FEM simulation of Double Diffusive Convective Flows using an Object Oriented Approach

Rajiv Sampath and Nicholas Zabaras
Sibley School of Mechanical and Aerospace Engineering
188 Frank H. T. Rhodes Hall
Cornell University
Ithaca, NY 14853-3801
June 26, 1997

Abstract

We present results for numerical simulations of double diffusive convection in a rectangular cavity in a variety of situations. The simulations are performed using a streamline-upwind Petrov-Galerkin penalty finite-element method. The fluid is modeled as a Boussinesq fluid and the Navier Stokes equations are solved in conjunction with the time dependent scalar-convection diffusion equations for temperature and solute concentration fields. The whole simulation is carried out in an object oriented environment and some details on the various classes developed are presented.

1 Introduction

Density induced convective motions are of significance in a variety of industrial applications and natural phenomena. The convective motions generated when more than one property affects the density variation, is termed double-diffusive convection and is described in some detail in [1, 2, 3]. Theories for the study of double diffusive convection were first developed in the field of oceanography in the late 1950s. Since then, research in this area has increased exponentially. Recent applications of these phenomena include solidification and industrial crystal growth processes.

As may be expected, experimental and field work in this area is difficult and limited; therefore numerical techniques should provide a useful tool in the study of the dynamics of double diffusive convective flows. Surprisingly however, until recently not much numerical work has been done in this area, exceptions being [1, 4, 5].

In this report, we present the results of numerical simulations of double diffusive convection carried out using a penalty FEM formulation based on a streamline-upwind Petrov-Galerkin formulation. Numerical calculations and comparisons with other numerical and experimental results are discussed and potential extensions of the present work are identified.
2 Definition of the double diffusive convection problem

We here consider the flow of a viscous, incompressible fluid with constant physical properties in which both heat and solute diffusion combine to affect the density stratification inside the domain. We further assume that the Boussinesq approximation is valid. The governing equations are the Navier-Stokes equations of conservation of linear momentum, the continuity equation, solute conservation equation and the energy equation, where we assume that the internal energy is a function of temperature only. With the above assumptions, the governing equations for the temperature $T$, solute concentration $c$, velocity field $v$ are summarized in Box 2.1. For simplicity, we assume that the temperature boundary conditions are either Dirichlet or Neumann type conditions.

Box 2.1
Governing equations for the double diffusive system : $(x, t) \in \Omega(t) \times [0, t_{\text{max}}]$

\[
\begin{align*}
\frac{\partial T}{\partial t}(x, t) + v(x, t) \cdot \nabla T(x, t) & = \alpha_i \nabla^2 T(x, t), \quad x \in \Omega(t) \quad (2.1) \\
\frac{\partial c}{\partial t}(x, t) + v(x, t) \cdot \nabla c(x, t) & = D_i \nabla^2 c(x, t), \quad x \in \Omega(t) \quad (2.2) \\
\frac{\partial v}{\partial t}(x, t) + v(x, t) \cdot \nabla v(x, t) & = \nu \nabla^2 v(x, t) - g \beta_T [T(x, t) - T_0] \\
& \quad + g \beta_c [c(x, t) - c_0] - \frac{1}{\rho} \nabla p(x, t), \quad x \in \Omega(t) \quad (2.3) \\
\nabla \cdot v(x, t) & = 0, \quad x \in \Omega(t) \quad (2.4) \\
T(x, 0) & = T_i, \quad c(x, 0) = c_i, \quad v(x, 0) = 0, \quad x \in \Omega(0) \quad (2.5) \\
v(x, t) & = 0, \quad x \in \Gamma(t) \quad (2.6) \\
\frac{\partial^*}{\partial n}(x, t) & = q^h(x, t), \quad x \in \Gamma^h(t) \quad (2.7) \\
T(x, t) & = T_s(x), \quad x \in \Gamma_s(t) \quad (2.8) \\
D_i \frac{\partial c}{\partial n}(x, t) & = 0, \quad x \in \Gamma_o \quad (2.9)
\end{align*}
\]

2.1 Nondimensionalization

Following standard practice, we introduce the following dimensionless variables that identify the significance of competing physical mechanisms:

\[
\begin{align*}
\hat{x} & = \frac{x}{L} \quad \hat{t} = \frac{t}{L^2} \quad \hat{T} = \frac{T - T_0}{\Delta T} \\
\hat{c} & = \frac{c}{c_0} \quad \hat{v} = \nu \frac{v}{L} \quad \hat{p} = \frac{p L^2}{\rho_0}
\end{align*}
\]

The reference time scale is defined by $\tau = L^2 / \alpha$, where $L$ is a characteristic length. The reference temperature difference is chosen based on the problem parameters for each example problem examined and will be considered later. The governing nondimensional parameters are the thermal Rayleigh number $Ra_T(= \frac{\beta_T k A T L^3}{\nu \alpha})$, the solutal Rayleigh number $Ra_c(= \frac{\beta_c k A T L^3}{\nu \alpha})$, the Lewis number $Le(= \frac{\alpha}{\beta})$ and the Prandtl number $Pr_T(= \frac{\nu}{\alpha})$. The nondimensional form of the governing equations thus obtained, is summarized in Box 2.2. For simplicity, the hats indicating dimensionless quantities are not shown in Box 2.2, and from now on, unless it is stated otherwise, all quantities will be assumed to be in a nondimensional form.
Box 2.2
Governing Equations for the double diffusive system (Nondimensional): \((x, t) \in \Omega(t) \times [0, t_{max}]\)

\[
\begin{align*}
\frac{\partial T}{\partial t}(x, t) + \mathbf{v}(x, t) \cdot \nabla T(x, t) &= \nabla^2 T(x, t), \quad x \in \Omega(t) \\
\frac{\partial c}{\partial t}(x, t) + \mathbf{v}(x, t) \cdot \nabla c(x, t) &= Le^{-1} \nabla^2 c(x, t), \quad x \in \Omega(t) \\
\frac{\partial \mathbf{v}}{\partial t}(x, t) + \mathbf{v}(x, t) \cdot \nabla \mathbf{v}(x, t) &= Pr \nabla^2 \mathbf{v}(x, t) - Pr Ra_T T(x, t) \mathbf{e}_g \\
&\quad + Pr Ra_c c(x, t) \mathbf{e}_g - \nabla p(x, t), \quad x \in \Omega(t) \\
\nabla \cdot \mathbf{v}(x, t) &= 0, \quad x \in \Omega(t) \\
T(x, 0) &= T_i, \quad c(x, 0) = c_i, \quad \mathbf{v}(x, 0) = 0, \quad x \in \Gamma(t) \\
\mathbf{v}(x, t) &= 0, \quad x \in \Omega(0), \\
\frac{\partial T}{\partial n}(x, t) &= q^b_i(x, t), \quad x \in \Gamma^b_i(t) \\
T(x, t) &= T_s(x), \quad x \in \Gamma_s^s(t) \\
\frac{\partial c}{\partial n}(x, t) &= 0, \quad x \in \Gamma_c
\end{align*}
\] (2.10) (2.11) (2.12) (2.13) (2.14) (2.15) (2.16) (2.17) (2.18)

3 Weighted residual formulation

The weighted residual variational formulation can be stated as: Find a velocity vector field \(\mathbf{v}\), a temperature field \(T\) and a concentration field \(c\) such that \(\mathbf{v}, T, c\) and their first partial derivatives with respect to the spatial coordinates and time are square integrable over \(\Omega\) and satisfy the prescribed boundary conditions on \(\Gamma\), and such that:

\[
\begin{align*}
\int_\Omega \left\{ w_i \delta_{ij} \frac{\partial v_j}{\partial t} + w_i \delta_{ij} v_k \frac{\partial v_j}{\partial x_k} + \lambda \frac{\partial w_i}{\partial x_i} \frac{\partial v_j}{\partial x_j} + Pr \frac{\partial w_i}{\partial x_i} \frac{\partial v_j}{\partial x_j} + Ra_T Pr w_i \delta_{ij} \left( T - \frac{Ra_c}{Ra_T} c \right) \cos \theta_j \right\} d\Omega \\
+ \sum_{\epsilon = 1}^{n_c} A_{\epsilon} \int_{\Omega_{\epsilon}} \left\{ p_i \delta_{ij} \left( \frac{\partial v_i}{\partial t} + v_r \frac{\partial v_i}{\partial x_k} + Ra_T Pr \left( T - \frac{Ra_c}{Ra_T} c \right) \cos \theta_j \right) \right\} d\Omega = \int_{\Gamma_{ht}} w_i \delta_{ij} h_j d\Gamma \\
\int_\Omega \left\{ W \frac{\partial T}{\partial t} + W v_k \frac{\partial T}{\partial x_k} + \frac{\partial W}{\partial x_k} \frac{\partial T}{\partial x_k} \right\} d\Omega \\
+ \sum_{\epsilon = 1}^{n_c} A_{\epsilon} \int_{\Omega_{\epsilon}} \left\{ Q \left( \frac{\partial T}{\partial t} + v_k \frac{\partial T}{\partial x_k} \right) \right\} d\Omega = \int_{\Gamma_{ht}} W q d\Gamma \\
\int_\Omega \left\{ \frac{\partial c}{\partial t} + U v_k \delta_{ij} \frac{\partial c}{\partial x_k} + \frac{\partial U}{\partial x_k} \delta_{ij} \frac{\partial c}{\partial x_k} \right\} d\Omega \\
+ \sum_{\epsilon = 1}^{n_c} A_{\epsilon} \int_{\Omega_{\epsilon}} \left\{ R \left( \frac{\partial c}{\partial t} + v_k \delta_{ij} \frac{\partial c}{\partial x_k} \right) \right\} d\Omega = \int_{\Gamma_{hc}} U j d\Gamma
\end{align*}
\] (3.1) (3.2) (3.3)
at each time $t$, where $w_i = 0$ on $\Gamma_{gw}$ (boundary with prescribed $w$), $W = 0$ on $\Gamma_{gT}$ and $U = 0$ on $\Gamma_{gc}$ (boundary with prescribed $T$, $c$). Also, $\Gamma_{hT}$ is the part of the boundary where a heat flux $\dot{q}_h$ is applied, $\Gamma_{hc}$ is the part of the boundary where a solute diffusive flux $\dot{j}$ is applied, while $\Gamma_{hu}$ is the part of the boundary with applied tractions. The function $w$, $W$, $U$ and their first partial derivatives with respect to spatial coordinates are square integrable over $\Omega$, while the functions $p$, $Q$, $R$ are discontinuous across element boundaries and continuous within each element. More detail on the weighting functions is available elsewhere (see [7, 8]).

4 The semi-discrete approximation and time stepping procedure

4.1 Semi-discrete Equations

The semidiscrete equations resulting from the momentum, energy and solute conservation equations are similar in nature to the equations given in [4] and their derivation will not be repeated here and only their final form is listed. The incompressibility condition is implemented using the penalty approach (here $\lambda$ denotes the penalty number). Further details and examples demonstrating the accuracy of such formulations can be found in references [7][10]

**Momentum equation**

$$Ma + C'v + C''v + N(v)v + E(T) = 0$$

(4.1)

where $v$ and $a$ are the vectors containing the velocity and acceleration degrees-of-freedom, respectively, and $T$ is the vector containing the temperature degrees-of-freedom. The matrices $M$, $C'$, $C''$, $N(v)$ and the vector $E(T)$ are given as follows:

$$M_{\hat{p}\hat{q}} = \sum_{\varepsilon = 1}^{n_{el}} \overline{A} \left\{ \int_{\Omega_{\varepsilon}} (N_\alpha^e + p_\alpha^e) N_{\beta}^e \delta_{ij} d\Omega \right\}$$

(4.2)

$$C''_{\hat{p}\hat{q}} = \lambda \sum_{\varepsilon = 1}^{n_{el}} \overline{A} \left\{ \int_{\Omega_{\varepsilon}} N_\alpha^e N_{\beta}^e \delta_{ij} d\Omega \right\}$$

(4.3)

$$C'_p = Pr \sum_{\varepsilon = 1}^{n_{el}} \overline{A} \left\{ \int_{\Omega_{\varepsilon}} N_{\alpha,ik} N_{\beta,k}^e \delta_{ij} d\Omega \right\}$$

(4.4)

$$N(v)_{\hat{p}\hat{q}} = \sum_{\varepsilon = 1}^{n_{el}} \overline{A} \left\{ \int_{\Omega_{\varepsilon}} (N_\alpha^e + p_\alpha^e) v_k^e N_{\beta,k}^e \delta_{ij} d\Omega \right\}$$

(4.5)

$$E(T)_{\hat{p}} = Pr Ra_T \sum_{\varepsilon = 1}^{n_{el}} \overline{A} \left\{ \int_{\Omega_{\varepsilon}} (N_\alpha^e + p_\alpha^e)(T - \frac{Ra_c}{Ra_T} \cos \theta_i) \delta_{ij} d\Omega \right\}$$

(4.6)

where $\hat{p}$ refers to the global velocity degree of freedom in the final system of equations that corresponds to the $i^{th}$ velocity degree of freedom of the elemental node $\alpha$ ($i = 1, \ldots, n_{sd}$) in the
element. Similarly, one can define \( \dot{q} \). Here, \( p^e_\alpha \) denotes the discontinuous contributions to the Petrov-Galerkin weights \([6]\). Also, note that \( \theta_j \) is the angle between the \( j^{th} \) coordinate axis and the gravity vector.

### Energy equation

\[
M\dot{T} + KT = -F(v)T
\]  
(4.7)

where the dot denotes time derivatives and \( T \) are the temperature degrees of freedom. The same notation is used as in the discrete momentum equation in order to emphasize the similar structure of the two subproblems. The matrices \( M, K, F(v) \) are here defined as follows:

\[
M_{pq} = \sum_{e=1}^{n_e} \int_{\Omega_e} \left( N^e_\alpha + p^e_\alpha \right) N^e_\beta d\Omega,
\]
(4.8)

\[
K_{pq} = \sum_{e=1}^{n_e} \int_{\Omega_e} N^e_\alpha N^e_\beta v_k d\Omega,
\]
(4.9)

\[
F(v)_{pq} = \sum_{e=1}^{n_e} \int_{\Omega_e} \left( N^e_\alpha + p^e_\alpha \right) v_k N^e_\beta d\Omega,
\]
(4.10)

where \( p \) and \( q \) refer to the global degrees of freedom in equation (4.7).

### Solute conservation equation

\[
M\dot{c} + Kc = -F(v)c
\]  
(4.11)

where the dot denotes time derivatives and \( c \) are the concentration degrees of freedom. The matrices \( M, K, F(v) \) are here given by:

\[
M_{pq} = \sum_{e=1}^{n_e} \int_{\Omega_e} \left( N^e_\alpha + p^e_\alpha \right) N^e_\beta d\Omega,
\]
(4.12)

\[
K_{pq} = \sum_{e=1}^{n_e} \int_{\Omega_e} N^e_\alpha N^e_\beta v_k d\Omega,
\]
(4.13)

\[
F(v)_{pq} = \sum_{e=1}^{n_e} \int_{\Omega_e} \left( N^e_\alpha + p^e_\alpha \right) v_k N^e_\beta d\Omega,
\]
(4.14)

where \( p \) and \( q \) refer to the global degrees of freedom in the concentration linear system of equations.
**Remark 1:** For the flows involving double diffusive convection, the body force term $E(T)p$ (equation (4.6)) was implemented using $N^p$ as weight (i.e. the discontinuous functions $p$ were not used here to produce a consistent Petrov-Galerkin formulation). This change was first discussed by Heinrich [4]. Even though no theoretical justification for the need of such an inconsistent Petrov-Galerkin formulation is as yet known, with an extensive number of tests, we found that such a formulation produces accurate results, whereas the consistent Petrov-Galerkin formulation (i.e. using all terms in equation (4.6)) was shown to exhibit unwanted oscillations and for many cases not to converge.

4.2 Time stepping algorithm

The discretization of the governing equations is complete and the discrete equations listed in the previous section can be solved in a manner similar to that described in [4]. The time stepping is done in the following steps:

1. All dependent variables are known at time level $t_n$
2. Calculate the temperature values at time $t_{n+1}$ from the energy discrete equation, using the velocity values at time level $t_n$.
3. Calculate the salinity concentration values at time $t_{n+1}$ from the solute concentration discrete equation, using the velocity values at time level $t_n$.
4. Calculate the new velocity field at time $t_{n+1}$ from the discretized Navier-Stokes equations using the updated values of the concentration and temperature fields and the velocity field at the previous time level $t_n$.
5. Time step is completed, go to step 1.

**Remark 2:** Note that due to the weak coupling of the various sub-problems making up the complex double-diffusive problem, decoupling of the various simulators has been performed. That is, no overall loop is required to enforce coupling of the $T$, $c$, and $v$ fields. This fact has been confirmed through numerical tests.

5 Numerical implementation

An object oriented programming environment provides an excellent framework for the simulation of flows involving more than one scalar contaminants (e.g. temperature and solute concentration). These problems involve several coupled sub-problems that can be analyzed and numerically formulated independent of each other. For example, the melt convection problem can be numerically implemented and solved to calculate the flow field assuming that the temperature and solute fields are known.

To allow ourselves to concentrate only on the implementation of physical mechanisms in the form of classes, we have selected to base our computations on the widely used **diffpack** library of classes for analysis of systems governed by partial differential equations [11]. Most of the present developments are general and other object oriented frameworks can also be considered. Details on **diffpack** is not provided here and is available elsewhere [11]. We will briefly review the development of the individual classes and the coupling details.
5.1 Class development for each transport process

Three main solvers are constructed as individual classes, one for each physical process. The implementation features of the flow solver (class ‘NavierStokes’) and heat solver (class ‘NlHeat’) are already available elsewhere [7, 8] and they are not repeated here. The solute concentration solver (class ‘Concentration’) is implemented in exactly the same way as that used to model the heat solver, as both fall under the category of scalar convection-diffusion problems. The main members of the various subproblems necessary to demonstrate the coupling are listed in Boxes 5.1, 5.2 and 5.3. Note that only the members essential to understand coupling details of the three independent solvers are shown in Boxes 5.1, 5.2 and 5.3.

Box 5.1

Members of the class ‘NavierStokes’ necessary to demonstrate the coupling.

class NavierStokes: public ... {
friend class DDConvection
protected:
    ...
    Handle(FieldFE) T;  // Known temperature field
    Handle(FieldFE) C;  // Known concentration field
    Handle(FieldsFE) u;  // The unknown fluid velocity
    ...
    Handle(TimePrm) tip;  // Time discretization parameters
    void setIC();
    void solveAtThisTimeLevel();
    ...
public:
    ...
    void define(MenuSystem& menu, int level=MAIN);
    void scan(MenuSystem& menu);
    void adm(MenuSystem& menu);
    ...
};
Box 5.2
Members of the class ‘NlHeat’ necessary to demonstrate the coupling.

class NlHeat: public ...
{  
friend class DDConvection
protected:
  ..
  Handle(FieldFE) T;  // The unknown temperature
  Handle(FieldsFE) u;  // Known fluid velocity
  ..
  Handle(TimePrm) tip;  // Time discretization parameters
  void setIC();
  void solveAtThisTimeLevel();
  ..
public:
  ..
  void define(MenuSystem& menu, int level=MAIN);
  void scan(MenuSystem& menu);
  void adm(MenuSystem& menu);
  ..
};

Box 5.3
Members of the class ‘Concentration’ necessary to demonstrate the coupling.

class Concentration: public ...
{  
friend class DDConvection
protected:
  ..
  Handle(FieldFE) C;  // The unknown concentration
  Handle(FieldsFE) u;  // Known fluid velocity
  ..
  Handle(TimePrm) tip;  // Time discretization parameters
  void setIC();
  void solveAtThisTimeLevel();
  ..
public:
  ..
  void define(MenuSystem& menu, int level=MAIN);
  void scan(MenuSystem& menu);
  void adm(MenuSystem& menu);
  ..
};
As can be seen, the class ‘NavierStokes’ needs the members ‘C’ and ‘T’ as input for solving for the velocity and acceleration at the next time step. Similarly, the classes ‘NlHeat’ and ‘Concentration’ need the velocity ‘u’ as input. As shown in an earlier report [8], the coupling is achieved very easily without disturbing the already tested individual sub-problem (e.g. the ‘NavierStokes’ code). The coupling is achieved by using pointers or what is known as ‘Handles’ in *diffpack* (These are basically pointers but with additional capabilities). Using pointers, each field in a parent class (e.g. ‘u’ in ‘NavierStokes’) is made to point to the corresponding fields in other classes. This is done in the main simulator ‘DDConvection’ and hence there is no need to touch the other classes. However, to let the main simulator class ‘DDConvection’ access private members of the sub-classes, the main class must be made a ‘Friend’ of the other classes. Boxes 5.4 and 5.5 explain exactly how the coupling is achieved.

The reader is encouraged to study the source code for further details on implementation.

*Box 5.4*

The main members of class ‘DDConvection’ to show the coupling - I.

```c++
class DDConvection {
protected:
    Handle(NlHeat) heat; // Pointer to an object of the heat simulator
    Handle(NavierStokes) flow; // Pointer to an object of the flow simulator
    Handle(Concentration) conc; // Pointer to an object of the concentration (mass diffusion) simulator
    Handle(TimePrm) tip; // Time discretization parameters

public:
    void define(MenuSystem& menu, int level=MAIN);
    void scan(MenuSystem& menu);
    void adm(MenuSystem & menu);
    void timeLoop();
    
};
```
6 Numerical examples

6.1 Example 1

This problem was obtained from [4]. The physical problem under consideration corresponds to a rectangular enclosure 1.27 cm wide by 6.35 cm high with physical properties tabulated in Table 6.1. At time \( t = 0 \), the right hand side walls' temperature is raised to the nondimensional value
$T=1.0$ while maintaining the left hand side at a constant temperature of $T=0$, the initial uniform temperature distribution. The geometry and boundary conditions are shown in Fig. 2.

The computational mesh used in the calculations consists of 896 nodes and 825 elements. The interior part of the grid consists of a uniform mesh of 9x49 elements. By interior, we mean the whole domain except the layer of thickness 0.1 on the boundary. The grid in exterior layer consists of three elements with approximate spacing 0.025, 0.025 and 0.05 from the outer boundary. The grid had to be chosen in such a manner so as to sufficiently capture the flow in the boundary layer. The grid is shown in Fig. 1.

<table>
<thead>
<tr>
<th>Table 6.1</th>
<th>Physical Properties of the system analyzed in Example 1.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Symbol</td>
<td>Value</td>
</tr>
<tr>
<td>Diffusivity</td>
<td>$D_l$</td>
</tr>
<tr>
<td>Reference temperature difference</td>
<td>$\Delta T$</td>
</tr>
<tr>
<td>Thermal diffusivity</td>
<td>$\alpha_L$</td>
</tr>
<tr>
<td>Kinematic viscosity</td>
<td>$\nu$</td>
</tr>
<tr>
<td>Thermal expansion coefficient</td>
<td>$\beta_T$</td>
</tr>
<tr>
<td>Melting temperature</td>
<td>$T_m$</td>
</tr>
<tr>
<td>Reference temperature</td>
<td>$T_o$</td>
</tr>
<tr>
<td>Initial salinity gradient</td>
<td>$\phi_o$</td>
</tr>
<tr>
<td>Reference length scale</td>
<td>$L$</td>
</tr>
<tr>
<td>Reference time scale</td>
<td>$\tau$</td>
</tr>
</tbody>
</table>
\[
\frac{\delta T}{\delta z} = 0 \quad \frac{\delta C}{\delta z} = 0 \quad u = 0
\]

\[
\frac{\delta C}{\delta z} = 0 \quad T = T_i \quad \bar{u} = 0
\]

\[
\frac{\delta T}{\delta z} = 0 \quad \frac{\delta C}{\delta z} = 0 \quad u = 0
\]

\[
\frac{\delta C}{\delta z} = 0 \quad T = T_s \quad \bar{u} = 0
\]

Figure 2: Geometry and boundary conditions for Example 1.

<table>
<thead>
<tr>
<th>Nondimensional parameters governing the system analysed in Example 1.</th>
<th>Symbol</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aspect ratio</td>
<td>A</td>
<td>5.0</td>
</tr>
<tr>
<td>Prandtl number</td>
<td>Pr</td>
<td>6.21</td>
</tr>
<tr>
<td>Solutal Rayleigh number</td>
<td>(Ra_c)</td>
<td>(2.07 \times 10^6)</td>
</tr>
<tr>
<td>Thermal Rayleigh number</td>
<td>(Ra_T)</td>
<td>(9.47 \times 10^5)</td>
</tr>
<tr>
<td>Lewis number</td>
<td>Le</td>
<td>105.0</td>
</tr>
</tbody>
</table>

The nondimensional parameters corresponding to the physical problem are listed in Table 6.2. The problem was run for a total nondimensional time of 0.3. A nondimensional time step of \(10^{-5}\) was used through the calculation.

The results of the numerical calculations are shown in Figs. 4 - 7. The results show the following steps, first the fluid rises rapidly along the heated wall, soon after this, the flow starts to turn from the wall and rolls begin to form which grow slowly and eventually a layered system, is formed as shown in Fig. 3. This is in confirmation with the numerical results of [4] and the experimental and numerical results of [1]. The results are in good comparison with the numerical results of Heinrich ([4]). There are however slight differences in the number of cells reported in [4] and that reported here. This is basically due to visualization differences (some of the reported cells have very weak strength). The average size of cells in the present example is around 0.55 cm which agrees well with the results of [4] and the scale analysis given in [12].
Figure 3: The formation of cells in a stratified fluid. Initially the fluid rises along the heated wall ($t_1$). The flow spontaneously turns from the wall ($t_2$) and forms equal sized rolls ($t_3$) which grow laterally ($t_4$) ultimately forming a layered structure ($t_5$).

6.2 Example 2

This problem was also obtained from reference [4]. The physical problem in the first example was slightly altered to illustrate the process of merging of cells as time scales become larger. In order to achieve this within reasonable computational time, the parameters in the first example were modified (exactly as in [4]) so that the Lewis number is made 10 and the ratio $R_c/R_T = 1$, keeping the same temperature Rayleigh number as in Example 1. The diffusive scale for the solute concentration is approximately ten times faster than in the first example, hence we expect the cell merging process to be a lot quicker. The same FE grid was used for the calculations but the time step was raised to $2 \times 10^{-5}$. The total simulated nondimensional time was 0.3 corresponding to 834 secs in actual time. The results of the numerical calculations are shown in the stream-function field development plots in Figs. 8 and 9. As can be seen from Fig. 8, at very early times two cells are developed at the bottom. These cells emerge very quickly and further cells develop in sequence after that. At later times (see Fig. 9), the cells merge with each other and form larger cells. There is however, one difference between the present results and those reported in [4], in that even though there is very good agreement in the flow behavior and flow patterns, the number of cells in the present case is exactly one less than that reported in [4]. The reason for this is not clear. However, due to the excellent agreement of the flow patterns and the merging process this discrepancy has not been investigated further. Furthermore, the temperature and salinity contours along the centerline plotted in Fig. 10 also agree very well with those reported in [4].
Figure 4: Salinity profiles, stream function contours and isotherms for Example 1 at $t = 2.8$ secs, Stream function contours $1.0, 0.5, -1, -3, \ldots$; isotherms in steps of 0.1 and isoconcentration lines in steps of 0.5.

Figure 5: Salinity profiles, stream function contours and isotherms for Example 1 at $t = 236.3$ secs, Stream function contours $1.0, 0.5, -1, -3, \ldots$; isotherms in steps of 0.1 and isoconcentration lines in steps of 0.5.
Figure 6: Salinity profiles, stream function contours and isotherms for Example 1 at \( t = 614.4 \) secs, Stream function contours 1.0, 0.5, -1, -3, ...; isotherms in steps of 0.1 and isoconcentration lines in steps of 0.5.

Figure 7: Salinity profiles, stream function contours and isotherms for Example 1 at \( t = 834 \) secs, Stream function contours 1.0, 0.5, -1, -3, ...; isotherms in steps of 0.1 and isoconcentration lines in steps of 0.5.
Figure 8: Stream function contours for Example 2, at times (a) $t = 55.6$ sec, (b) $t = 111.2$ sec, (c) $t = 166.9$ sec and (d) $278.0$ sec, contours 1-9 : -3, -5, ..; A : -1; B : 0.5; C : 1.0.

Figure 9: Stream function contours for Example 2, at times (a) $t = 55.6$ sec, (b) $t = 111.2$ sec, (c) $t = 166.9$ sec and (d) $278.0$ sec, contours 1-9 : -3, -5, ..; A : -1; B : 0.5; C : 1.0.
6.3 Example 3

This problem was obtained from [5]. It is almost similar to the first and second examples except that instead of a vertical enclosure, we look at double diffusive convection in an inclined cavity having an aspect ratio of 5 to 1 and inclined at a 30° angle to the vertical. The problem geometry and boundary conditions are shown in Fig. 11. The temperature gradient is applied symmetrically on the two side walls and in such a manner that the heat flows from the bottom to the top side wall. The physical parameters are the same as those used in Example 1.

A finite element mesh similar to the one used in the first example was used in this simulation consisting of 896 nodes and 825 elements as shown in Fig. 12. The calculations were simulated up to a nondimensional time of 0.2 and were stopped due to excess computational cost. The time step was $10^{-5}$ as in the first example. The results of the numerical simulation are shown in Figs. 13-15. As can be seen clearly, the concentration profile does not change. As can be seen from the stream function plots, the flow forms two cells at the top and bottom of the enclosure very early. As time progresses, the flow develops more and more cells, until it finally forms a layered profile. The calculations, however, are stopped when all cells just begin to appear as can be seen in the final stream function plot. The temperature profile, as that in Example 1, shows a wavy pattern characteristic of double diffusive flow. The results compare well with the results of [5].
1. Side: Cold Wall
   \[ T = T_c \]
   \[ \frac{dT}{dn} = 0 \]
   \[ \vec{u} = 0 \]

2. Side: Hot wall
   \[ T = T_H \]
   \[ \frac{dT}{dn} = 0 \]
   \[ \vec{u} = 0 \]

3. Side: Top and Bottom
   \[ \frac{dC}{dn} = 0 \]
   \[ \vec{u} = 0 \]

Figure 11: Geometry and Boundary conditions for Example 3.

Figure 12: Finite Element mesh used in Example 3.
Figure 13: Isoconcentration, streamline, and isotherm contours at $t = 55.6$ secs; plotted at intervals of 0.5, 1, and $\approx 0.1$, respectively. The maximum stream function $\approx -6.0$.

Figure 14: Isoconcentration, streamline and isotherm contours at $t = 166.8$ secs; plotted at intervals of 0.5, 1, and $\approx 0.1$, respectively. The maximum stream function $\approx -9.0$. 
Figure 15: Isoconcentration, streamline and isotherm contours at t = 556 secs; plotted at intervals of 0.5, 1. and ≈ 0.1, respectively. The maximum stream function ≈ -17.0.

References


