c. Both the front and derriere boundaries are adiabatic walls.

MAGMA calculated the steady state solution by marching the transient solution out to a steady state, while Reference 12 used a steady state solution algorithm to avoid computing the transient solution. In addition, full donor cell differencing of the convective terms limited the the MAGMA solution to first order accuracy, while the numerical solution of Reference 12 was second order accurate.

Table 4. Summary of MAGMA test problems

Test Category	Test Number and Description				Reference
1. Advection (2-D)	1.	Name:	Driven Cavity		
		Coord. System:	2-D Rectangular		
		Grid:	40X40		
		Boundaries:	Left	- No-slip	
			Right	- No-slip	
			Front	- Specified	6, 7
			Derriere	- No-slip	
		Initial Conditions:	Zero internal velocity		
		Parameters:	Re = 400		
		Comparison:	Qualitative & quantitative with numerical results		
2. Conduction (2-D)	2a.	Name:	Transient Conduction-Rectangular		
		Coord. System:	2-D Rectangular		
		Grid:	20X40 (rect. plate)		
		Boundaries:	Zero surface temp.		8
		Initial Conditions:	$T(x,y,t{=}0) = xy$		
		Parameters:	$k = c = r = 1$		
		Comparison:	Quantitative with analytical solution		
	2b.	Name:	Transient Conduction-Cylindrical		
		Coord. System:	2-D Cylindrical		
		Grid:	10X10 (finite cylinder)		
		Boundaries:	Zero surface temp.		9
		Initial Conditions:	$T(x,y,t{=}0) = 1$		
		Parameters:	Thermal diffusivity = 0.1		
		Comparison:	Qualitative with analytical solution		
3. Natural Convection (2-D)	3a.	Name:	Heated Fluid Layer		
		Coord. System:	2-D Rectangular		
		Grid:	40X20 (rect. plate, $g = -g\, e_y$)		
		Boundries:	No Slip walls		
			Front	--Isothermal	
			Derriere	--Isothermal ($T_D > T_F$)	10, 11
			Left, Right	--Adiabatic	
		Initial Conditions:	Zero, velocity Uniform Temp. gradient Initial sinusoidal temp. perturbation		
		Parameters:	Ra = 30,000, Pr = 1.0		
		Comparison:	Qualitative with steady state numerical solution		

Table 4. (continued)

Test Category	Test Number and Description				Reference
	3b.	Name:	Benchmark Natural Convection		
		Coord. System:	2-D Rectangular		
		Grid:	10X10, 20X20, 40X40 ($\mathbf{g} = -g\ \mathbf{e}_y$)		
		Boundaries:	No-slip walls		
			Left	--Isothermal	
			Right	--Isothermal ($T_L > T_R$)	
			Front	--Adiabatic	
			Derriere	--Adiabatic	12
		Initial Conditions:	Uniform temperature		
		Parameters:	Ra = 10,000, Pr = 0.71		
		Comparison:	Qualitative & quantitative with steady state numerical solution		
	3c.	Name:	Benchmark Natural Convection		
		Coord. System:	2-D Rectangular		
		Grid:	10X10, 20X20, 40X40 ($\mathbf{g} = -g\ \mathbf{e}_y$)		
		Boundaries:	No-slip walls		
			Left	--Isothermal	
			Right	--Isothermal ($T_L > T_R$)	12
			Front	--Adiabatic	
			Derriere	--Adiabatic	
		Initial Conditions:	Uniform temperature		
		Parameters:	Ra = 100,000, Pr = 0.71		
		Comparison:	Qualitative & quantitative with steady state numerical solution		
4. Natural Convection (3-D)	4a.	Name:	3-D Natural Convection		
		Coord. System:	3-D Rectangular		
		Grid:	9X9X9 ($\mathbf{g} = -g\ \mathbf{e}_y$)		
		Boundaries:	No-slip walls		
			Left	--Isothermal, $T_L = T$	
			Right	--Isothermal, $T_R = T$	
			Front	--Isothermal, $T_F = T$	
			Top	--Isothermal, $T_T = T$	13
			Bottom	--Isothermal, $T_B = T$	
			Derriere	--Isothermal--$T_D > T$	
		Initial Conditions:	Uniform temperature		
		Parameters:	Ra = 10,000, Pr = 0.72		
		Comparison:	Qualitative with steady state numerical solution		

Table 4. (continued)

Test Category	Test Number and Description			Reference
	4b.	Name:	3-D Natural Convection	
		Coord. System:	3-D Rectangular	
		Grid:	9X9X9 ($\mathbf{g} = -g\,\mathbf{e}_y$)	
		Boundaries:	No-slip walls	
			Left --Adiabatic	
			Right --Adiabatic	
			Front --Isothermal, T_F	
			Top --Adiabatic	13
			Bottom --Adiabatic	
			Derriere -- Isothermal -- $T_D > T_F$	
		Initial Conditions:	Uniform temperature Temp. perturbation	
		Parameters:	Ra = 10,000, Pr = 0.72	
		Comparison:	Qualitative with steady state numerical solution	
5. Melting (2-D)	5.	Name:	Melting about a Vertical Cylinder	
		Coord. System:	2-D Cylindrical	
		Grid:	40X20 (cylindrical disk, $\mathbf{g} = -g\,\mathbf{e}_y$)	
		Boundaries:	No-slip walls	
			Left --Isothermal, T_F ($T_F > T_m$)	
			Right --Adiabatic	
			Front --Adiabatic	
			Derriere --Adiabatic	14
		Initial Conditions:	Phase Change Material (PCM) at melt temp.=T_m	
		Parameters:	Ra = 3.16◊10^{6}, Pr = 56.2, Ste = 0.094	
		Comparison:	Qualitative with transient numerical solution	

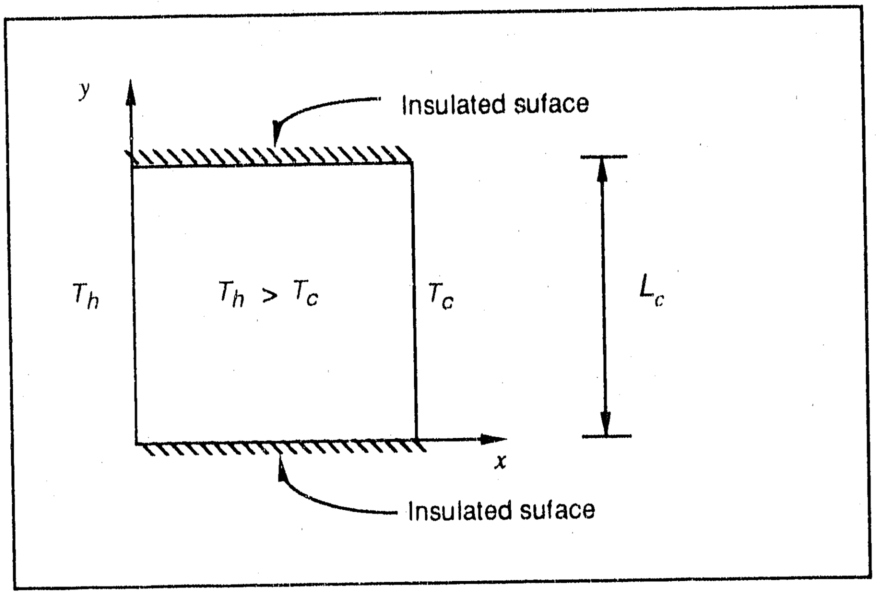


Figure 9. Geometry and thermal boundary conditions for test 3b.

Quantitative comparisons of data values from Reference 12 and the MAGMA solution are shown in Table 5. The Reference 12 benchmark solution in the last column was calculated by extrapolating the solutions from the two finest meshes ($\Delta x = \Delta y = 0.05$ and $\Delta x = \Delta y = 0.025$) to approximate the exact solution ($\Delta x = \Delta y = 0$). Reference 12 maximum velocity values were obtained from interpolation, while the MAGMA values represent the maximum cell centered mesh values. Note that velocities have been scaled by $\frac{\alpha}{L_C}$ and lengths by L_C, where α is the thermal diffusivity. Generally good agreement was obtained, considering the differences in the numerical techniques. Table 5 indicates that the MAGMA solution is converging to the benchmark solution as the grid size is reduced.

Figures 10 and 11 present the steady state velocity vectors and temperature isotherms, respectively, for MAGMA calculations on a 40X40 ($\Delta x = \Delta y = 0.025$) mesh. Comparison of Figure 11 against the steady state temperature isotherms from Reference 12 (Figure 12) demonstrates good agreement.

Table 5. Data comparison with test number number 3b

	$\Delta x = \Delta y = 0.1$		$\Delta x = \Delta y = 0.05$		$\Delta x = \Delta y = 0.025$		$\Delta x \to 0$ $\Delta y \to 0$
Data Value	MAGMA	Reference 13	MAGMA	Reference 13	MAGMA	Reference 13	Reference 13 Bench-mark
u_{max}	15.047	16.243	15.723	16.189	15.955	16.182	16.178
y	0.85	0.808	0.825	0.820	0.8125	0.823	0.823
v_{max}	16.733	18.055	18.757	19.197	19.263	19.509	19.617
x	0.15	0.139	0.125	0.125	0.1125	0.120	0.119
$\bar{Nu}$	2.190	2.171	2.206	2.212	2.230	2.234	2.243
Nu $(x = \frac{1}{2})$	2.042	2.170	2.164	2.213	2.218	2.235	2.243
Nu $(x = 0)$	2.383	2.307	2.271	2.255	2.254	2.242	2.238

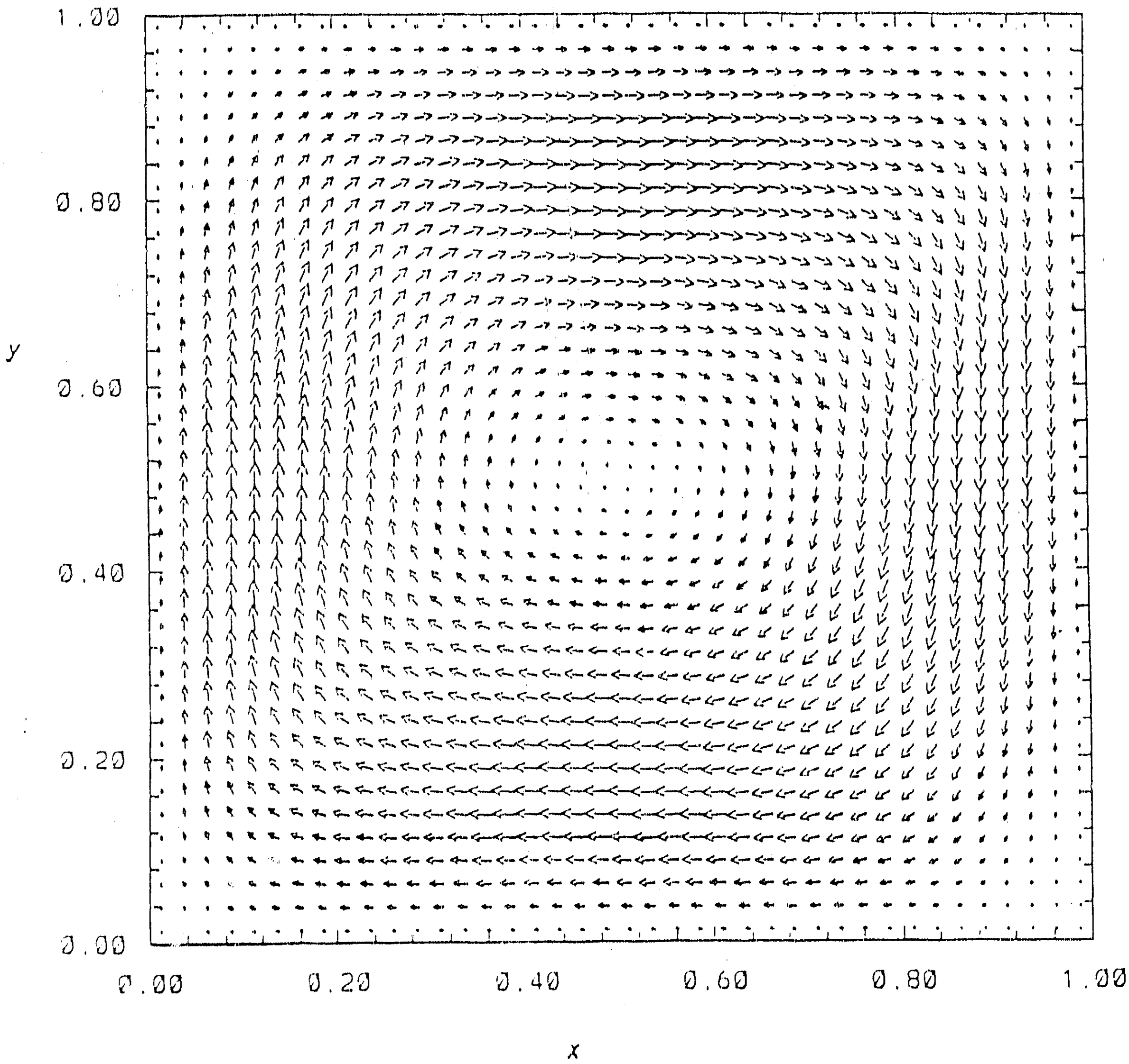


Figure 10. MAGMA steady state velocity vectors for test 3b (maximum dimensionless velocity vector = 19.25).

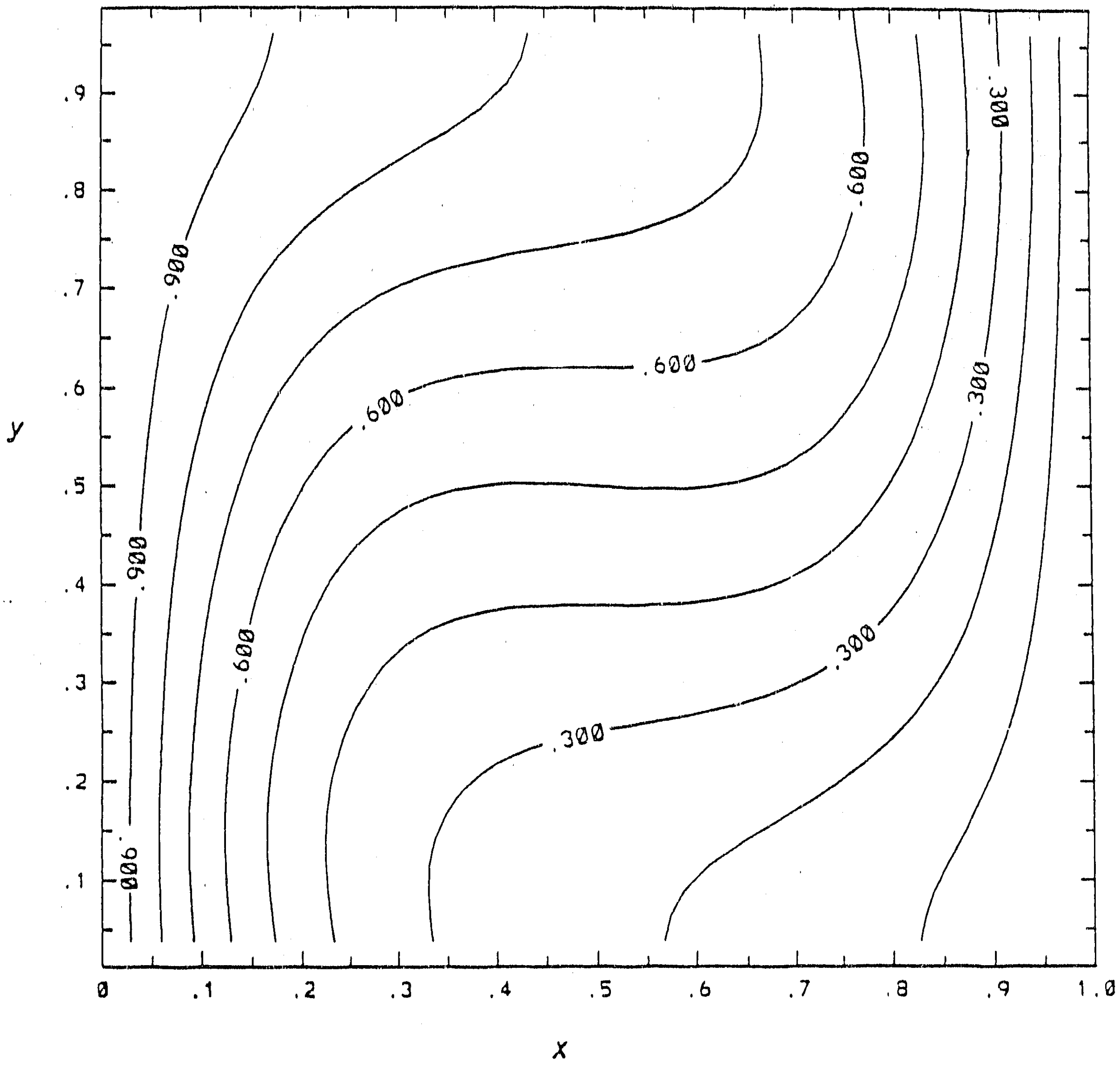


Figure 11. MAGMA steady state temperature isotherms for test 3b (contours at 0 (0.10) 1).

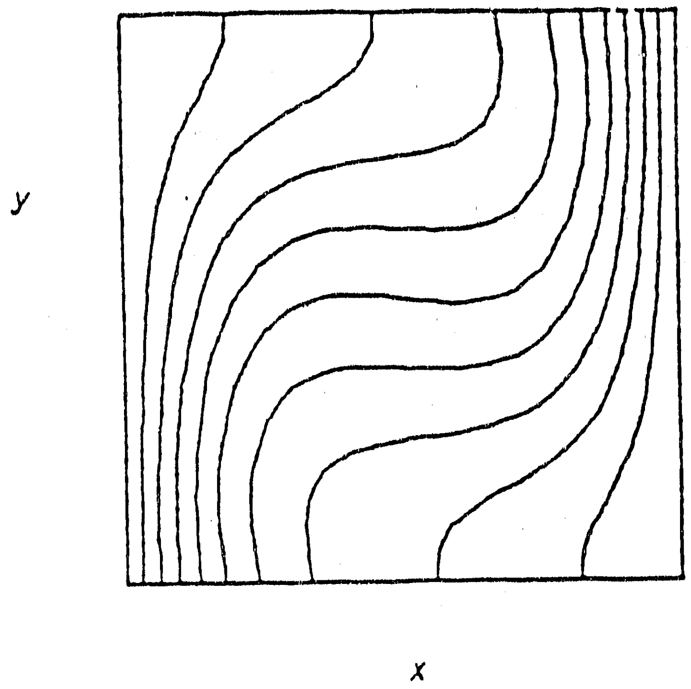


Figure 12. Reference 12 steady state temperature isotherms for test 3b (contours at 0 (0.10) 1).

Test 4a is a three dimensional natural convection simulation. The geometry and thermal boundary conditions for this test are illustrated in Figure 13.[13] All boundaries are rigid, no-slip, isothermal, walls. The temperatures on the left, right, bottom, top and front walls are equal to the cold temperature, T_c. The derriere wall is set to the hot temperature, T_h. The initial temperature is set to the average of T_c and T_h. These and other parameter specifications are listed in Table 4.

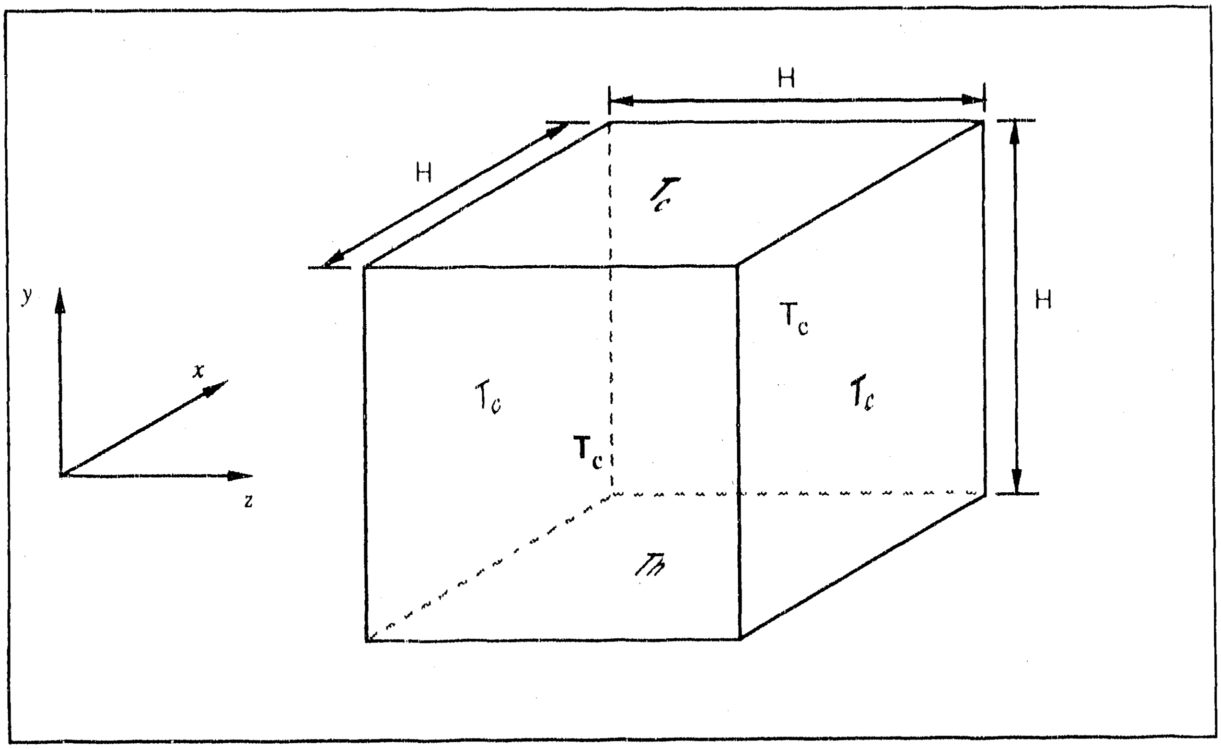


Figure 13. Geometry and thermal boundary conditions for test 4a.

Cooling on the side walls immediately initiates a downward flow of liquid. This liquid is heated as it passes along the derriere wall causing it to rise up through the center of the cube. The steady state natural circulation pattern is thus characterized by liquid falling along the side walls and subsequently rising through the center of the cube. MAGMA-calculated temperature and vertical velocity (v) profiles at mid-height ($y/H = 0.5$) for each of the x-y planes are shown in Figures 14 and 15, respectively. Calculated results from Reference 13 are shown in Figures 16 and 17 for comparison purposes. Qualitatively the plots are similar, but quantitative agreement is marginal.

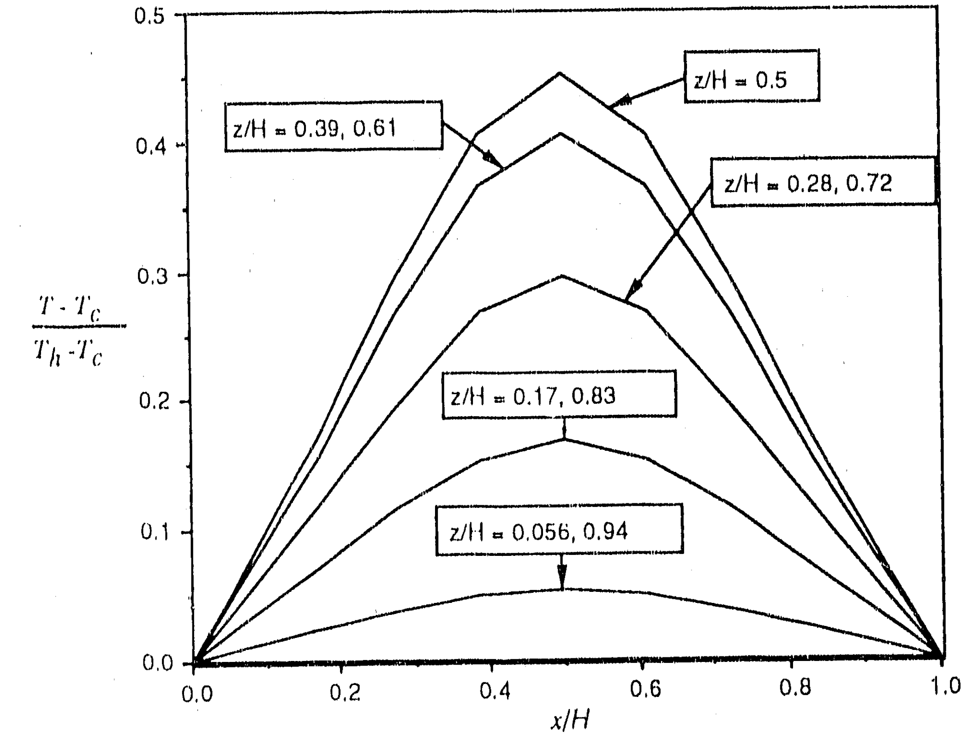


Figure 14. MAGMA temperature profiles at mid-height for test 4a.

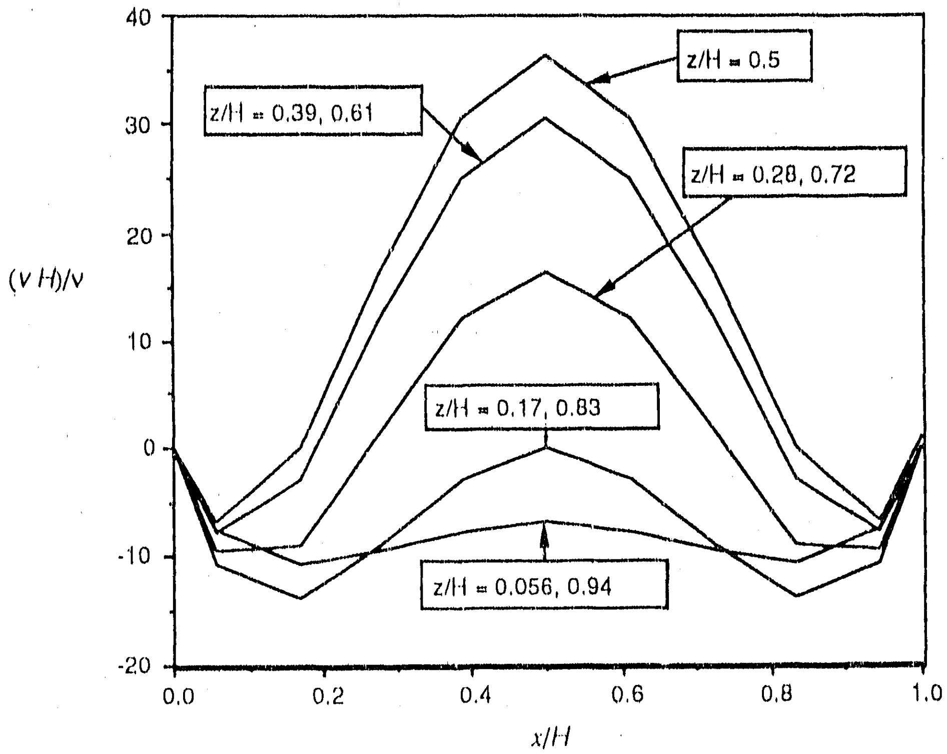


Figure 15. MAGMA vertical velocity profiles at mid-height for test 4a.

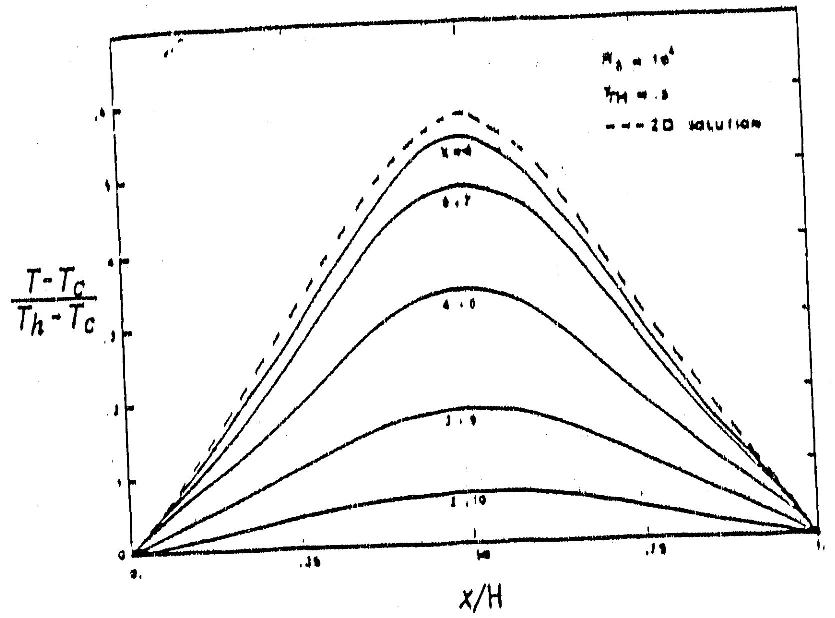


Figure 16. Reference 13 steady state temperature profiles at mid-height for test 4a.

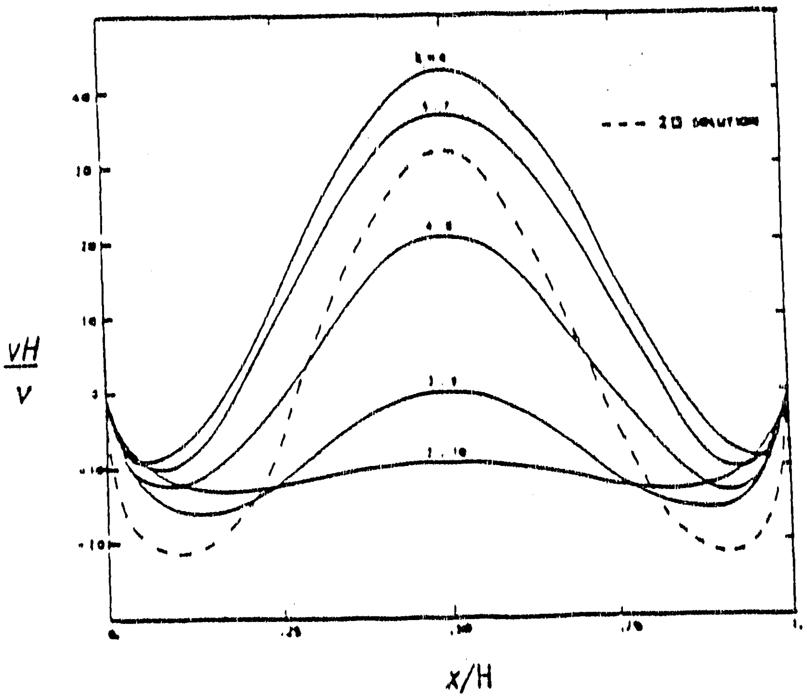


Figure 17. Reference 13 steady state vertical velocity profiles at mid-height for test 4a.

An explanation for the quantitative differences between the two numerical results has not yet been determined. MAGMA-calculated steady state temperature contours and velocity vectors for the plane $z/H = 0.5$ are shown in Figures 18 and 19, respectively. Corresponding plots from Reference 13 are shown in Figures 20 and 21. The two numerical solutions qualitatively predict similar steady state behavior.

Test 5 investigates natural convection during melting about a heated finite cylinder.[14] The geometry and thermal boundary conditions for this test are illustrated in Figure 22. All boundaries are rigid, and have no-slip walls. The left wall coincides with the surface of the cylinder and is maintained at the temperature T_o, which is higher than the melt temperature, T_m, of the Phase Change Material (PCM). All other boundaries are insulated, adiabatic walls. The transient results of this simulation are identified by a dimensionless time defined by Reference 14 as

$$\tau = \frac{\alpha tc(T_o - T_m)}{H^2L} \tag{78}$$

where α is the thermal diffusivity, t is the physical time, c is the specific heat capacity, L is the latent heat of fusion, and H is the height of the cylinder.

The initial temperature of the PCM was set to the PCM melt temperature, T_m. MAGMA assumed all cells were initially in the solid phase (no melting). Heat is transferred from the left wall (i.e., the cylinder) to the PCM, due to the initial temperature difference. Initially, this energy transfer is used to supply the latent heat necessary to initiate cell melting in the layer of cells immediately adjacent to the cylinder. The entire first layer of cells melt when $\tau = 0.00147$. Cell melting is followed by cell temperature increases, which in turn initiates heat transfer to the second layer of cells. Natural convection begins when the first two layers of cells have completely melted. The buoyancy driven natural convection flow transfers more heat to the cells near the front boundary, causing these cells to melt before the other cells in the layer. MAGMA calculated melt front profiles resulting from this melting process are shown in Figure 23. Qualitatively, the MAGMA melt front profiles agree reasonably well with the results from Reference 14, presented in Figure 24. Note that both numerical solutions predict the melt front extending farther across the front boundary. MAGMA velocity vector and temperature contour plots at $\tau = 0.0983$ (Figures 25 and 26, respectively) demonstrate the heat transfer and fluid flow behavior in the melt region.

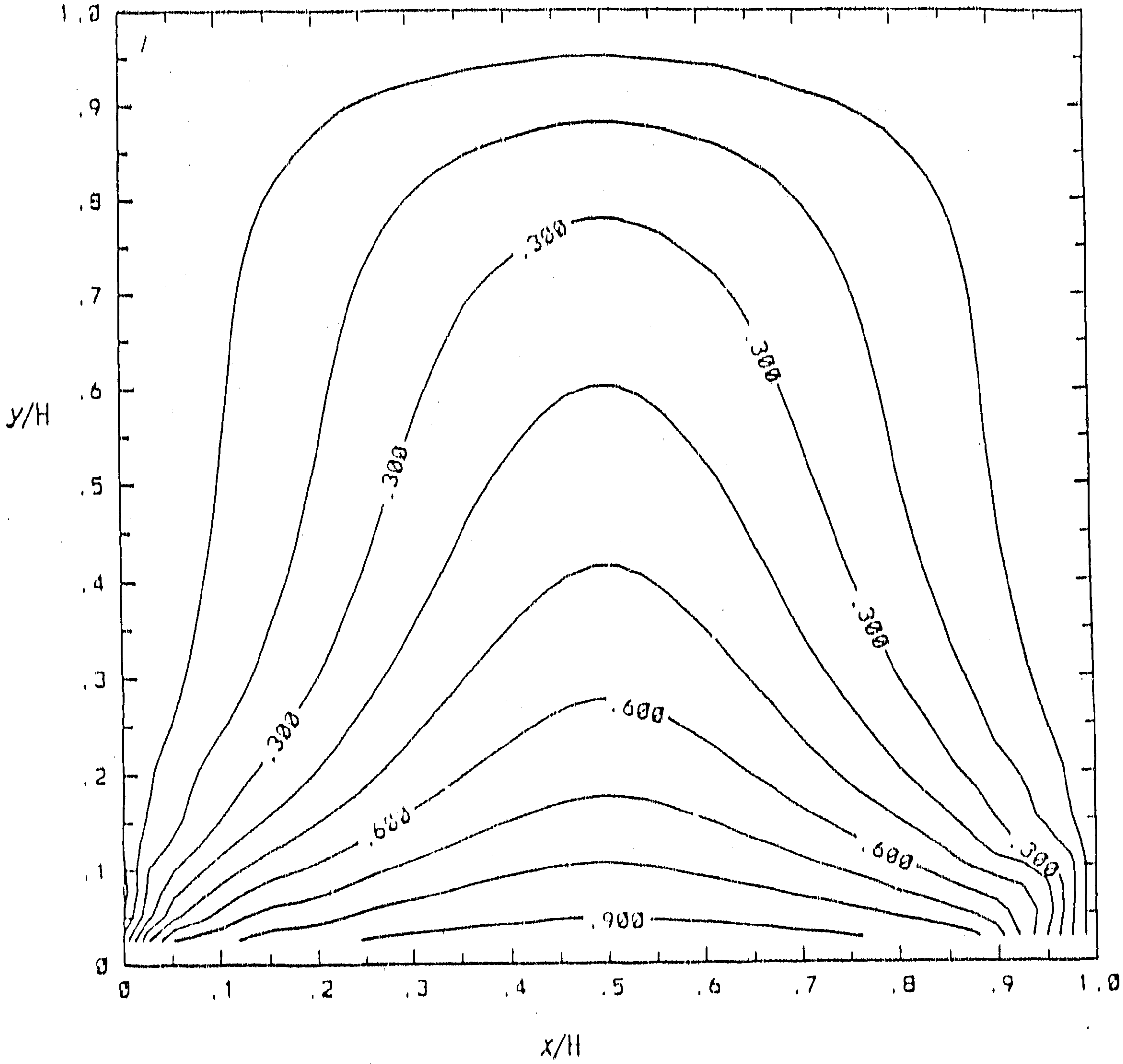


Figure 18. MAGMA steady state temperature contours at z/H = 0.5 for test 4a (contours at 0 (0.10) 1).

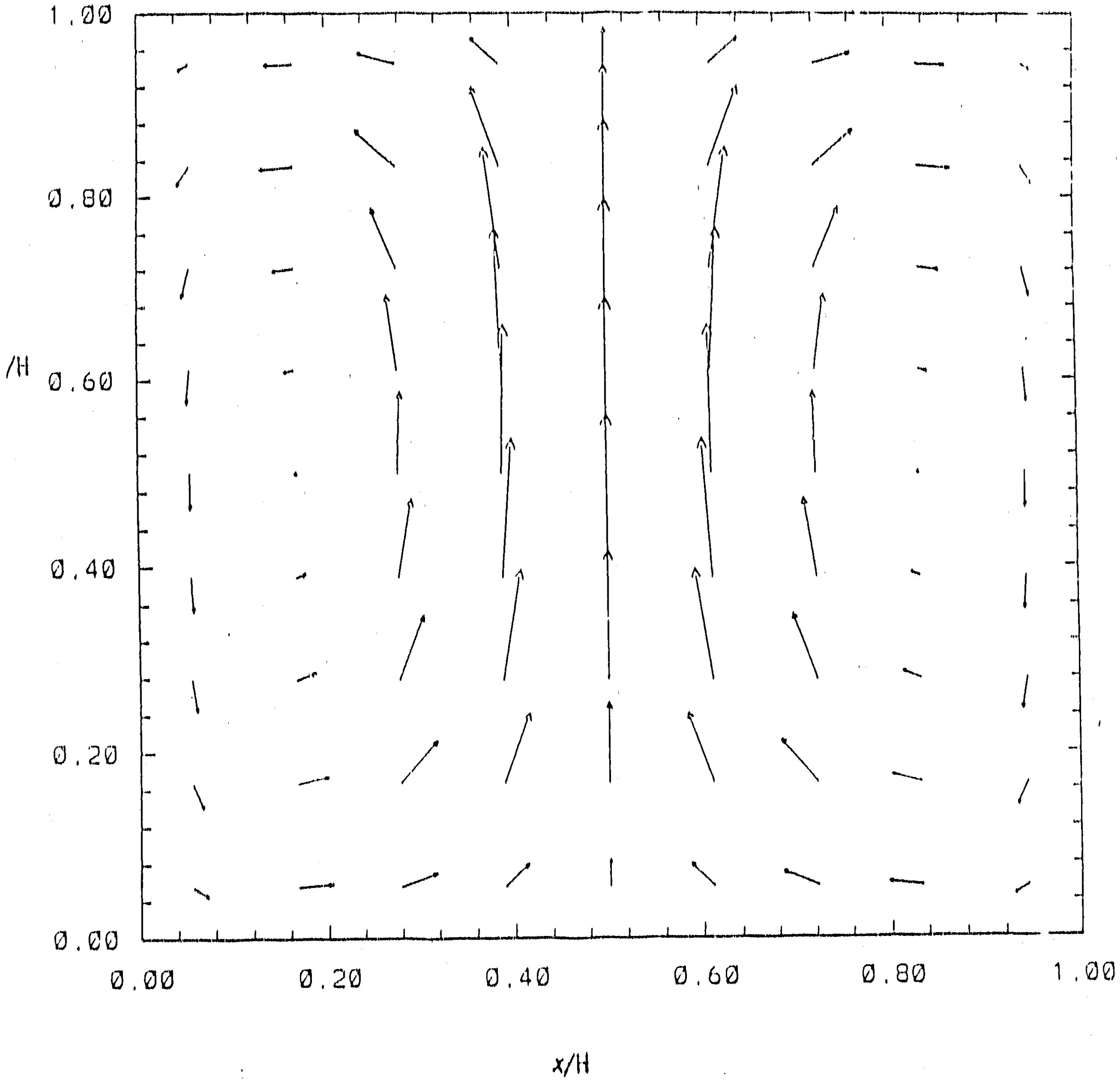


Figure 19. MAGMA steady state velocity vectors at z/H = 0.5 for test 4a (maximum dimensionless velocity = 36.34).

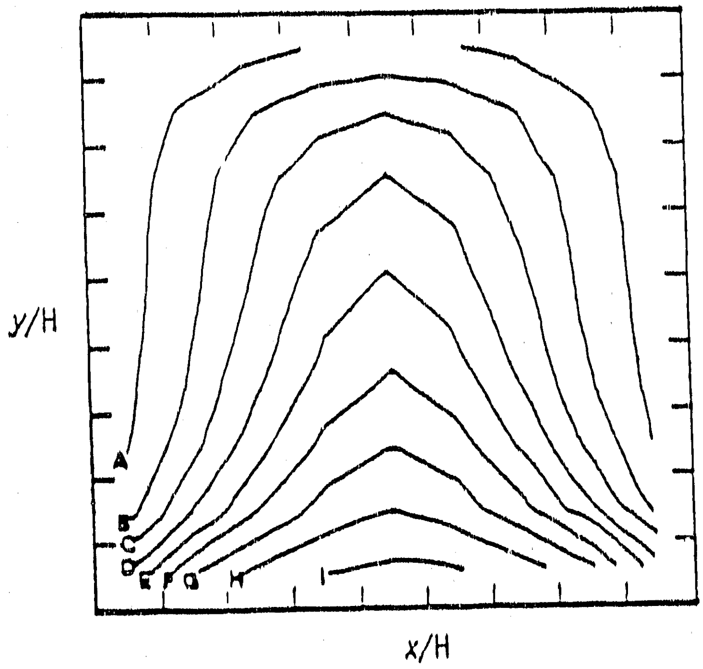


Figure 20. Reference 13 steady state temperature contours at z/H = 0.5 for test 4a (contours at 0 (0.10) 1).

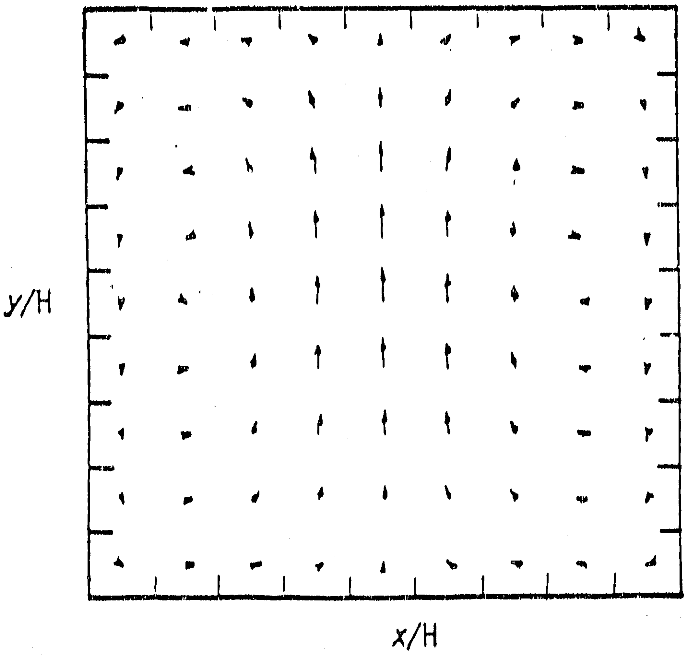


Figure 21. Reference 13 steady state velocity vectors at z/H = 0.5 for test 4a (maximum dimensionless velocity = b43).

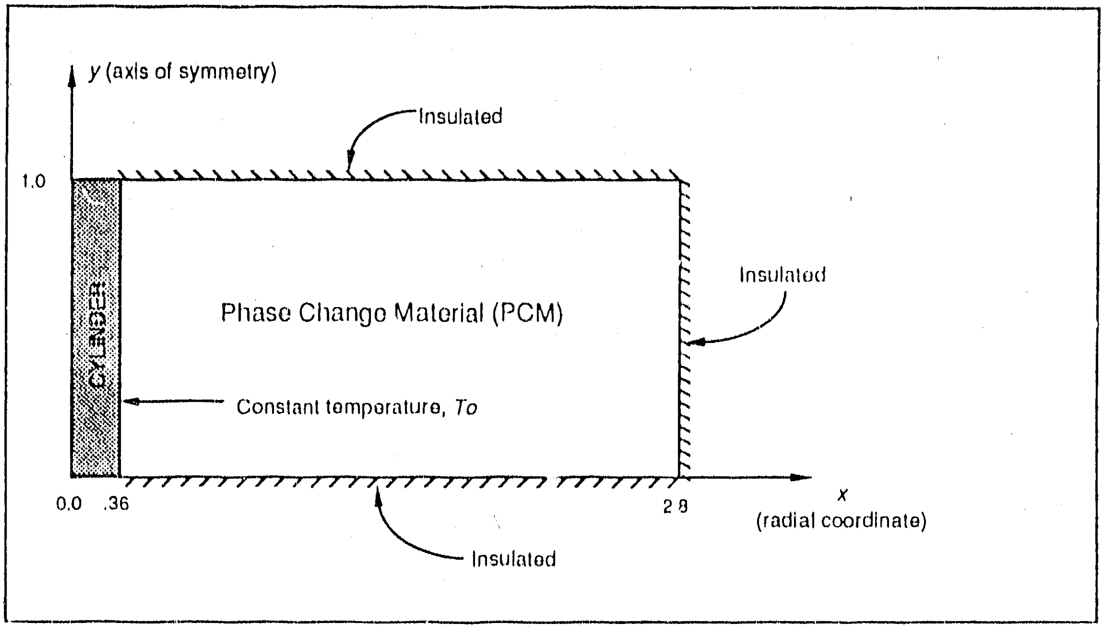


Figure 22. Geometry and thermal boundary conditions for test 5.

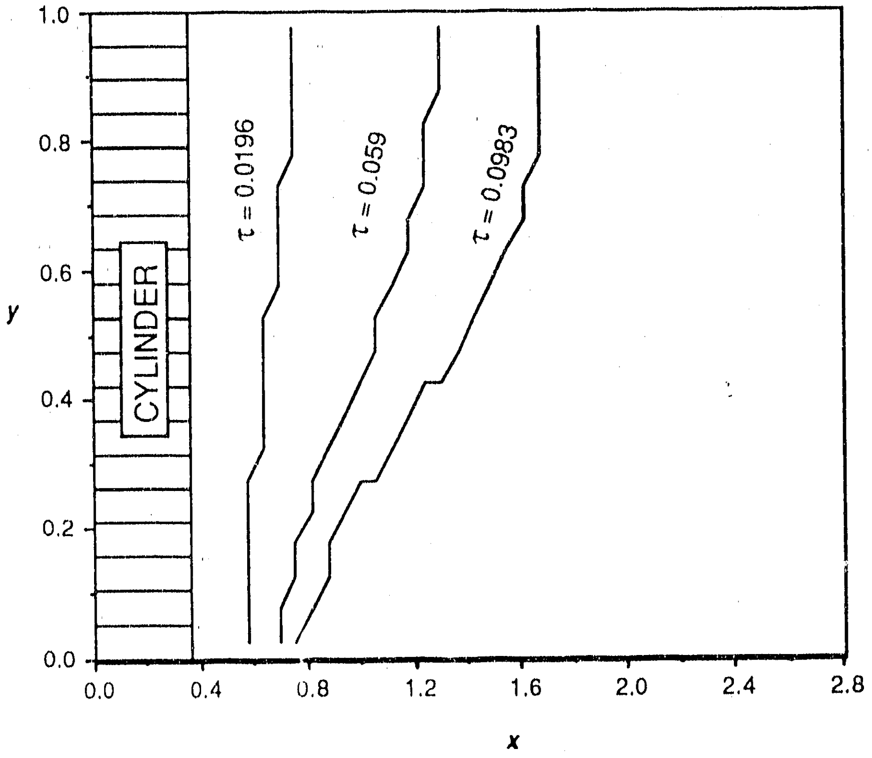


Figure 23. MAGMA melt front profiles versus nondimensional time.

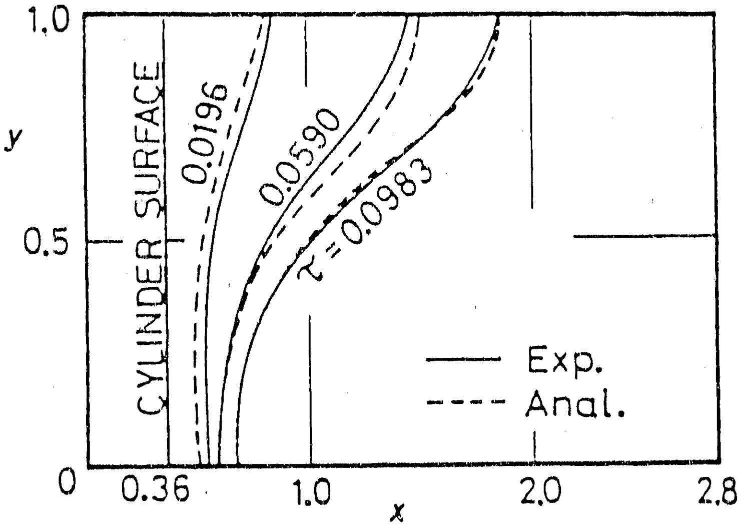


Figure 24. Reference 14 melt front profiles versus nondimesional time.

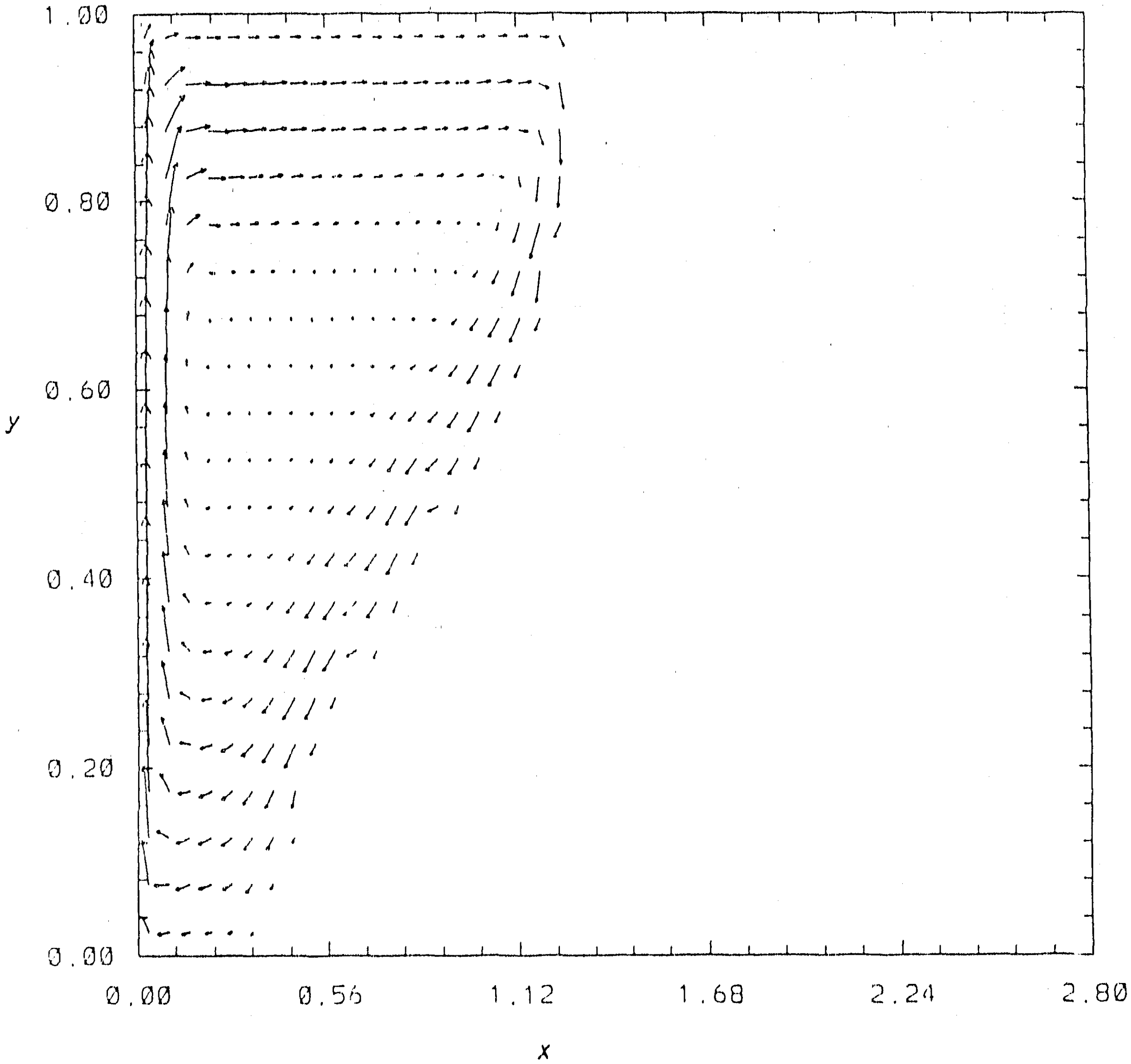


Figure 25. MAGMA velocity vectors at $\tau = 0.0983$ for test 5.

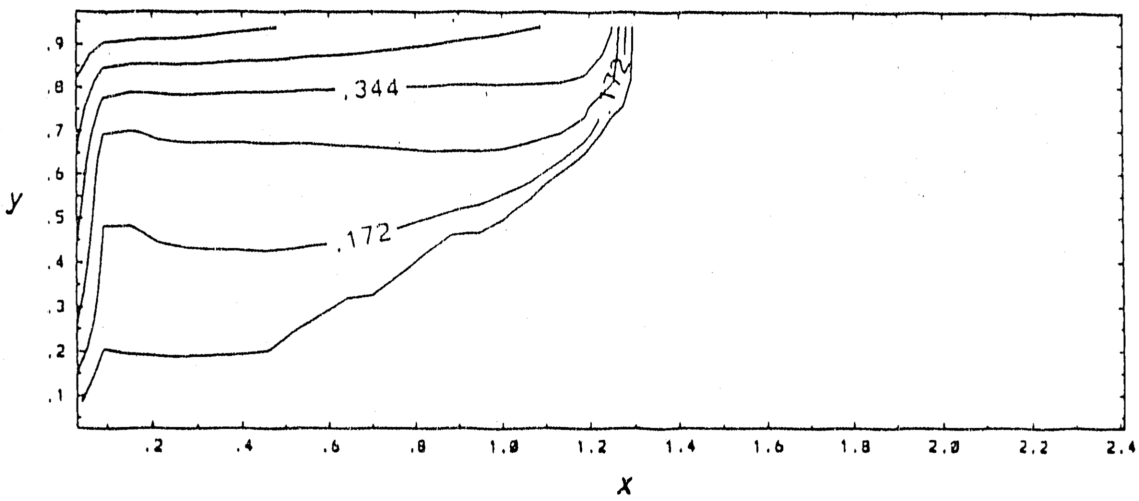


Figure 26. MAGMA temperature contours at $\tau = 0.0983$ for test 5 (contours at 0 (0.10) 1).

Table 6 compares the calculated dimensionless molten volume of the melt region obtained from MAGMA and from Reference 14. The dimensionless molten volume is defined by Reference 14 as

$$\text{Vol} = \frac{H^2}{x_o^2}\int_0^1 \frac{x_m^2}{H^2}\,dy - 1 \tag{79}$$

where H is the height of the cylinder, x_o^2 is the radial coordinate of the cylinder surface, and x_m^2 is the radial coordinate of the melt front. Generally good agreement is obtained.

Figure 27 presents the transient response of the average Nusselt number. The large oscillations in this figure result from the nature of the MAGMA melting model. When a cell or group of cells melt, the average Nusselt number is significantly affected by the subsequent rise in cell temperatures. This effect is especially apparent during the early stages of the transient, when entire layers of cells melt simultaneously. Oscillations were not encountered in the Reference 14 simulation because the melt front was allowed to expand continuously. Figure 28 shows the results of the average Nusselt number calculated from Reference 14. However, MAGMA and Reference 14 calculations predicted similar transient behavior of the average Nusselt number.

Table 6. Comparison of the transient dimensionless molten volume for test number 5

Dimensionless Time	MAGMA	Reference 14
0.0196	2.06	2.0
0.059	6.89	6.0
0.0983	11.09	10.0

It is anticipated that improved accuracy and agreement with Reference 14 could have been obtained through mesh refinement. Reference 14 used a variable transformation from the physical coordinates to the computational coordinates to obtain a mesh with considerably higher resolution than the MAGMA mesh.

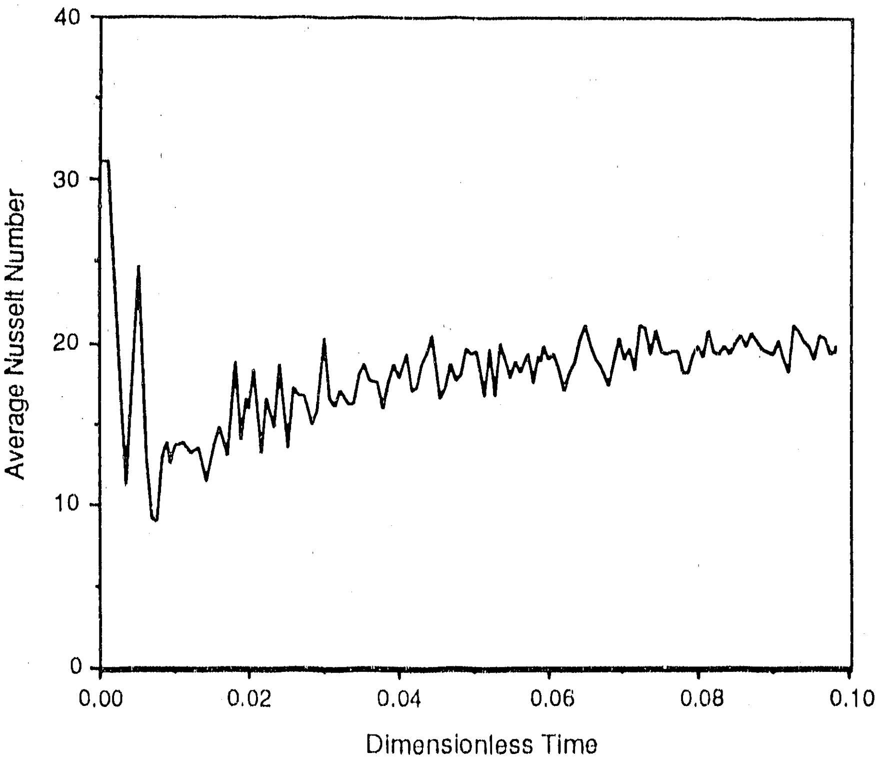


Figure 27. MAGMA calculated transient response of the average Nusselt number.

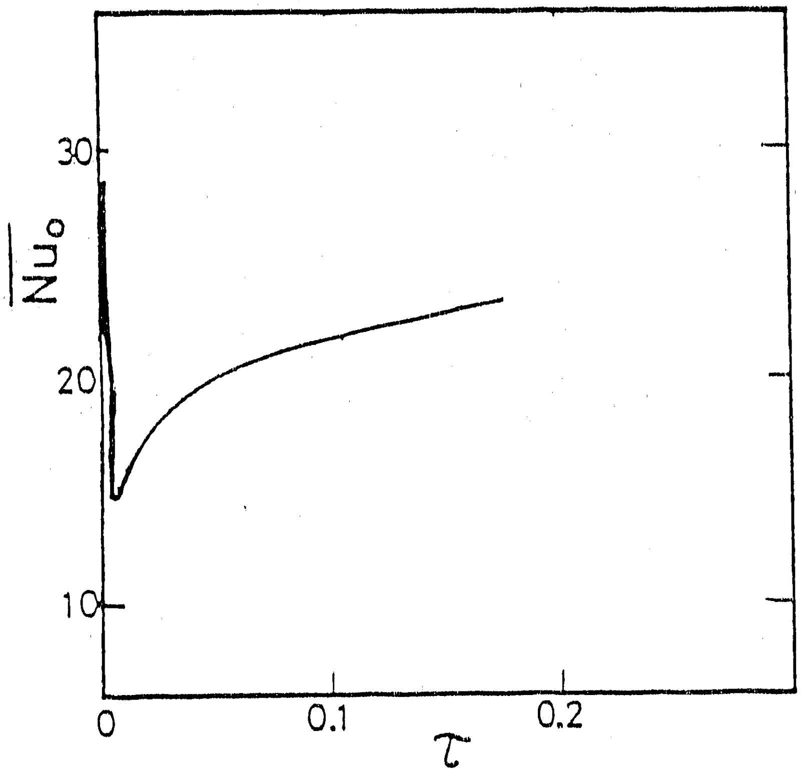


Figure 28. Reference 14 calculated transient response of the average Nusselt number.

5. SUMMARY AND RECOMMENDATIONS FOR FURTHER CODE DEVELOPMENT

MAGMA is a three-dimensional finite difference computer code designed to simulate fluid flow and heat transfer in ISV melt pools. The equations governing fluid flow are the incompressible Navier-Stokes equations. Heat transfer is governed by the energy equation. Energy is related to temperature through an equation of state that includes melting. The momentum equation is coupled to the temperature field through the Boussinesq model approximations.[1] The coupled governing equations can be solved simultaneously, or the incompressible Navier-Stokes equations can be solved separately in the absence of heat transfer. All fluid properties in MAGMA are assumed variable except density. Fluid density is allowed to vary only in the buoyancy terms of the momentum equation.

MAGMA has been tested against the results of many numerical and analytical test problems. MAGMA has demonstrated the capability to simulate two dimensional advection, two dimensional conduction, two and three dimensional natural convection, and natural convection during melting.

Recommendations for further investigation and development of the MAGMA code include:

- Conversion of the explicit temporal differencing of the energy equation to a fully implicit differencing scheme.
- Addition of a Joule heating model.
- Optimization of the iteration parameters.
- Investigation and determination of optimum convergence criteria for the iterative solution.
- Modification of the variable viscosity formulation to eliminate artificial time step restrictions arising when this option is selected.
- Assessment of MAGMA against ISV test data.

It is anticipated that the above recommendations could be readily implemented, and have the potential to significantly enhance code performance.

REFERENCES

1. L. D. Landau and E. M. Lifshitz, *Fluid Mechanics* 2nd *edition*, *6*, 1987.

2. R. Peyret and T. D. Taylor, *Computational Methods for Fluid Flow*, Springer-Verlag, New York, 1983.

3. J. D. Ramshaw and V. A. Mousseau, "Accelerated Artificial Compressibility Method for Steady-State Incompressible Flow Calculations," *Computers & Fluids*, *18*, 1990, p. 361 .

4. J. D. Ramshaw and J. K. Dukowicz, *APACHE; A Generalized-Mesh Eulerian Computer Code for Multicomponent Chemically Reactive Fluid Flow*, Los Alamos Scientific Laboratory, LA-7427, 1979.

5. F. P. Incropera and D. P. DeWitt, *Introduction to Heat Transfer 2nd edition.*, John Wiley & Sons, New York, 1990.

6. W. H. Soh and J. W. Goodrich, "Unsteady Solution of Incompressible Navier-Stokes Equations," *Journal of Computational Physics*, *79*, 1984, p.113.

7. U. Gia, et. al., "High-Re Solutions for Incompressible Flow Using the Navier-Stokes Equations and a Multigrid Method," *Journal of Computational Physics 48*, 1982, pp. 387-411.

8. D. L. Powers, *Boundary Value Problems*, Academic Press, 1979, p. 205.

9. H. S. Carslaw and J. C. Jaeger, *Conduction of Heat in Solids 2nd edition.*, Clarendon Press, 1959.

10. J. E. Fromm, "Numerical Solution of the Nonlinear Equations for a Heated Fluid Layer," *Physics of Fluids*, *8*, 1965, pp. 1757-69.

11. J. S. Turner, *Buoyancy Effects in Fluids*, Cambridge University Press, New York, 1973.

12. G. De Vahl Davis, "Natural Convection of Air in a Square Cavity: A Benchmark Numerical Solution," *International Journal of Numerical Methods in Fluids*, *3*, 1983, pp. 249-264.

13. A. M. C. Chan and S. Banerjee, "A Numerical Study of Three-Dimensional Roll Cells Within Rigid Boundaries," *Journal of Heat Transfer*, *101*, 1979, p. 233.

14. Masashi Okada, "Heat Transfer During Melting Around a Vertical Cylinder (Effects of Aspect Ratio of the Cylinder and Subcooling)," *JSME International Journal*, *30*, 1987, pp. 1430-1436.

APPENDIX A

AVERAGING TECHNIQUES

APPENDIX A

AVERAGING TECHNIQUES

This appendix describes the averaging procedure used in the MAGMA code. MAGMA uses a staggered grid in which some variables are located at cell centers and others are located at the centers of cell faces. Frequently, values of certain variables are needed at locations where they are not explicitly defined. Since MAGMA assumes a uniform mesh, simple averaging can be used to obtain these values.

Values of cell centered quantities are often needed on cell faces or on cell edges. The value of the quantity at these locations is obtained by averaging the quantity values located at the centers of the active cells immediately adjacent to that cell face or edge. Consider Figure A.1 as an example, which shows four active cells in the x-y plane (the z-direction is out of the paper). In this figure, the quantity, q, is a cell centered quantity. The value of q on the right (subscript R) face of cell (i,j,k) is computed by averaging the value of q in cell (i,j,k) and cell $(i+1,j,k)$, ie.,

$$q_R = \frac{1}{2}(q_{i,j,k} + q_{i+1,j,k}) \tag{A-1}$$

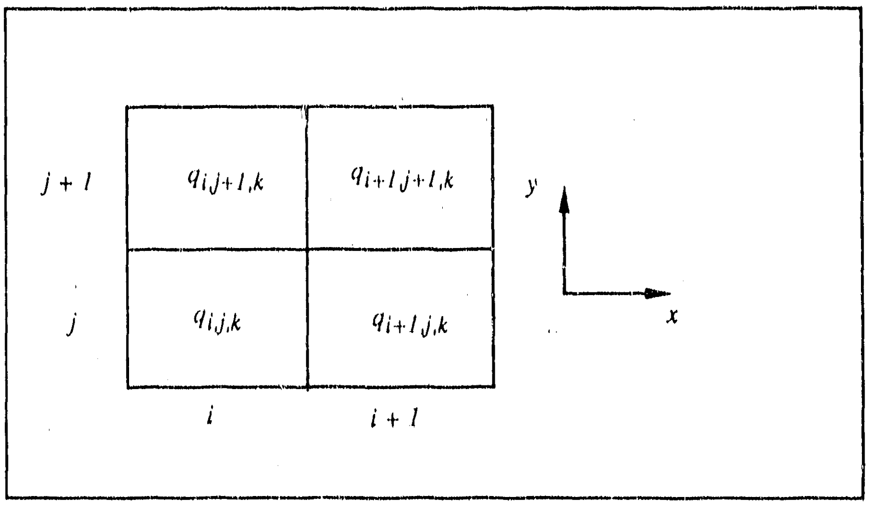


Figure A-1. Four active cells in the x-y plane.

Similarly, the value of q on the right-front (subscript RF) edge of cell (i,j,k) is given by

$$q_{RF} = \frac{1}{4}\left(q_{i,j,k} + q_{i+1,j,k} + q_{i+1,j+1,k} + q_{i,j+1,k} \right) \tag{A-2}$$

Velocity components, which live on cell faces, are often needed on cell edges, cell centers, and on other cell faces where they are not explicitly defined. Consider Figure A-2 as an example, which is a schematic of two active cells in the x-y plane. Only the u-velocity components are shown in this figure for simplicity. The value of u at the center (subscript C) of cell (i,j,k) is calculated by

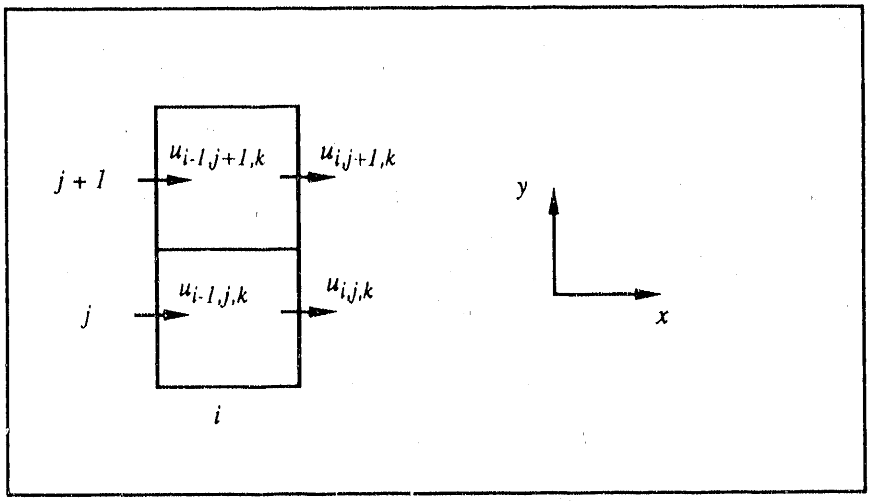


Figure A-2. Two active cells in the x-y plane.

$$u_C = \frac{1}{2}\left(u_{i,j,k} + u_{i-1,j,k} \right) \tag{A-3}$$

The value of u on the right-front (subscript RF) edge is computed from

$$u_{RF} = \frac{1}{2}\left(u_{i,j,k} + u_{i,j+1,k} \right) \tag{A-4}$$

The value of *u* on the front (subscript *F*) cell face is obtained from

$$u_F = \frac{1}{4}\left(u_{i,j,k} + u_{i,j+1,k} + u_{i-1,j+1,k} + u_{i-1,j,k} \right) \qquad \text{(A-5)}$$

The other two velocity components are averaged in a similar fashion.

APPENDIX B

WEIGHTED AVERAGING OF CENTERED AND UPWINDED VALUES

APPENDIX B

WEIGHTED AVERAGING OF CENTERED AND UPWINDED VALUES

This appendix describes, by example, how a quantity, q, is computed from a weighted average of its centered, q_c, and its "upwinded" value, q_u, at a given location. Figure B.1 is a schematic of two computational cells in the x-y plane. The quantity, q, is being convected by the velocity, u, and is to be computed on the interface between the two cells. q_L is the value of q at the center of the left cell, while q_R is the value of q at the center of the right cell. Note that the necessary variables may have to be calculated at the required locations using the averaging techniques described in Appendix A.

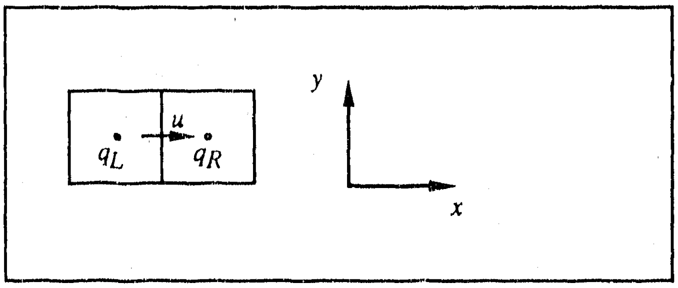


Figure B-1. Two computational cells in the x-y plane.

The "upwinded" value of q in Figure A-1 is q_L. Note, that if the direction of u is reversed, the "upwinded" value of q is q_R. q_u is thus given by

$$q_u = \left\{ \begin{array}{ll} q_L \ , & u > 0 \\ q_R \ , & u < 0 \end{array} \right\} \qquad \text{(B-1)}$$

The centered value of q on the interface is simply the average of q_L and q_R,

$$q_C = \frac{1}{2} (q_L + q_R) \qquad \text{(B-2)}$$

The weighted average of the centered value of q and the upwinded value of q on the interface between the cells is computed from

$$q = \alpha q_u + (1 - \alpha) q_c \qquad \text{(B-3)}$$

where α represents the fraction of upwinding, and $(1 - \alpha)$ represents the fraction of centering. For stability of the iteration scheme, α must exceed the greater of $\frac{|u|\Delta\tau}{\Delta x}$, $\frac{|v|\Delta\tau}{\Delta y}$, and $\frac{|w|\Delta\tau}{\Delta z}$. For stability of the energy equation, α must exceed the greater of $\frac{|u|\Delta t}{\Delta x}$, $\frac{|v|\Delta t}{\Delta y}$, and $\frac{|w|\Delta t}{\Delta z}$.

APPENDIX C

VARIABLE AND SUBROUTINE DICTIONARY

APPENDIX C

VARIABLE AND SUBROUTINE DICTIONARY

VARIABLES

SAMPLE ENTRY

VARIABLE RS DESCRIPTION

R	-	REAL
I	-	INTEGER
S	-	SINGLE VARIABLE
A	-	VARIABLE ARRAY

ALPHA	RS	UPWIND/CENTER-DIFFERENCING WEIGHTING FUNCTION = 0 MEANS CENTERED DIFFERENCING = 1 MEANS UPWIND DIFFERENCING
ALPHID	RS	INTERPOLATED DONOR CELL DIFFERENCING WEIGHTING PARAMETER (FRACTION DONOR CELL DIFF.)
ASQR	RS	WAVE PARAMETER, SQUARE OF WAVE SPEED
BETA	RS	MULTIPLIER. D*BETA/OMG IS DELP. IT INCLUDES OMG.
BETAID	RS	INTERPOLATED DONOR CELL DIFF. PARAMETER (FRACTION INTERPOLATED DONOR CELL DIFF.)
BCTE	RS	BETA, COEFF. OF THERMAL EXPANSION
BFX	RS	BUOYANCY FORCE IN THE X-DIRECTION
BFY	RS	BUOYANCY FORCE IN THE Y-DIRECTION
BFZ	RS	BUOYANCY FORCE IN THE Z-DIRECTION
C	C1	FIRST CHARACTER FROM A LINE OF INPUT DATA
CREU	RS	CELL REYNOLDS NUMBER BASED ON U-VELOCITY
CREV	RS	CELL REYNOLDS NUMBER BASED ON V-VELOCITY
CREW	RS	CELL REYNOLDS NUMBER BASED ON W-VELOCITY
CRXI8	RS	MULTIPLIER USED IN RZ GEOMETRY MOMENTUM EQUATION
CRXIH8	RS	MULTIPLIER USED IN RZ GEOMETRY MOMENTUM EQUATION
CF	RA	SPECIFIC HEAT CAPACITY ARRAY
CVREF	RS	REFERENCE SPECIFIC HEAT CAPACITY

CVMAX	RS	MAXIMUM SPECIFIC HEAT VALUE IN REGION
CWPRT	RS	THE INCREMENT OF CUMULATIVE TIME BETWEEN PLOT-TIME WRITES
CWPRT	RS	THE INCREMENT OF CUMULATIVE TIME BETWEEN DETAILED EDITS
CYCLE	IS	THE NUMBER OF TIME STEPS TAKEN THUS FAR
CYL	RS	CYLINDRICAL/RECTANGULAR COORD SYSTEM SWITCH = DP0 MEANS RETANGULAR = DP1 MEANS CYLINDRICAL
D	RS	THE DISCRETE DIVERGENCE OF THE VELOCITY FIELD AS MEASURED AT THE CENTER OF THE CURRENT CELL
DALPH	RS	THERMAL DIFFUSIVITY
DELP	RS	THE PRESSURE UPDATE FOR THE CURRENT CELL
DELT	RS	TIME STEP
DELTUB	RS	UPPER LIMIT FOR THE TIME STEP BASED ON VELMX = MIN (!DX/U!, !DY/V!, (DX*DY)**2/(2*NU*(DX**2+DY**2)))
DELX	RS	X-DIRECTION MESH-SPACING
DELY	RS	Y-DIRECTION MESH-SPACING
DIFFMX	RS	MAXIMUM VALUE OF THE THERMAL DIFFUSIVITY
DIFFX	RS	TEMPERATURE DIFFUSION TERM (X-DIRECTION)
DIFFY	RS	TEMPERATURE DIFFUSION TERM (Y-DIRECTION)
DIFFZ	RS	TEMPERATURE DIFFUSION TERM (Z-DIRECTION)
DP0	RS	VARIABLE HOLDING REAL ZERO (DBL PREC)
DP1	RS	VARIABLE HOLDING REAL ONE (DBL PREC)
DP1P2	RS	VARIABLE HOLDING REAL ONE POINT TWO (DBL PREC)
DP1P5	RS	VARIABLE HOLDING REAL ONE POINT FIVE (DBL PREC)
DP2	RS	VARIABLE HOLDING REAL TWO (DBL PREC)
DP8	RS	VARIABLE HOLDING REAL EIGHT (DBL PREC)
DTDXL	RS	DERIVATIVE OF T WRT X ON THE LEFT CELL FACE
DTDXR	RS	DERIVATIVE OF T WRT X ON THE RIGHT CELL FACE
DTDYF	RS	DERIVATIVE OF T WRT X ON THE FRONT CELL FACE
DTDYD	RS	DERIVATIVE OF T WRT X ON THE DERRIERE CELL FACE
DTDZB	RS	DERIVATIVE OF T WRT X ON THE BOTTOM CELL FACE
DTDZT	RS	DERIVATIVE OF T WRT X ON THE TOP CELL FACE
DTU	RS	ARTIFICIAL TIME STEP
DUB	RS	DIFFERENCE BETWEEN U(I,J,K) AND ITS BOTTOM NEIGHBOR

DUD	RS	DIFFERENCE BETWEEN U(I,J,K) AND ITS DERRIERE NEIGHBOR
DUDXR	RS	DERIVATIVE OF U WRT X ON THE RIGHT FACE OF THE X-MOMENTUM CELL
DUDXL	RS	DERIVATIVE OF U WRT X ON THE LEFT FACE OF THE X-MOMENTUM CELL
DUDYLF	RS	DERIVATIVE OF U WRT Y ON THE LEFT-FRONT CELL EDGE
DUDYRF	RS	DERIVATIVE OF U WRT Y ON THE RIGHT-FRONT CELL EDGE
DUDYRD	RS	DERIVATIVE OF U WRT Y ON THE RIGHT-DERRIERE CELL EDGE
DUDZLT	RS	DERIVATIVE OF U WRT Z ON THE LEFT-TOP CELL EDGE
DUDZRT	RS	DERIVATIVE OF U WRT Z ON THE RIGHT-TOP CELL EDGE
DUDYRB	RS	DERIVATIVE OF U WRT Z ON THE RIGHT-BOTTOM CELL EDGE
DUEDX	RS	CONVECTIVE TERM IN ENERGY EQ. DERIVATIVE OF (UE) WRT X
DUF	RS	DIFFERENCE BETWEEN U(I,J,K) AND ITS FRONT NEIGHBOR
DUL	RS	DIFFERENCE BETWEEN U(I,J,K) AND ITS LEFT NEIGHBOR
DUR	RS	DIFFERENCE BETWEEN U(I,J,K) AND ITS RIGHT NEIGHBOR
DUT	RS	DIFFERENCE BETWEEN U(I,J,K) AND ITS TOP NEIGHBOR
DVB	RS	DIFFERENCE BETWEEN V(I,J,K) AND ITS BOTTOM NEIGHBOR
DVD	RS	DIFFERENCE BETWEEN V(I,J,K) AND ITS DERRIERE NEIGHBOR
DVDXDR	RS	DERIVATIVE OF V WRT X ON THE DERRIERE-RIGHT CELL EDGE
DVDXFR	RS	DERIVATIVE OF V WRT X ON THE FRONT-RIGHT CELL EDGE
DVDXFL	RS	DERIVATIVE OF V WRT X ON THE FRONT-LEFT CELL EDGE
DVDYD	RS	DERIVATIVE OF V WRT Y ON THE DERRIERE FACE OF THE Y-MOMENTUM CELL
DVDYF	RS	DERIVATIVE OF V WRT Y ON THE FRONT FACE OF THE Y-MOMENTUM CELL
DVDZDT	TS	DERIVATIVE OF V WRT Z ON THE DERRIERE-TOP CELL EDGE
DVDZFB	TS	DERIVATIVE OF V WRT Z ON THE FRONT-BOTTOM CELL EDGE
DVDZFT	TS	DERIVATIVE OF V WRT Z ON THE FRONT-TOP CELL EDGE
DVF	RS	DIFFERENCE BETWEEN V(I,J,K) AND ITS FRONT NEIGHBOR
DVL	RS	DIFFERENCE BETWEEN V(I,J,K) AND ITS LEFT NEIGHBOR
DVR	RS	DIFFERENCE BETWEEN V(I,J,K) AND ITS RIGHT NEIGHBOR
DVT	RS	DIFFERENCE BETWEEN V(I,J,K) AND ITS TOP NEIGHBOR
DWB	RS	DIFFERENCE BETWEEN W(I,J,K) AND ITS BOTTOM NEIGHBOR

DWD	RS	DIFFERENCE BETWEEN W(I,J,K) AND ITS DERRIERE NEIGHBOR
DWDXBR	RS	DERIVATIVE OF W WRT X ON THE BOTTOM-RIGHT CELL EDGE
DWDXBF	RS	DERIVATIVE OF W WRT X ON THE BOTTOM-FRONT CELL EDGE
DWDXTR	RS	DERIVATIVE OF W WRT X ON THE TOP-RIGHT CELL EDGE
DWDYBF	TS	DERIVATIVE OF W WRT Y ON THE BOTTOM-FRONT CELL EDGE
DWDYTD	TS	DERIVATIVE OF W WRT Y ON THE TOP-DERRIERE CELL EDGE
DWDYTF	TS	DERIVATIVE OF W WRT Y ON THE TOP-FRONT CELL EDGE
DWDZB	RS	DERIVATIVE OF W WRT Z ON THE BOTTOM FACE OF THE Y-MOMENTUM CELL
DWDZT	RS	DERIVATIVE OF W WRT Z ON THE TOP FACE OF THE Y-MOMENTUM CELL
DVEDX	RS	CONVECTIVE TERM IN ENERGY EQ. DERIVATIVE OF (VE) WRT X
DWEDX	RS	CONVECTIVE TERM IN ENERGY EQ. DERIVATIVE OF (WE) WRT X
DWF	RS	DIFFERENCE BETWEEN W(I,J,K) AND ITS FRONT NEIGHBOR
DWL	RS	DIFFERENCE BETWEEN W(I,J,K) AND ITS LEFT NEIGHBOR
DWR	RS	DIFFERENCE BETWEEN W(I,J,K) AND ITS RIGHT NEIGHBOR
DWT	RS	DIFFERENCE BETWEEN W(I,J,K) AND ITS TOP NEIGHBOR
DZEPS	RS	DZRO*EPSI, THE LARGEST TOLERABLE VALUE OF D IN ANY CELL
DZRO	RS	BENCHMARK VALUE OF D
E	RA	ENERGY ARRAY
ECOEFF1	RS	LINEAR COEFF. IN EQ. OF STATE RELATING TEMP. TO ENERGY
ECOEFF2	RS	QUADRATIC COEFF. IN EQ. OF STATE RELATING TEMP. TO ENERGY
ENLOOP	RS	TIME ON THE INTERNAL CLOCK AT THE END OF LOOP
ENTIME	RS	TIME ON INTERNAL CLOCK AT END OF RUN
EPSI	RS	CONVERGNCE TOLERANCE FOR THE SOR ITERATION
FK	RA	THERMAL CONDUCTIVITY ARRAY
FKB	RS	THERMAL CONDUCTIVITY ON THE BOTTOM CELL FACE
FKD	RS	THERMAL CONDUCTIVITY ON THE DERRIERE CELL FACE
FKF	RS	THERMAL CONDUCTIVITY ON THE FRONT CELL FACE
FKL	RS	THERMAL CONDUCTIVITY ON THE LEFT CELL FACE
FKMAX	RS	MAXIMUM VALUE OF THE THERMAL CONDUCTIVITY

FKR	RS	THERMAL CONDUCTIVITY ON THE RIGHT CELL FACE
FKREF	RS	REFERENCE THERMAL CONDUCTIVITY
FKT	RS	THERMAL CONDUCTIVITY ON THE BOTTOM CELL FACE
FL	RS	LATENT HEAT OF FUSION
FLCHAR	RS	CHARACTERISTIC LENGTH

FLG	RS	CONVERGNCE CONDITION ATTAINMENT FLAG:
		FLG = DP1 MEANS CONVERGENCE CONDITION VIOLATED IN SOME
		FLG = DP0 MEANS CONVERGENCE CONDITION ATTAINED.

FUC	RS	WEIGHTED AVERAGE OF CENTERED AND UPWIND DIFFERENCING IN THE RZ SUMMAND OF THE CONVECTIVE TERM OF THE U-DIRECTION MOMENTUM EQUATION
FUX	RS	WEIGHTED AVERAGE OF CENTERED UPWIND DIFFERENCING IN THE D(U**2)/DX SUMMAND OF THE CONVECTIVE TERM OF THE U-DIRECTION MOMENTUM EQUATION
FUY	RS	WEIGHTED AVERAGE OF AND UPWIND DIFFERENCING IN THE D(UV)/DX SUMMAND OF THE CONVECTIVE TERM OF THE U-MOMENTUM EQUATION
FUZ	RS	WEIGHTED AVERAGE OF CENTERED AND UPWIND DIFFERENCING IN THE D(UW)/DZ SUMMAND OF THE CONVECTIVE TERM OF THE U-DI MOMENTUM EQUATION
FVC	RS	WEIGHTED AVERAGE OF CENTERED AND UPWIND DIFFERENCING IN THE RZ SUMMAND OF THE CONVECTIVE TERM OF THE V-DIRECTIO MOMENTUM EQUATION
FVX	RS	WEIGHTED AVERAGE OF CENTERED AND UPWIND DIFFERENCING IN THE D(UV)/DX SUMMAND OF THE CONVECTIVE TERM OF THE V-DIRECTION MOMENTUM EQUATION
FVY	RS	WEIGHTED AVERAGE OF CENTERED AND UPWIND DIFFERENCING IN THE D(V**2)/DX SUMMAND OF THE CONVECTIVE TERM OF THE V-DIRECTION MOMENTUM EQUATION

FVZ	RS	WEIGHTED AVERAGE OF CENTERED AND UPWIND DIFFERENCING IN THE D(VW)/DZ SUMMAND OF THE CONVECTIVE TERM OF THE V-DIRECTION MOMENTUM EQUATION
FWC	RS	WEIGHTED AVERAGE OF CENTERED AND UPWIND DIFFERENCING IN THE RZ SUMMAND OF THE CONVECTIVE TERM OF THE W-DIRECTION MOMENTUM EQUATION
FWX	RS	WEIGHTED AVERAGE OF CENTERED AND UPWIND DIFFERENCING IN THE D(UW)/DX SUMMAND OF THE CONVECTIVE TERM OF THE W-DIRECTION MOMENTUM EQUATION
FWY	RS	WEIGHTED AVERAGE OF CENTERED AND UPWIND DIFFERENCING IN THE D(VW)/DY SUMMAND OF THE CONVECTIVE TERM OF THE W-DIRECTION MOMENTUM EQUATION
FWZ	RS	WEIGHTED AVERAGE OF CENTERED AND UPWIND DIFFERENCING IN THE D(W**2)/DZ SUMMAND OF THE CONVECTIVE TERM OF THE W-DIRECTION MOMENTUM EQUATION
GX	RS	X-DIRECTION COMPONENT OF GRAVITATIONAL FORCE
GY	RS	Y-DIRECTION COMPONENT OF GRAVITATIONAL FORCE
GZ	RS	Z-DIRECTION COMPONENT OF GRAVITATIONAL FORCE
I	IS	DO-LOOP INDEX IBAR
IBAR	IS	NUMBER OF MESH CELLS IN THE X-DIRECTION IN THE REGION
ID	IA	INDIRECT ADDRESSING ARRAY FOR ALL INTERIOR CELLS ID(1,N) ==> I INDEX FOR Nth CELL ID(2,N) ==> J INDEX FOR Nth CELL ID(3,N) ==> K INDEX FOR Nth CELL
IDBC	IA	INDIRECT ADDRESSING ARRAY FOR ALL BOUNDARY CELLS IDBC(1,N) ==> I INDEX FOR Nth CELL IDBC(2,N) ==> J INDEX FOR Nth CELL IDBC(3,N) ==> K INDEX FOR Nth CELL
IENERGY	IS	ENERGY SOLVER FLAG, =1 SOLVE ENERGY EQ. =0 NO ENERGY EQ.

IFIMP	IS	FLAG DETERMINING FULLY/LINEARLY IMPLICIT OPTION =1 ==> LINEARLY IMPLICIT =0 ==> FULLY IMPLICIT
IGUESS	IS	FLAG DETERMINING INITIAL GUESS FOR FLOW VELOCITY =0 ==> EXPLICIT APPROX. TO MOM. E.Q =1 ==> EXTRAPOLATION FROM PREV. 2 TIME LEVELS =2 ==> PREVIOUS TIME VELOCITY
IM1	IS	IMAX-1
IM2	IS	IMAX-2
IMAP	IA	INDIRECT ADDRESSING ARRAY (4 DIMENSIONAL) IMAP(I,J,K,1) = 0 ==> CELL (I,J,K) IS INACTIVE IMAP(I,J,K,1) = 1 ==> CELL (I,J,K) IS ACTIVE IMAP(I,J,K,2) = 0 ==> BOUNDARY CELL (I,J,K) IS A NO-SLIP WALL IMAP(I,J,K,2) = 1 ==> BOUNDARY CELL (I,J,K) IS A FREE-SLIP WALL IMAP(I,J,K,2) = 2 ==> BOUNDARY CELL (I,J,K) IS A WALL WITH SPECIFIED VELOCITIES
IMAX	IS	NUMBER OF CELLS IN THE X-DIRECTION OF THE AUGMENTED REGION INCLUDES THE BOUNDARY CELLS.
IMELT	IS	MELTING FLAG - IMELT=0, NO MELTING - IMELT=1, MELTING IS ALLOWED
IMLT	IS	LOGICAL UNIT NUMBER FOR THE MELT FRONT DATA FILE "mltdat"
INDEX	IS	USED TO COMBINE THE CONTINUATIVE AND PERIODIC B. C. LOOPS
IOIN	IS	LOGICAL UNIT NUMBER FOR THE INPUT DEVICE
IOOUT	IS	LOGICAL UNIT NUMBER FOR OUTPUT FILE "outdat"
IOPLT	IS	LOGICAL UNIT NUMBER FOR OUTPUT FILE "pltdat"
IPRS	IS	LOGICAL UNIT NUMBER FOR PRESSURE DATA FILE "prsdat"
IRIN	IS	LOGICAL UNIT NUMBER FOR THE RESTART INPUT DEVICE
IROUT	IS	LOGICAL UNIT NUMBER FOR THE RESTART OUTPUT DEVICE
ISTART	IS	STARTING VALUE FOR DO-LOOP WITH INDEX I
ISTOP	IS	STOPPING VALUE FOR DO-LOOP WITH INDEX I
ISTRM1	IS	ISTART - 1
ITER	IS	NUMBER OF SOR ITERATIONS TAKEN ON A TIME STEP

ITMP IS LOGICAL UNIT NUMBER FOR TEMPERATURE DATA FILE "tmpdat"

ITMPB IS BOTTOM TEMP. BOUNDARY COND. FLAG
=1 CONSTANT TEMP.
=2 CONSTANT HEAT FLUX/SYMETRIC/ADIABATIC
=3 CONVECTIVE/MIXED B.C.

ITMPD IS DERRIERE TEMP. BOUNDARY COND. FLAG
=1 CONSTANT TEMP.
=2 CONSTANT HEAT FLUX/SYMETRIC/ADIABATIC
=3 CONVECTIVE/MIXED B.C.

ITMPF IS FRONT TEMP. BOUNDARY COND. FLAG
=1 CONSTANT TEMP.
=2 CONSTANT HEAT FLUX/SYMETRIC/ADIABATIC
=3 CONVECTIVE/MIXED B.C.

ITMPL IS LEFT TEMP. BOUNDARY COND. FLAG
=1 CONSTANT TEMP.
=2 CONSTANT HEAT FLUX/SYMETRIC/ADIABATIC
=3 CONVECTIVE/MIXED B.C.

ITMPR IS RIGHT TEMP. BOUNDARY COND. FLAG
=1 CONSTANT TEMP.
=2 CONSTANT HEAT FLUX/SYMETRIC/ADIABATIC
=3 CONVECTIVE/MIXED B.C.

ITMPT IS TOP TEMP. BOUNDARY COND. FLAG
=1 CONSTANT TEMP.
=2 CONSTANT HEAT FLUX/SYMETRIC/ADIABATIC
=3 CONVECTIVE/MIXED B.C.

ITSUM IS THE TOTAL NUMBER OF ITERATIONS TAKEN ON ALL CYCLES

IVEL IS LOGICAL UNIT NUMBER FOR THE VELOCITY DATA FILE "veldat"

IVISC IS VARIABLE VISCOSITY FLAG, =1 VARIABLE VISCOSITY FORMULATION, =0 CONSTANT VISCOSITY FORMULATION

J IS DO-LOOP INDEX

JBAR IS NUMBER OF MESH CELLS IN THE X-DIRECTION IN THE REGION

JM1 IS JMAX-1

JM2 IS JMAX-2

JMAX	IS	NUMBER OF CELLS IN THE Y-DIRECTION OF THE AUGMENTED REGION INCLUDES THE BOUNDARY CELLS.
JSTART	IS	STARTING VALUE FOR DO-LOOP WITH INDEX J
JSTOP	IS	STOPPING VALUE FOR DO-LOOP WITH INDEX J
JSTRM1	IS	JSTART - 1
K	IS	DO-LOOP INDEX
KBAR	IS	NUMBER OF MESH CELLS IN THE Z-DIRECTION IN THE REGION
KM1	IS	KMAX-1 C
KM2	IS	KMAX-2
KMAX	IS	NUMBER OF CELLS IN THE Z-DIRECTION OF THE AUGMENTED REGION INCLUDES THE BOUNDARY CELLS.
NAME	IA	HOLDS CHARACTER DATA WITH TITLE OF PROBLEM BEING RUN.
NBC	IS	NUMBER OF BOUNDARY CELLS
NCELLS	IS	NUMBER OF ACTIVE CELLS IN COMPUTATIONAL REGION
NU	RS	VISCOSITY
NXD	IS	ABSOLUTE MAXIMUM X DIMENSION
NYD	IS	ABSOLUTE MAXIMUM Y DIMENSION
NZD	IS	ABSOLUTE MAXIMUM Z DIMENSION
OMG	RS	OMEGA, SAFETY FACTOR
OMG1	RS	OMEGA1, SAFETY FACTOR FOR PHYSICAL TIME STEP
OMG2	RS	OMEGA2, SAFETY FACTOR FOR ARTIFICIAL TIME STEP
P	RA	PRESSURE FIELD; DOUBLY INDEXED ARRAY
QDOT	RA	ENERGY GENERATION ARRAY
RC	RA	ARRAY CONTAINING X-VALUES AT CELL CENTERS FOR USE IN THE 2-D CYLINDRICAL COORDINATES FORMULATION
RR	RA	ARRAY CONTAINING X-VALUES AT CELL RIGHT FACES FOR USE IN THE 2-D CYLINDRICAL COORDINATES FORMULATION
RDX	RS	1/DX. USED TO AVOID FLOATING POINT DIVISION IN LOOPS
RDX4	RS	1/(4*DX)
RDXSQ	RS	1/(DX*DX)
RDY	RS	1/DY. IBID.
RDY4	RS	1/(4*DY)
RDYSQ	RS	1/(DY*DY)
RDZ	RS	1/DZ. IBID.

RDZ4	RS	1/(4*DZ)
RDZSQ	RS	1/(DY*DZ)
RHO	RS	FLUID DENSITY
RNUDT	RS	VISCOSITY OF THE DERRIERE-TOP CELL EDGE
RNUFB	RS	VISCOSITY OF THE FRONT-BOTTOM CELL EDGE
RNUFT	RS	VISCOSITY OF THE FRONT-TOP CELL EDGE
RNULF	RS	VISCOSITY OF THE LEFT-FRONT CELL EDGE
RNULT	RS	VISCOSITY OF THE LEFT-TOP CELL EDGE
RNURB	RS	VISCOSITY OF THE RIGHT-BOTTOM CELL EDGE
RNURD	RS	VISCOSITY OF THE RIGHT-TOP CELL EDGE
RNURF	RS	VISCOSITY OF THE RIGHT-FRONT CELL EDGE
RNURT	RS	VISCOSITY OF THE RIGHT-TOP CELL EDGE
RXIDX2	RS	MULTIPLIER IN RZ GEOMETRY MOMENTUM EQUATION
RXIHDX	RS	MULTIPLIER IN RZ GEOMETRY MOMENTUM EQUATION
RXISQ	RS	MULTIPLIER IN RZ GEOMETRY MOMENTUM EQUATION
SIGMA	RA	ELECTRICAL CONDUCTIVITY ARRAY
STARTM	RS	TIME ON THE INTERNAL CLOCK WHEN THE RUN WAS STARTED
STLOOP	RS	TIME ON THE INTERNAL CLOCK WHEN THE LOOP WAS STARTED
SUR	RS	SUM OF U(I,J,K) AND ITS RIGHT NEIGHBOR
SUF	RS	SUM OF U(I,J,K) AND ITS FRONT NEIGHBOR
SUD	RS	SUM OF U(I,J,K) AND ITS DERRIERE NEIGHBOR
SULF	RS	SUM OF U(I-1,J,K) AND ITS FRONT NEIGHBOR SULT RS SUM OF U(I-1,J,K) AND ITS TOP NEIGHBOR
SUL	RS	SUM OF U(I,J,K) AND ITS LEFT NEIGHBOR
SUT	RS	SUM OF U(I,J,K) AND ITS TOP NEIGHBOR
SUB	RS	SUM OF U(I,J,K) AND ITS BOTTOM NEIGHBOR
SVR	RS	SUM OF V(I,J,K) AND ITS RIGHT NEIGHBOR
SVF	RS	SUM OF V(I,J,K) AND ITS FRONT NEIGHBOR
SVB	RS	SUM OF V(I,J,K) AND ITS DERRIERE NEIGHBOR
SVDR	RS	SUM OF V(I,J-1,K) AND ITS RIGHT NEIGHBOR
SVDT	RS	SUM OF V(I,J-1,K) AND ITS TOP NEIGHBOR
SVL	RS	SUM OF V(I,J,K) AND ITS LEFT NEIGHBOR
SVT	RS	SUM OF V(I,J,K) AND ITS TOP NEIGHBOR
SVB	RS	SUM OF V(I,J,K) AND ITS BOTTOM NEIGHBOR

SWR	RS	SUM OF W(I,J,K) AND ITS RIGHT NEIGHBOR
SWF	RS	SUM OF W(I,J,K) AND ITS FRONT NEIGHBOR
SWB	RS	SUM OF W(I,J,K) AND ITS DERRIERE NEIGHBOR
SWBF	RS	SUM OF W(I,J,K-1) AND ITS FRONT NEIGHBOR
SWBR	RS	SUM OF W(I,J,K-1) AND ITS RIGHT NEIGHBOR
SWL	RS	SUM OF W(I,J,K) AND ITS LEFT NEIGHBOR
SWT	RS	SUM OF W(I,J,K) AND ITS TOP NEIGHBOR
SWB	RS	SUM OF W(I,J,K) AND ITS BOTTOM NEIGHBOR
SXXR	RS	VISCOUS STRESS TERM LOCATED ON THE RIGHT FACE OF THE X-MOMENTUM CELL
SXXL	RS	VISCOUS STRESS TERM LOCATED ON THE LEFT FACE OF THE X-MOMENTUM CELL
SXYLF	RS	VISCOUS STRESS TERM ON THE LEFT-FRONT CELL EDGE
SXYRD	RS	VISCOUS STRESS TERM ON THE RIGHT-DERRIERE CELL EDGE
SXYRF	RS	VISCOUS STRESS TERM ON THE RIGHT-FRONT CELL EDGE
SXZLT	RS	VISCOUS STRESS TERM ON THE LEFT-TOP CELL EDGE
SXZRB	RS	VISCOUS STRESS TERM ON THE RIGHT-BOTTOM CELL EDGE
SXZRT	RS	VISCOUS STRESS TERM ON THE RIGHT-TOP CELL EDGE
SYYD	RS	VISCOUS STRESS TERM LOCATED ON THE DERRIERE FACE OF THE X-MOMENTUM CELL
SYYF	RS	VISCOUS STRESS TERM LOCATED ON THE FRONT FACE OF THE X-MOMENTUM CELL
SYZDT	RS	VISCOUS STRESS TERM ON THE DERRIERE-TOP CELL EDGE
SYZFB	RS	ISCOUS STRESS TERM ON THE FRONT-BOTTOM CELL EDGE
SYZFT	RS	VISCOUS STRESS TERM ON THE FRONT-TOP CELL EDGE
SZZB	RS	VISCOUS STRESS TERM LOCATED ON THE BOTTOM FACE OF THE X-MOMENTUM CELL
SZZT	RS	VISCOUS STRESS TERM LOCATED ON THE TOP FACE OF THE X-MOMENTUM CELL
T	RS	CUMULATIVE TIME
TAVE	RS	AVERAGE TEMP. IN COMPUTATIONAL REGION
TCRNTU	RS	COURANT TIME STEP LIMIT BASED ON U-VELOCITY
TCRNTV	RS	COURANT TIME STEP LIMIT BASED ON V-VELOCITY
TCRNTW	RS	COURANT TIME STEP LIMIT BASED ON W-VELOCITY
TMP	RA	TEMPERATURE ARRAY
TNU	RA	TEMPERATURE DEPENDENT VISCOSITY ARRAY

TQB	RS	TEMP. B.C. PARAMETER FOR BOTTOM OF REGION
TQD	RS	TEMP. B.C. PARAMETER FOR DERRIERE OF REGION
TQF	RS	TEMP. B.C. PARAMETER FOR FRONT OF REGION
TQL	RS	TEMP. B.C. PARAMETER FOR LEFT OF REGION
TQR	RS	TEMP. B.C. PARAMETER FOR RIGHT OF REGION
TQT	RS	TEMP. B.C. PARAMETER FOR BOTTOM OF REGION
TREF	RS	REFERENCE TEMPERATURE
TWFIN	RS	FINISH TIME. STOP CALCULATION WHEN T = TWFIN.
TWPLT	RS	PLOTTING TIME. WRITE A PLOT WHENEVER T = TWPLT.
TWPRT	RS	EDITTING TIME. WRITE OUTPUT EDIT WHENEVER T = TWPRT.
U	RA	X-DIRECTION VELOCITY ITERATE VECTOR FOR SOR.
UI	RS	UNIFORM INITIAL U-VELOCITY
UMAX	RS	MAXIMUM VALUE OF THE U-VELOCITY IN THE REGION
UN	RA	X-DIRECTION VELOCITY FIELD A TIME-STEP N.
US	RA	X-DIRECTION SPECIFIED VELOCITY FIELD
UTLDA	RA	X-COMPONENT OF VELOCITY CONSISTING OF THE VELOCITY OF THE PREV. TIME STEP PLUS THE CONVECTIVE AND VISCOUS FLUXES ASSOCIATED WITH THE CURRENT ITERATE FOR THE VELOCITY.
V	RA	Y-DIRECTION VELOCITY ITERATE VECTOR FOR SOR.
VELMX	RS	MAXIMUM EXPECTED VELOCITY AT ANY POINT AND TIME
VELMX1	RS	THE MAXIMUM MESH-SPACING DIVIDED BY VELMX
VI	RS	UNIFORM INITIAL V-VELOCITY
VISX	RS	VISCOUS DIFFUSION TERM IN THE X-DIRECTION MOMENTUM EQUATION
VISY	RS	VISCOUS DIFFUSION TERM IN THE Y-DIRECTION MOMENTUM EQUATION
VISZ	RS	VISCOUS DIFFUSION TERM IN THE Z-DIRECTION MOMENTUM EQUATION
VMAX	RS	MAXIMUM VALUE OF THE V-VELOCITY IN THE REGION
VN	RA	Y-DIRECTION VELOCITY FIELD A TIME-STEP N.
VS	RA	Y-DIRECTION SPECIFIED VELOCITY FIELD
VTLDA	RA	Y-COMPONENT OF VELOCITY CONSISTING OF THE VELOCITY OF THE PREV. TIME STEP PLUS THE CONVECTIVE AND VISCOUS FLUXES ASSOCIATED WITH THE CURRENT ITERATE OF THE VELOCITY.

W	RA	Z-DIRECTION VELOCITY ITERATE VECTOR FOR SOR.
WI	RS	UNIFORM INITIAL W-VELOCITY
WMAX	RS	MAXIMUM VALUE OF THE W-VELOCITY IN THE REGION
WN	RA	W-DIRECTION VELOCITY FIELD A TIME-STEP N.
WTLDA	RA	Z-COMPONENT OF VELOCITY CONSISTING OF THE VELOCITY OF THE PREV. TIME STEP PLUS THE CONVECTIVE AND VISCOUS FLUXES ASSOCIATED WITH THE CURRENT ITERATE OF THE VELOCITY.

SUBROUTINES

ADDRESS	==>	DEFINES THE INDIRECT ADDRESSING SCHEME
ADVANCE	==>	UPDATES VARIABLES TO THE NEW TIME LEVEL
BNDRY	==>	SETS THE BOUNDARY CONDITIONS
CONVERG	==>	CHECKS FOR CONVERGENCE OF THE ITERATION SCHEME
DENCHK	==>	COMPUTES THE DEVIATION FROM INCOMPRESSIBILITY
DUMP	==>	PERFORMS BINARY DUMP OF DATA FOR RESTART
ENERGY	==>	SOLVES DISCRETIZED FORM OF ENERGY EQUATION (EXPLICIT)
FLUX	==>	COMPUTES THE VELOCITY CONTRIBUTIONS DUE TO THE CONVECTIVE, VISCOUS, AND BODY FORCE TERMS IN THE MOMEMTUM EQUATION
GUESS	==>	SETS THE INITIAL VELOCITY GUESS FOR THE ITERATION SCHEME --> 1.) EXPLICIT APPROX. TO MOM. EQ. 2.) OLD TIME VELOCITIES 3.) EXTRAPOLATION FROM 2 PREV. TIME LEVELS
INITLZ	==>	INITIALIZES PROGRAM VARIABLES
INPUT	==>	INPUTS THE NECESSARY VARIABLES
MELT	==>	ALLOWS CELL MELTING AND FREEZING
PRNT	==>	PRINTS INFORMATION FOR OUTPUT AND PLOTTING
RSTART	==>	READS BINARY DATA FILE FOR PROGRAM RESTART

STATE ==> USES EQ. OF STATE TO COMPUTE TEMP. OR ENERGY
ALSO COMPUTES TEMP. DEPENDENT FLUID PROPERTIES

TBNDRY ==> SETS TEMPERATURE BOUNDARY CONDITIONS

TIMER ==> TIMES PROGRAM

UPDATE ==> UPDATES PRESSURES AND VELOCITIES ACCORDING TO
FULLY/LINEARLY IMPLICIT ARTIFICIAL COMPRESSIBILITY
ITERATION SCHEME WITH DAMPING

APPENDIX D

SAMPLE INPUT FILE

APPENDIX D

SAMPLE INPUT FILE

```
MAGMA TEST PROBLEM - 3B
C..4..7.10..3..6..20
            0   IRSTRT    RESTART PARAMETER =0,INPUT FILE;=1,RESTART FILE
        1.000   ALPHA     WEIGHTING FACTOR FOR AVE. UPWIND AND CENTER
                          DIFF.
      1.0E+00   BCTE      BETA--COEFFICIENT OF THERMAL EXPANSION
    1.00D+00    CVREF     REFERENCE SPECIFIC HEAT CAPACITY
         0.D0   CYL       CYLINDRICAL/RECTANGULAR COORDINATES INDICATOR
      1.0E+00   DELT      TIME STEP
 2.500000E-02   DELX      X-DIRECTION MESH SPACING
 2.500000E-02   DELY      Y-DIRECTION MESH SPACING
    -2.50E-02   DELZ      Y-DIRECTION MESH SPACING (<0 for 2-D)
         1.D0   DZRO      BENCHMARK VALUE FOR DIVERGENCE OF VELOCITY
                          FIELD
     1.00D+00   ECOEF1    COEFF. IN ENERGY EQ. OF STATE (LINEAR TERM)
      0.0D-00   ECOEF2    COEFF. IN ENERGY EQ. OF STATE (QUADRATIC TERM)
     1.00D+00   ECOEF3    COEFF. IN ENERGY EQ. OF STATE (LINEAR TERM)
      0.0D-00   ECOEF4    COEFF. IN ENERGY EQ. OF STATE (QUADRATIC TERM)
      1.0E-05   EPSI      CONVERGENCE CONDITION
 1.00000E-00    FKREF     REFERENCE THERMAL CONDUCTIVITY
      0.0D+00   FL        SPECIFIC LATENT HEAT OF FUSION
        1.0D0   FLCHAR    CHARACTERISTIC LENGTH
         0.D0   GX        BODY ACCELERATION IN X DIRECTION
       9.81D0   GY        BODY ACCELERATION IN Y DIRECTION
         0.D0   GZ        BODY ACCELERATION IN Z DIRECTION
 0.71000D-00    NU        VISCOSITY
      1.47500   OMG       EFFECTIVE DAMPING COEFFICIENT
      0.80D00   OMG1      SAFETY FACTOR FOR PHYSICAL TIME STEP
      0.80D00   OMG2      SAFETY FACTOR FOR ARTIFICIAL TIME STEP
      1.0D00    RHO       REFERENCE DENSITY
      1.0E-03   TOL       TOLERANCE OF DU/DTAU & DV/DTAU
  661.88D+00    TMPI      INITIAL TEMPERATURE
```

```
     0.0E+00   TQB      TEMP./HEAT FLUX FOR BOTTOM BOUNDARY COND.
     0.0D+00   TQD      TEMP./HEAT FLUX FOR DERRIERE BOUNDARY COND.
     0.0E+00   TQF      TEMP./HEAT FLUX FOR FRONT BOUNDARY COND.
     0.0E+00   TQH      HEAT TRANSFER COEFF. FOR MIXED TYPE B.C.
  1.0238E+03   TQL      TEMP./HEAT FLUX FOR LEFT BOUNDARY COND.
     3.0E+02   TQR      TEMP./HEAT FLUX FOR RIGHT BOUNDARY COND.
     0.0E+00   TQT      TEMP./HEAT FLUX FOR TOP BOUNDARY COND.
  661.880.00   TREF     REFERENCE TEMPERATURE
 1.00000D-00   TWFIN    STOPPING TIME
        0.0D0  UI       INITIAL U-VELOCITY
     2.5D+01   VELMX    MAXIMUM EXPECTED VELOCITY
        0.0D0  VI       INITIAL Y-VELOCITY
        0.0D0  WI       INITIAL Z-VELOCITY
C
C INTEGER SCALARS
C
          100  IFREQ    PLOT EVERY IFREQ CYCLES
           40  IBAR     MAX. NO. OF INTERIOR CELLS IN X-DIRECTION
           40  BAR      MAX. NO. OF INTERIOR CELLS IN Y-DIRECTION
            1  KBAR     MAX. NO. OF INTERIOR CELLS IN Z-DIRECTION
            1  IENRGY   =0 NO ENERGY CALCULATIONS, =1 ENERGY CALC.S
            0  IFIMP    =0 FULLY IMPLCT =1 LINEARLY IMPLCT =2 SEMI-IMPLCT
            1  IGUESS   =0 NORMAL GUESS, =1 OLD VEL., =2 EXTRAPOLATED
                        VEL.
            0  IMELT    =0 NO MELTING, =1 MELTING ALLOWED
            2  ITMPB    =1 CONST T, =2 CONST Q, =3 CONVECTIVE B.C.
            2  ITMPD    =1 CONST T, =2 CONST Q, =3 CONVECTIVE B.C.
            2  ITMPF    =1 CONST T, =2 CONST Q, =3 CONVECTIVE B.C.
            1  ITMPL    =1 CONST T, =2 CONST Q, =3 CONVECTIVE B.C.
            1  ITMPR    =1 CONST T, =2 CONST Q, =3 CONVECTIVE B.C.
            2  ITMPT    =1 CONST T, =2 CONST Q, =3 CONVECTIVE B.C.
            0  IVISC    =0 TURNS OFF VARIABLE VISCOSITY FORMULATION
```

DATE
FILMED

6/9/92