Simulation of glass flow in an oven

Citation for published version (APA):

Document status and date:
Published: 01/01/1999

Document Version:
Publisher’s PDF, also known as Version of Record (includes final page, issue and volume numbers)

Please check the document version of this publication:
• A submitted manuscript is the version of the article upon submission and before peer-review. There can be important differences between the submitted version and the official published version of record. People interested in the research are advised to contact the author for the final version of the publication, or visit the DOI to the publisher’s website.
• The final author version and the galley proof are versions of the publication after peer review.
• The final published version features the final layout of the paper including the volume, issue and page numbers.

Link to publication

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

• Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
• You may not further distribute the material or use it for any profit-making activity or commercial gain
• You may freely distribute the URL identifying the publication in the public portal.

If the publication is distributed under the terms of Article 25fa of the Dutch Copyright Act, indicated by the “Taverne” license above, please follow below link for the End User Agreement:
www.tue.nl/taverne

Take down policy
If you believe that this document breaches copyright please contact us at:
openaccess@tue.nl
providing details and we will investigate your claim.
Simulation of glass flow in an oven

Seva Nefedov
Scientific Computing Group
Technische Universiteit Eindhoven
Den Dolech 2
5600 MB Eindhoven
nefedov@win.tue.nl

1 Introduction

![Simulation of a glass tank](image)

Figure 1: Simulation of a glass tank, computed with TNO’s industrial package GTM

Glass-based products can be found in virtually all areas of human activity. There exists a lot of different sorts of glass, but whatever their final form, they are all produced in an oven, the so-called glass tank.

The actual production of glass in an oven is a rather complicated process. Raw batch materials (main component: siliciumdioxide and soda) enter the oven through the inlets, and inside they are heated by burners. The heating process and the structure of the oven are designed in such a way, that the melt remains for a period of time in the tank. Several techniques are used in order to obtain a uniform density, such as electrochemical boosting and bubbling.

This paper introduces a method to simulate the flow and heat transfer in the tank. The mathematical model is formulated in the second section. Section 3 describes Patankar’s algorithm SIMPLER [5], the numerical method on which our approach is based. Due to the local nature of some processes, it turns out that the grid in the method needs to be refined on certain small subregions.
This can be done without a global refining by means of a method called \textit{Local Defect Correction} (LDC) \cite{1}. We shall briefly describe this in section 4. The last sections contain results of numerical experiments demonstrating efficiency of LDC.

2 Glass tank model

By the \textit{Glass tank model} (GTM) we mean a set of equations complemented by corresponding boundary conditions, which describe different processes inside a glass tank. In this paper we focus our attention on the flow field and heat transfer.

The flow in the tank is modelled by the well-known Navier-Stokes equations (see e.g.\cite{3}).

\begin{equation}
\begin{cases}
(\rho(x)u, \nabla u) = F - \nabla p + \nabla \cdot (\mu(x) \nabla u), \\
\nabla \cdot (\rho(x)u) = 0.
\end{cases}
\end{equation}

The coefficients $\rho(x)$ and $\mu(x)$ are \textit{density} and \textit{viscosity} of the glass respectively and $F$ is the \textit{gravitational force}.

For a steady low-velocity flow with negligible dissipation we can derive the energy equation in the following form

$$\nabla \cdot (C_p \rho(x) u T) = \nabla \cdot (k(x) \nabla T),$$

where $C_p$ is the \textit{heat capacity}, $T$ is the \textit{temperature} and $k$ is the \textit{heat conductivity}.

![Sketch of a glass tank](image)

Figure 2: Sketch of a glass tank

In order to make the model complete we need to know the values of the physical parameters. The most important factor here is the exponential behaviour of the viscosity (see fig 3). Due to this behaviour, the momentum equations might become convection dominant, which induces certain computational problems. Two different solutions to mend this are discussed in [2] and [4].
3 Patankar’s algorithm SIMPLER

Let us restrict ourselves to two dimensional computations and consider the flow field equations. After applying the finite volume method (see e.g. [2]) on a 2D staggered grid (see fig. 4), we obtain the following system

\[
\begin{pmatrix}
M_1(u) & 0 & B^e_x \\
0 & M_2(u) & B^e_y \\
B^w & B^v & 0
\end{pmatrix}
\begin{pmatrix}
u \\
v
p
\end{pmatrix}
= \begin{pmatrix}
F_1 \\
F_2 \\
0
\end{pmatrix}.
\] (3)

Here \( M_1(u) \) and \( M_2(u) \) are non-linear discrete momentum operators. Matrices \( B^e_x, B^e_y \) correspond to partial derivatives in the continuity equation, while \( B^w, B^v \) correspond to the gradient operator. We denote the diagonal parts of \( M_1 \) and \( M_2 \) as

\[ D_1 = \text{diag}(M_1), \quad D_2 = \text{diag}(M_2). \]

The following computational scheme represents the algorithm SIMPLER:

0. \( u^{n-1}, v^{n-1}, T^{n-1} \) known
1. Calculate

\[
\ddot{u} = D_1^{-1}[F_1 - (M_1(u^{n-1}) - D_1)u^{n-1}], \quad \ddot{v} = D_2^{-1}[F_2 - (M_2(u^{n-1}) - D_2)v^{n-1}].
\]
2. Assemble
   \[ P := (B^x D^{-1}_1 B^x_1 + B^y D^{-1}_2 B^y_2). \]

3. Solve
   \[ Pp^* = B^x \hat{u} + B^y \hat{v}. \]

4. Solve
   \[ M_1(u^{n-1})u^* = F_1 - B^x_1 p, \quad M_2(u^{n-1})v^* = F_2 - B^y_2 p. \]

5. Solve
   \[ Pp' = B^x u^* + B^y v^* \]

6. Correct velocity
   \[ u^n = u^* - D^{-1}_1 B^x_1 p', \quad v^n = v^* - D^{-1}_2 B^y_2 p'. \]

If the method diverges, the underrelaxed formulae should be used:
\[ u^n = \alpha u^{n-1} + (1 - \alpha) (u^* - D^{-1}_1 B^x_1 p'), \quad v^n = \alpha v^{n-1} + (1 - \alpha) (v^* - D^{-1}_2 B^y_2 p') \]

7. Substitute computed velocity field into the energy equation, thus compute new temperature.

8. Check convergence; return to step 1, if not accurate enough.

A special feature of SIMPLER is that the newly computed velocity satisfies the continuity equation exactly, in contrast to the momentum equations. Although the pressure correction \( p' \) is computed, the pressure is not updated. Even if we were to update the pressure, the momentum equations would not be satisfied exactly. Residuals of the momentum equations can be easily computed:
\[ r^n_u = F_1 - M_1(u^{n-1})u^n + B^x_1 p = \]
\[ \left\{ \begin{array}{l}
M_1(u^{n-1}) D^{-1}_1 B^x_1 p', \quad \text{if the pressure is not corrected,}

(M(u^{n-1}) D^{-1}_1 I) B^x_1 p', \quad \text{if the pressure is corrected.}
\end{array} \right. \]

It should be mentioned, that one iteration of SIMPLER requires solution of the four linear systems. We use Bi-CGSTAB as a linear solver [6].

4 Local defect correction

Local Defect Correction (LDC) is a technique which uses (local) uniform grids, thereby employing their simple data structure in solving the global problem. In order to describe the method we consider a model boundary value problem

\[
Lu = f \text{ in } \Omega = (0,1) \times (0,1), \\
u = \varphi \text{ on } \partial\Omega.
\]

Here \( L \) is an arbitrary operator. The composite grid \( \Omega^{h,h} \) is composed of a global coarse grid and a local fine grid. The global coarse grid \( \Omega^H \) is a uniform
grid with grid size $H$, the local fine grid $\Omega_{loc}^h$ has grid size $h$. There is one more important grid (see fig. 5), namely the local coarse grid $\Omega_{loc}^H$ which is defined by

$$\Omega_{loc}^H = \Omega^H \cap \Omega_{loc}^h$$

Using notation introduced above we now can describe LDC procedure:

1. Solve

$$L^H u^H_n = f^H_n \text{ on } \Omega^H, \quad f^H_0 = f^H.$$  

2. Solve

$$L^h_{loc} u^h_{loc,n} = f^h_{loc,n}(u^H_n) \text{ on } \Omega^h_{loc}.$$  

(Boundary conditions for this problem are obtained from $u^H_n$ by means of interpolation.)

3. Construct the composite approximation $u^{H,h}_n(x)$

$$u^{H,h}_n(x) = \begin{cases} u^h_{loc,n}(x) & x \in \Omega_{loc}^h, \\ u^H_n(x) & x \in \Omega^H \setminus \Omega_{loc}^h. \end{cases}$$

4. Construct the global coarse grid function $w^H$

$$w^H(x) = \begin{cases} u^h_{loc,n}(x) & x \in \Omega_{loc}^h, \\ u^H_n(x) & x \in \Omega^H \setminus \Omega_{loc}^h. \end{cases}$$

5. Compute the defect $d^H$

$$d^H = L^H w^H - f^H.$$  

6. Update the right-hand part of the global problem

$$f^H_{n+1}(x) = \begin{cases} f^H_n(x) + d^H(x) & x \in \Omega_{loc}^H, \\ f^H_n(x) & x \in \Omega^H \setminus \Omega_{loc}^h. \end{cases}$$
7. Check convergence; return to step 1, if not accurate enough.
To illustrate performance of the LDC applied to the flow equations we shall consider a model problem next.

5 Uniform channel flow with stirrer

Figure 6: Stirred flow - magnified velocity field (left), absolute value of velocity (right)

Figure 7: Solution obtained with LDC - magnified velocity field (left), absolute value of velocity (right)

Let us consider the domain \( \Omega := (0, 2) \times (0, 1) \) and on this a flow with a parabolic profile, perturbed by a stirrer. The stirrer is modelled as a small solid square with prescribed tangential velocity. The stirring effect will be completely missed unless the mesh size is smaller than the size of the stirrer. Hence we need to refine locally. We place the local fine mesh \( \Omega_{loc}^h \) (21 \times 21 cells) in the center of the domain; the refinement factor, \( H/h \) is 3. In order to obtain error estimates we compute a numerical solution of the problem on the global fine mesh (see fig. 6). We denote it as \( \hat{u} \) and \( \hat{v} \). The following expressions are used to compare results

\[
e_u = \frac{\|\hat{u} - u^H,h\|_{\infty}}{\|\hat{u}\|_{\infty}}, \quad e_v = \frac{\|\hat{v} - v^H,h\|_{\infty}}{\|\hat{v}\|_{\infty}};
\]

here \( u^H,h \) and \( v^H,h \) denote solution on the composite grid (see fig. 7).
<table>
<thead>
<tr>
<th>#LDC iter.</th>
<th>$e_u$</th>
<th>$e_v$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>$2.63 \cdot 10^{-4}$</td>
<td>$1.42 \cdot 10^{-1}$</td>
</tr>
<tr>
<td>1</td>
<td>$1.17 \cdot 10^{-1}$</td>
<td>$5.20 \cdot 10^{-2}$</td>
</tr>
<tr>
<td>2</td>
<td>$7.16 \cdot 10^{-2}$</td>
<td>$3.19 \cdot 10^{-2}$</td>
</tr>
<tr>
<td>3</td>
<td>$4.03 \cdot 10^{-2}$</td>
<td>$1.76 \cdot 10^{-2}$</td>
</tr>
</tbody>
</table>

Table 1: LDC iteration results

The solution computed on fine mesh (fig.6) and the LDC solution computed on the composite grid (fig.7) are almost identical. It means, that LDC provides us with an accurate solution, while significantly reducing computational costs. A distinctive feature of LDC is the creation of the defect. If the procedure is stopped after the zeroth iteration, that is the update of the global problem doesn’t occur (step 6), then no information would be transferred onto the global mesh and the solution would not be accurate enough (see fig. 8).

6 Simplified glass tank

Figure 9: Glass tank - vector field (left), temperature field (right)
Let us consider a simplified variant of a glass tank (see fig. 9). A cold stream is entering the domain; it being warmed up from the top by a heat flux and finally hot fluid is flowing out of the tank. Close to the top part of the inlet the temperature gradient causes a small twist to appear. We choose the region $[0.1, 0.5] \times [0.7, 0.95]$ for the local refinement, the refinement factor 3 and the global coarse mesh $20 \times 20$ cells. In the region of the refinement viscosity decays rapidly on a very short area from approximately 2500 to 25, but, still, LDC gives us a solution (see fig.10).

From the results shown above we can conclude that LDC is a powerful tool for resolving local phenomena. It is effective both with respect to memory use and CPU time, and we do not have to work with a data structure arising from the combined grid.

References


