Local defect correction for glass flow simulation

EINDHOVEN UNIVERSITY OF TECHNOLOGY
Department of Mathematics and Computing Science

RANA 00-22
December 2000

Local defect correction for glass flow simulation

by

V. Nefedov and R.M.M. Mattheij

Reports on Applied and Numerical Analysis
Department of Mathematics and Computing Science
Eindhoven University of Technology
P.O. Box 513
5600 MB Eindhoven, The Netherlands
ISSN: 0926-4507

Local Defect Correction for Glass Flow Simulation

V.Nefedov* and R.M.M.Mattheij*

23 December 2000

Abstract

This paper describes a model to perform simulation of the glass flow in an oven. The Navier-Stokes equations are discretised by a finite difference method on a collocated grid and solved by a pressure correction method. Efficiency of the solution method can be enhanced if the consistency condition for the boundary values is satisfied. A special feature is that we employ refinement by uniform grids and solve the global system by so called local defect correction (LDC). A special correction of the boundary conditions for LDC, preserving the consistency condition, is introduced. The performance of the method is illustrated by practical examples.

1 Introduction

Products made completely or partially from glass, such as TV and computer screens, windows, tableware, play an important role in our daily life. Before one is able to produce such products, the glass itself needs to be made. This process takes place in a so-called glass oven or melting tank. The manufacturing starts with putting raw material, the main components being soda and sand, into one side of the oven, which is heated by gas burners from above. During the melting all components mix with each other. This heating and mixing takes about 20 hours, after which the glass flows via feeding channels to the production lines. There are therefore several processes involved, like melting, flow and heat transfer, as well as various chemical reactions. Numerical simulations are an essential tool for understanding these processes. In this paper we restrict our attention to the molten glass flow simulation. The glass oven can be considered as a three-dimensional domain domain composed of blocks. The molten glass is then essentially a three-dimensional flow. For a proper simulation of the latter, we would need about 100 grid points in each spatial dimension. This leads naturally to the choice of tensor grids and as compact and as simple data structures as possible.

*Scientific Computing group, Department of Mathematics and Computing Science, Eindhoven University of Technology, PO BOX 513, 5600MB Eindhoven, The Netherlands, nefedov@win.tue.nl, mattheij@win.tue.nl

Since the flow has some areas of high activity a uniform grid does not give enough resolution there or is too expensive. Such areas occur for instance where stirring is needed to make the glass more uniform or air bubbles are being injected to insure a proper mixing process. In order to properly resolve these phenomena a local refinement procedure is needed. The method we are using was originally suggested in [9], see also [1]. It is called Local Defect Correction (LDC) and is a an iterative procedure which accurately combines solutions computed on a global coarse and local fine meshes. The paper is built up as follows: the model of the glass flow is formulated in the next section, discretisation techniques and the solution method for the flow equation are described in section 3, section 4 contains the detailed description of LDC. The fifth section deals with the estimates for the perturbation parameters needed for construction of the local boundary conditions. In the last section we illustrate an application of LDC to the glass flow and compare LDC and non-uniform refinement using tensor grids.

2 The glass oven model

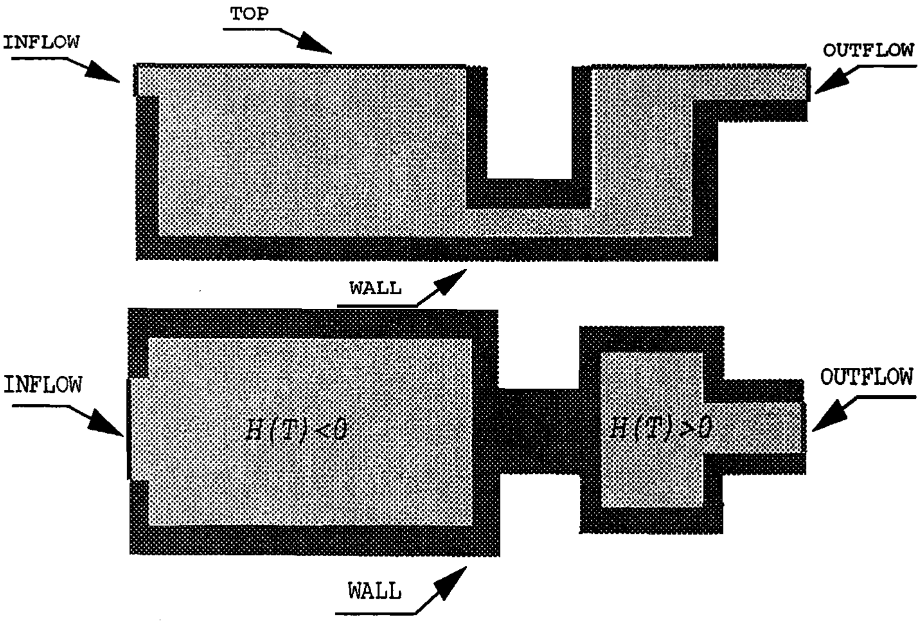

Figure 1: Sketch of the glass tank (horizontal and vertical cross sections)

The geometry of industrial glass ovens may vary, still each one contains some typical part In figure 1 we have sketched a simple oven. It consists of two tanks connected via a channel. The left tank is an area where the heating and melting occur. The smaller, right tank is connected via the feeders with the production lines. Since our primal interest is the flow itself, we ignore the melting, assuming that the glass melt of a certain temperature flows into the oven via the inflow area. We assume the glass to be a Newtonian fluid, cf

[16]. Let $\tilde{\mathbf{u}} = (\tilde{u}, \tilde{v}, \tilde{w})^T$ denote the velocity, $\tilde{p}$ the pressure, $\tilde{\mu}$ the viscosity of glass, $\tilde{\rho}$ the density of glass and let $\tilde{\mathbf{F}} = (0, 0, -\tilde{\rho}g)^T$ be the body force per unit volume. The flow equations can be written as follows

$$(\tilde{\rho}(\tilde{\mathbf{x}})\tilde{\mathbf{u}}, \tilde{\nabla}\tilde{\mathbf{u}}) = \tilde{\mathbf{F}} - \tilde{\nabla}\tilde{p} + \tilde{\nabla} \cdot (\tilde{\mu}(\tilde{\mathbf{x}})\tilde{\nabla}\tilde{\mathbf{u}}), \quad \text{(conservation of momentum)}$$

and

$$\tilde{\nabla} \cdot (\tilde{\rho}(\tilde{\mathbf{x}})\tilde{\mathbf{u}}) = 0. \quad \text{(conservation of mass)}$$

Tildes indicate that variables and parameters are dimensionfull. The equations are written in a time-independent form, because for larger time scales the glass flow exhibits steady-state behaviour. The viscosity $\tilde{\mu}$ and density $\tilde{\rho}$ are temperature dependent. In order to obtain the temperature values we need to add the energy equation to our model. However, it would make the computations significantly more complex and not really add to our major objective i.e. to demonstrate our refinement technique. Instead, we use a realistic temperature field which varies in space and approximates the actual temperature in the oven. The flow equations are still too general since they might as well represent any viscous fluid in a oven. In order to restrict the model to the particular case of the glass flow we are interested in, we need to specify the parameters of the equation, namely the viscosity and the density. Both entities are functions of the temperature. The viscosity decays exponentially as the temperature increases. The expression for the viscosity is known as the Vogel Fucher Tamman law, [16]

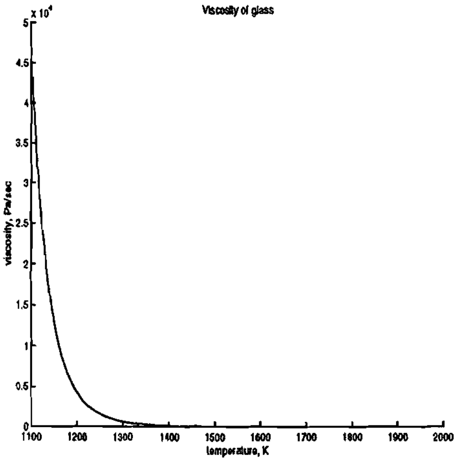

Figure 2: Viscosity of tv-glass

$$\tilde{\mu} = \tilde{\mu}(\tilde{T}) = ae^{b/(\tilde{T}-c)}. \tag{1}$$

Since the viscosity is a function of the temperature and the temperature varies in space, the viscosity is also a spatially dependent function. We will further refer to it as $\tilde{\mu}(\tilde{\mathbf{x}})$. The coefficients a, b, c in (1) are specific for different glass types (tv-glass, window-glass etc).

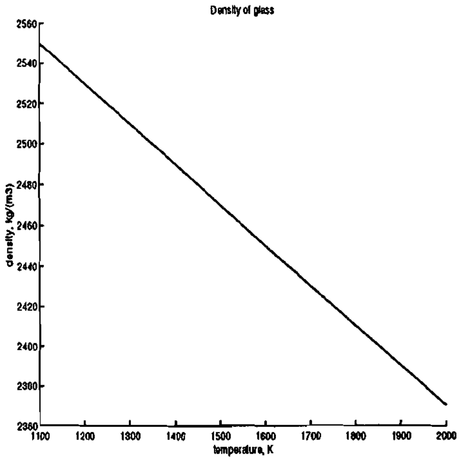

Figure 3: Density of tv-glass

The most significant factor in a glass flow computation is the density. It may be modeled as a linear function of the temperature, more precisely

$$\tilde{\rho} = \tilde{\rho}(\tilde{T}) = \alpha(1 - \beta(\tilde{T} - \gamma)), \tag{2}$$

The changes in density, however small, are the driving force behind the glass flow. In order to understand the relevant phenomena in our model we introduce dimensionless quantities as follows:

$$\mathbf{x} := \frac{1}{X}\tilde{\mathbf{x}}, \;\; \mathbf{u} := \frac{1}{U}\tilde{\mathbf{u}}, \;\; \rho := \frac{1}{\rho_0}\tilde{\rho}, \;\; \mu := \frac{1}{\mu_0}\tilde{\mu},$$

where X, U, ρ_0 and μ_0 are the characteristic length, velocity, density and viscosity respectively For a typical glass flow these values are

$$X = 2[m], \;\; U = 0.01[\frac{m}{s}], \;\; \rho_0 = 2200[\frac{kg}{m^3}], \;\; \mu_0 = 40[\frac{kg}{m \cdot s}].$$

The gradient and divergence operators written in old variables – $\tilde{\nabla}$, $\tilde{\nabla}\cdot$ and new variables ∇ and $\nabla\cdot$ are related via

$$\nabla = X\tilde{\nabla}, \;\; \nabla\cdot = X\tilde{\nabla}\cdot.$$

Substituting this information into the flow equations we have

$$\frac{\rho_0 u^2}{X}(\rho\mathbf{u}, \nabla\mathbf{u}) = \tilde{\mathbf{F}} - \frac{1}{X}\nabla\tilde{p} + \frac{\mu_0 U}{X^2}\nabla(\mu\nabla\cdot\mathbf{u}),$$

$$\frac{\rho_0 U}{X}\nabla\cdot(\rho\mathbf{u}) = 0.$$

Dividing the momentum equation on both sides by $X/(\rho_0 U^2)$, the continuity equation by $\rho_0 U/X$ and defining

$$\mathbf{F} := \frac{X}{\rho_0 U^2}\tilde{\mathbf{F}}, \;\; p := \frac{1}{\rho_0 U^2}\tilde{p}, \;\; Re := \frac{\rho_0 X U}{\mu_0},$$

(where *Re* is a Reynolds number) we obtain the dimensionless Navier-Stokes equations

$$\begin{cases} (\rho(\mathbf{x})\mathbf{u}, \nabla\mathbf{u}) = \mathbf{F} - \nabla p + \frac{1}{Re}\nabla \cdot (\mu(\mathbf{x})\nabla\mathbf{u}), \\ \nabla \cdot (\rho(\mathbf{x})\mathbf{u}) = 0. \end{cases} \tag{3}$$

The Reynolds number is not small enough to neglect the convection term, thus we need to consider full Navier-Stokes equations. The boundary of any oven is composed of several parts, such as inflow, outflow, the top layer, and glass-wall border, see Figure 1. On each of these boundary conditions for the velocity must be defined. We shall prescribe fixed inflow and outflow velocities. At the border between wall and glass we assume no-slip. The top layer is modelled as a symmetry plane, that is the normal component of the velocity is zero; elsewhere we prescribe homogeneous Neumann boundary conditions.

$$\begin{gathered} u|_{\Gamma_{inflow}} = u_0, \;\; v|_{\Gamma_{inflow}} = 0, \;\; w|_{inflow} = 0, \\ u|_{\Gamma_{outflow}} = u_1, \;\; v|_{\Gamma_{outflow}} = 0, \;\; w|_{outflow} = 0, \\ u|_{\Gamma_{wall,glass}} = u_1, \;\; v|_{\Gamma_{wall,glass}} = 0, \;\; w|_{wall,glass} = 0, \\ \tfrac{\partial u}{\partial z}|_{\Gamma_{top}} = 0, \;\; \tfrac{\partial v}{\partial z}|_{\Gamma_{top}} = 0, \;\; w|_{top} = 0. \end{gathered}$$

3 The solution-method for the governing equations

For simplicity and efficiency reasons we opt for the finite difference method (FDM) among the various discretisation techniques. The natural choice for the (FDM) applied to incompressible viscous flow problems is a staggered grid, where all dependent variables are computed in different locations, see e.g. [13]. The problem discretised on a staggered grid is always well-posed. However, the grid is far more complex than a standard tensor or collocated grid, where all variables are computed at the same mesh nodes. Since we like to carry out a simulation in a three dimensional space, where the domain is composed of blocks and the number of grid points can easily be as large as 10^6, we would prefer to use a collocated grid. On such a domain we introduce a tensor grid which is a product of three one-dimensional grids

$$\begin{gathered} x_i = iH_x, \;\; H_x = a/N_x, \;\; i = 1...N_x, \\ y_j = jH_y, \;\; H_y = b/N_y, \;\; j = 1...N_y, \\ z_k = kH_z, \;\; H_z = c/N_z, \;\; k = 1...N_z. \end{gathered}$$

Here a, b and c are the length, width and height of the oven and H_x, H_y, H_z are the grid sizes. We would refer to the maximum grid size as H, i.e. $H = max(H_x, H_y, H_z)$. The grid domain Ω_H is a set of grid points belonging to the physical domain. We do not describe the details of the discretisation of the diffusion-convection and the body force here, see e.g. [7]. Instead we concentrate on the gradient and the divergence operators. The discrete gradient ∇_h is defined by

$$(\nabla_h p)_{i,j,k} := (\frac{p_{i+1,j,k} - p_{i-1,j,k}}{2H_x}, \frac{p_{i,j+1,k} - p_{i,j-1,k}}{2H_y}, \frac{p_{i,j,k+1} - p_{i,j,k+1}}{2H_z})^T. \tag{4}$$

Similarly the discrete divergence ∇_h is defined as

$$(\nabla_h \cdot \mathbf{u})_{i,j,k} := \frac{u_{i+1,j,k} - u_{i-1,j,k}}{2H_x} + \frac{v_{i,j+1,k} - v_{i,j-1,k}}{2H_y} + \frac{w_{i,j,k+1} - w_{i,j,k+1}}{2H_z}. \tag{5}$$

After collecting these relations we obtain a discrete Navier-Stokes system of the following form

$$\begin{cases} \mathbf{M_H}(\mathbf{u})\mathbf{u} + \nabla_H p = \mathbf{F}, \\ \nabla_H \cdot \rho\mathbf{u} = 0. \end{cases} \tag{6}$$

The discrete operator $\mathbf{M}_H$ includes diffusion and convection components. Its non-linearity is indicated by explicit dependence on $\mathbf{u}$.

The solution method we use for solving (6) is of pressure correction type cf.[10]. We first rewrite the momentum equation

$$\mathbf{0} = \mathbf{F} - \mathbf{M}_H(\mathbf{u})\mathbf{u} - \nabla_H p. \tag{7}$$

Rather than solving (7) directly we consider a transient equation

$$\frac{\partial \rho u}{\partial t} = \mathbf{F} - \mathbf{M}_H(\mathbf{u})\mathbf{u} - \nabla_H p. \tag{8}$$

Using Euler-backwords for the time dependent term and Picards iteration to handle the non-linearity (a part of the nonlinear term is computed using the information from the previous iteration), we can formulate the predictor step

$$\frac{1}{\Delta t}(\rho\mathbf{u}^* - \rho\mathbf{u}^n) = \mathbf{F} - \mathbf{M}_H(\mathbf{u}^n)\mathbf{u}^* - \nabla_H p^n. \tag{9}$$

Assuming $\mathbf{u}^n$ and p^n to be known we can solve (9) for $\mathbf{u}^*$. Since $\mathbf{u}^*$ might not satisfy the continuity equation, we need to correct it. We use an expression like (9) but now in explicit form

$$\frac{1}{\Delta t}(\rho\mathbf{u}^{n+1} - \rho\mathbf{u}^n) = \mathbf{F} - \mathbf{M}_H(\mathbf{u}^*)\mathbf{u}^* - \nabla_H p^{n+1}. \tag{10}$$

This equation involves two unknown entities: the new velocity field $\mathbf{u}^{n+1}$ and the new pressure p^{n+1}. To resolve this problem we apply the discrete divergence (5) to both parts of (10). Requiring $\nabla_H \cdot \rho\mathbf{u}^{n+1} = 0$ we obtain an equation from which we can compute the new pressure p^{n+1}

$$-\nabla_H \cdot \nabla_H p^{n+1} = -\nabla_H \cdot (\mathbf{F} - \mathbf{M}_H(\mathbf{u}^*)\mathbf{u}^* + \frac{1}{\Delta t}\rho\mathbf{u}^n). \tag{11}$$

To construct (11) we used the continuity equation. It turns out, that for some applications, other constraints can used, see e.g. [15]. The boundary conditions for (11) follow from the fact that we do not correct the velocity at the boundary

$$\mathbf{0} = \frac{1}{\Delta t}(\rho \mathbf{u}^{n+1} - \rho \mathbf{u}^{n}) = \mathbf{F} - \mathbf{M}_H(\mathbf{u}^n)\mathbf{u}^* - \nabla_H p^{n+1} \quad \text{at the boundary.}$$

Hence

$$\nabla_H p^{n+1} = \mathbf{F} - \mathbf{M}_H(\mathbf{u}^*)\mathbf{u}^* \quad \text{at the boundary.}$$

After having found p^{n+1} we substitute it into (10) and compute the new velocity $\mathbf{u}^{n+1}$. The algorithm thus far can now be summarized as the follows:

```
Algorithm 1 Pressure Correction
0. Let $\mathbf{u}^n$, $p^n$, $\Delta t$ be given.
1. Find the predicted value of the velocity $\mathbf{u}^*$ from
       $\frac{1}{\Delta t}(\rho \mathbf{u}^* - \rho \mathbf{u}^n) = \mathbf{F} - \mathbf{M}(\mathbf{u}^n)\mathbf{u}^* - \nabla_h p^n.$
2 Find $p^{n+1}$ from the pressure equation
       $-\nabla_h \cdot \nabla_h p^{n+1} = -\nabla_h \cdot (\mathbf{F} - \mathbf{M}(\mathbf{u}^*)\mathbf{u}^* + \frac{1}{\Delta t}\rho \mathbf{u}^n).$
3 Compute the corrected velocity field $\mathbf{u}^{n+1}$ from
       $\frac{1}{\Delta t}(\rho \mathbf{u}^{n+1} - \rho \mathbf{u}^n) = \mathbf{F} - \mathbf{M}(\mathbf{u}^*)\mathbf{u}^* - \nabla_h p^{n+1}.$
4. Check convergence, returtn to step 1 if residuals are not small enough.
```

The major problem arises when solving the system for the pressure. If the pressure matrix $\nabla_H \cdot \nabla_H$ is constructed as a combination of the discrete divergence and the discrete gradient as introduced above thus acting on a pressure as follows

$$(-\nabla_H \cdot \nabla_H p)_{i,j,k} = \frac{2p_{i,j,k} - p_{i-2,j,k} - p_{i+2,j,k}}{4H_x^2} + \frac{2p_{i,j,k} - p_{i,j-2,k} - p_{i,j+2,k}}{4H_y^2} + \frac{2p_{i,j,k} - p_{i,j,k-2} - p_{i,j,k+2}}{4H_z^2}. \tag{12}$$

The pressure computed from (12) might exhibit high oscillations. This unphysical behaviour is caused by the fact that the pressure matrix $-\nabla_h \cdot \nabla_h$ has eight zero eigenvalues. To avoid the problem we replace the pressure matrix by a standard 5-point discretisation of the Laplace operator denoted as ∇_h^2,

$$(-\nabla_H^2 p)_{i,j,k} := \frac{2p_{i,j,k} - p_{i-1,j,k} - p_{i+1,j,k}}{H_x^2} + \frac{2p_{i,j,k} - p_{i,j-1,k} - p_{i,j+1,k}}{H_y^2} + \frac{2p_{i,j,k} - p_{i,j,k-1} - p_{i,j,k+1}}{H_z^2}. \tag{13}$$

The two matrices $-\nabla_H \cdot \nabla_H$ and $-\nabla_H^2$ both are approximate the same continuous Laplace operator. It can be shown see e.g. [7] that the difference between them can be estimated as

$$||-\nabla_H \cdot \nabla_H p + \nabla_H^2 p||_\infty \leq CH^2,$$

where C depends on fourth order derivatives of p, e.i. for smooth p the error introduced by changing the pressure matrix would be small. For the overall performance of the method it is essential to be able to solve the pressure system efficiently; therefore we would like to analyse $-\nabla_h^2$ a bit better. It has the following properties:

1. The diagonal coefficients are positive, off-diagonal – non-negative.
2. The row-sums are zero for each row.
3. It is irreducible.
4. It can be symmetrised by diagonal scaling.

Thus for any vector p the sum of all components on the left-hand side of the system applied to p is equal to zero

$$\sum_{i,j,k} (-\nabla_H^2 p)_{i,j,k} = 0.$$

The same should hold for the right-hand side

$$\sum_{i,j,k} [-\nabla_h \cdot (\mathbf{F} - \mathbf{M}_H(\mathbf{u}^*)\mathbf{u}^* + \frac{1}{\Delta t}\rho \mathbf{u}^n]_{i,j,k} = 0. \tag{14}$$

All components on the left-hand side of (14) corresponding to the internal points of the grid cancel each other and we are left with an expression which involves only the boundary nodes. We recall that at the boundary the following relation holds

$$\nabla_H p^{n+1} = \mathbf{F} - \mathbf{M}_H(\mathbf{u}^*)\mathbf{u}^*$$

Taking this into account (14) can be reformulated as

$$\sum_{j,k=1}^{N_y,N_z} H_y H_z (u_{1,j,k} - u_{N_x,j,k}) + \sum_{i,k=1}^{N_x,N_z} H_x H_z (v_{i,1,k} - v_{i,N_y,k}) + \sum_{i,j=1}^{N_x,N_y} H_x H_y (w_{i,j,1} - w_{i,j,N_z}) = 0. \tag{15}$$

Condition (15) is thus a *consistency condition* for the pressure system. Since (15) depends only on the boundary values of the velocity, it can be checked prior to solving the Navier-Stokes equations. If (15) is satisfied the pressure system has infinitely many solutions which differ by a constant vector. If (15) is not satisfied the situation can be repaired in various ways. Most common is to simply fix the pressure at one point; it makes the pressure matrix non-singular, i.e. the pressure system will always have a solution. However, this way of avoiding the problem has a drawback; fixing the pressure at one point often significantly increases the number of iterations needed to solve the pressure system. An alternative approach will be analysed in the next section.

4 Local defect correction

Since computing on a uniform fine mesh in a three-dimensional space is too expensive we rather like to refine only locally. One way to do this is employing Local Defect Correction (LDC) cf. [5], [6]. LDC enables us to combine results computed on two or more grids. One grid is always the coarse grid covering the whole domain while other (fine) grids cover regions with higher activity. Besides the global coarse grid Ω_H we shall use a local fine grid Ω_h^l which is a tensor product of three grids inside a block $[x_{i_0}, x_{i_1}] \times [y_{j_0}, y_{j_1}] \times [z_{k_0}, z_{k_1}]$,

$$\Omega_h^l = \{x_i^l, y_j^l, z_k^l\}, \tag{16}$$

$$x_i^l = x_{i_0} + ih_x, \;\; h_x = H_x/\sigma_x, \;\; 0 \le i \le N_x^l := 1 + (x_{i_1} - x_{i_0})/h_x, \tag{17}$$

$$y_j^l = y_{j_0} + jh_y, \;\; h_y = H_y/\sigma_y, \;\; 0 \le j \le N_y^l := 1 + (y_{j_1} - y_{j_0})/h_y, \tag{18}$$

$$z_k^l = z_{k_0} + kh_z, \;\; h_z = H_z/\sigma_z, \;\; 0 \le k \le N_z^l := 1 + (z_{k_1} - z_{k_0})/h_z. \tag{19}$$

A superscript l indicates that that the variable is defined locally, the subscripts H and h indicate the coarse and fine meshes respectively. The σ_x, σ_y and σ_z are called the *refinement factors*. They are assumed to be positive integers larger than one, i.e.

$$\sigma_x \in \mathbb{N}, \;\; \sigma_x > 1, \;\; \sigma_y \in \mathbb{N}, \;\; \sigma_y > 1, \;\; \sigma_z \in \mathbb{N}, \;\; \sigma_z > 1.$$

The grids Ω_H and Ω_h^l form together the composite grid $\Omega_{H,h}$

$$\Omega_{H.h} = \Omega_H \cup \Omega_h^l.$$

We also need a grid which is a set of vertexes belonging to both global and local grids, denoted as Ω_H^l, i.e.

$$\Omega_H^l = \{x_i, y_j, z_k\}, \;\; i_0 \le i \le i_1, \;\; j_0 \le j \le j_1, \;\; k_0 \le k \le k_1.$$

First, we solve the discrete Navier-Stokes equations on Ω_H

$$\begin{aligned} \mathbf{M}_H(\mathbf{u}_H)\mathbf{u}_H + \nabla_H p_H &= \mathbf{F}_H, \\ \nabla_H \cdot \rho \mathbf{u}_H &= 0. \end{aligned} \tag{20}$$

Next, by using linear interpolation for the computed velocity field we find Dirichlet boundary conditions for the problem on the local fine mesh. In order to be able to solve the local problem we need to check whether the boundary conditions for the local problem satisfy the compatibility condition (15). In general (15) is not satisfied. It means that

$$\sum_{j,k=1}^{N_y^l,N_z^l} h_y h_z (u_{1,j,k}^l - u_{N_x^l,j,k}^l) + \sum_{i,k=1}^{N_x^l,N_z^l} h_x h_z (v_{i,1,k}^l - v_{i,N_y^l,k}^l) + \sum_{i,j=1}^{N_x^l,N_y^l} h_x h_y (w_{i,j,1}^l - w_{i,j,N_z^l}^l) = \delta. \tag{21}$$

To mend this this we can adjust the boundary values at the grid points which belong to the local grid only. First we need a notation for the indices of local grid points at the faces of the Ω_h^l

$$\begin{aligned}
I^l &:= \{(j,k)|(x_i^l, y_j^l, z_k^l) \in \Omega_h^l \setminus \Omega_H^l\}, \\
J^l &:= \{(i,k)|(x_i^l, y_j^l, z_k^l) \in \Omega_h^l \setminus \Omega_H^l\}, \\
J^l &:= \{(i,k)|(x_i^l, y_j^l, z_k^l) \in \Omega_h^l \setminus \Omega_H^l\}.
\end{aligned}$$

When the consistency condition is not satisfied, this means that inflow and outflow do not balance each other. The residual δ can be zero, positive or negative. If zero, there is no problem. If $\delta > 0$ the outflow is larger than inflow and if $\delta < 0$ the inflow dominates. Let us assume, for simplicity, that $\delta > 0$, i.e. the outflow is larger than inflow. The idea is to make the inflow larger and the outflow smaller by correcting the velocity at the local points only. Thus we define the new velocity value $\tilde{u}^l$ as

$$\tilde{u}_{1,j,k}^l = (1 - sign(u_{1,j,k}^l)\alpha)u_{1,j,k}^l, \quad \tilde{u}_{N_x^l,j,k}^l = (1 + sign(u_{N_x^l,j,k}^l)\alpha)u_{N_x^l,j,k}^l, \quad (i,j) \in K^l. \tag{22}$$

By analogy we correct the other components of the velocity field. Substituting the updated velocity into (21) we have

$$\begin{aligned}
&\sum_{j,k=1}^{N_y^l,N_z^l} h_y h_z(\tilde{u}_{1,j,k} - \tilde{u}_{N_x^l,j,k}) + \sum_{i,k=1}^{N_x^l,N_z^l} h_x h_z(\tilde{v}_{i,1,k} - \tilde{v}_{i,N_y^l,k}) + \sum_{i,j=1}^{N_x^l,N_z^l} h_x h_y(\tilde{w}_{i,j,1} - \tilde{w}_{i,j,N_z^l}) = \\
&\qquad \delta - \alpha \sum_{(j,k)\in I^l} h_y h_z(|u_{1,j,k}^l| + |u_{N_x^l,j,k}^l|) + \alpha \sum_{(i,k)\in J^l} h_x h_z(|v_{i,1,k}^l| + |v_{i,N_y^l,k}^l|) + \\
&\qquad\qquad \alpha \sum_{(i,j)\in K^l} h_x h_y(|w_{i,j,1}^l| + |w_{i,j,N_z^l}^l|).
\end{aligned}$$

Therefore, the natural choice for α is given by

$$\begin{aligned}
\alpha^{-1} = \delta^{-1}[&\sum_{(j,k)\in X_l} h_y h_z(|u_{1,j,k}^l| + |u_{N_x^l,j,k}^l|) + \sum_{(i,k)\in Y_l} h_x h_z(|v_{i,1,k}^l| + |v_{i,N_y^l,k}^l|) + \\
&\sum_{(i,j)\in Z_l} h_x h_y(|w_{i,j,1}^l| + |w_{i,j,N_z^l}^l|)].
\end{aligned} \tag{23}$$

Using the newly corrected velocity at the boundary of Ω_h^l we can formulate the local problem

$$\begin{aligned}
\mathbf{M}_h(\mathbf{u}_h^l)\mathbf{u}_h^l + \nabla_h p_h^l &= \mathbf{F}_h^l, \\
\nabla_h \cdot \rho \mathbf{u}_h^l &= 0.
\end{aligned} \tag{24}$$

After having solved the local problem we combine the two results into the solution at the composite mesh

$$\mathbf{u}_{H,h} = \begin{cases} \mathbf{u}_h^l, & \text{if the node is in } \Omega_h^l, \\ \mathbf{u}_H, & \text{otherwise.} \end{cases} \tag{25}$$

If we would combine the solution on the local fine grid with that on the complementary coarse grid we will not obtain a sufficiently globally accurate solution in general. We need to provide a certain exchange of information between solutions on coarse and fine grids to correct the approximation on the composite grid. First we construct a (grid) function $\mathbf{g}_H$ that is defined on the coarse mesh points

$$\mathbf{g}_\mathbf{H} = \begin{cases} \mathbf{u}_\mathbf{h}, & \text{if the node is in } \Omega^l_H, \\ \mathbf{u}_\mathbf{H}, & \text{otherwise .} \end{cases}$$

The function $\mathbf{g}_H$ can be considered to be a projection of $\mathbf{u}_{H,h}$ onto the coarse mesh Ω_H. Next we compute the defect $\mathbf{d}$

$$\mathbf{d} := \mathbf{F}_\mathbf{H} - \mathrm{M}_H(\mathbf{g}_H)\mathbf{g}_H + \nabla_H p_H. \tag{26}$$

Since $\mathbf{g}_H$ coincides with $\mathbf{u}_H$ outside Ω^l_H, the defect is only nonzero in Ω^l_H. The right-hand side of (20) is updated with the defect and we have

$$\begin{aligned} &\mathrm{M}_H(\mathbf{u}^1_H)\mathbf{u}^1_H + \nabla_H p^1_H = \mathbf{F}_H - \mathbf{d}, \\ &\nabla_H \cdot \rho\mathbf{u}^1_H = 0. \end{aligned} \tag{27}$$

The boundary conditions for the problem on the local fine grid are obtained by interpolating $\mathbf{u}^1_H$. Since they will generally not satisfy the compatibility condition (15) we correct them in the above-described fashion. After solving the local problem

$$\begin{aligned} &\mathrm{M}_h(\mathbf{u}^{l,1}_h)\mathbf{u}^{l,1}_h + \nabla_h p^{l,1}_h = \mathbf{F}^l_h, \\ &\nabla_h \cdot \rho\mathbf{u}^{l,1}_h = 0, \end{aligned} \tag{28}$$

we combine $\mathbf{u}^1_H$ and $\mathbf{u}^{l,1}_h$ into a new composite solution

$$\mathbf{u}^1_{H,h} = \begin{cases} \mathbf{u}^{l,1}_h, & \text{if the node is in } \Omega^l_h, \\ \mathbf{u}^1_H, & \text{otherwise.} \end{cases} \tag{29}$$

The solution method for the glass flow can be now formulated as follows:
Algorithm 2 *Local Defect Correction*

0. $\mathbf{d} := 0$, i:=0

1. Solve by algorithm 1

$$\begin{aligned} &\mathrm{M}_H(\mathbf{u}^i_H)\mathbf{u}^i_H + \nabla_H p^i_H = \mathbf{F}_H - \mathbf{d}, \\ &\nabla_H \cdot \rho\mathbf{u}^i_H = 0. \end{aligned}$$

2. Create boundary conditions for the local problem by interpolation.

3. Correct the boundary conditions if (15) is not satisfied.

4. Solve by algorithm 1

$$\begin{aligned} &\mathrm{M}_h(\mathbf{u}^{l,i}_h)\mathbf{u}^{l,i}_h + \nabla_h p^{l,i}_h = \mathbf{F}^l_h, \\ &\nabla_h \cdot \rho\mathbf{u}^{l,i}_h = 0. \end{aligned}$$

5. Construct the composite approximation

$$\mathbf{u}^i_{H,h} = \begin{cases} \mathbf{u}^{l,i}_h, & \text{if the node is in } \Omega^l_h, \\ \mathbf{u}^i_H, & \text{otherwise.} \end{cases}$$

6. i:=i+1; return to step 1 if residual is not small enough.

5 Estimate for the perturbation parameter

Since we perturbed the boundary conditions for the local problem by $(1 - \alpha)$ we would like to obtain an estimate for α. This is borne out in a property, the derivation of which is deferred till the end of this section. We start with a definition of a discrete integral norm.

Definition 1 *For $f(x,y) \in C([a,b] \times [c,d])$, $h_x = (b-a)/(N-1)$, $h_y = (d-c)/(M-1)$, $x_i = a + ih_x$, $y_b = c + jh_y$, the discrete integral norm of f is defined by*

$$||f||_{L^1_h} := \sum_{i,j=1}^{N,M} h_x h_y |f(x_i, y_j)|$$

Next we introduce a concept which is useful to describe a function requirements needed below.

Definition 2 *A function $f \in C([a,b] \times [c,d])$ is called infinitely oscillating around zero if there exists a point $\mathbf{x} \in [a,b] \times [c,d]$ and a direction $\mathbf{r}$, such that for any $\epsilon > 0$, there exist $0 < t_1 < t_2 < \epsilon$ such that*

$$f(\mathbf{x} + t_1\mathbf{r}) f(\mathbf{x} + t_2\mathbf{r}) < 0.$$

The following lemma gives an estimate of the difference between the integral and discrete integral norms

Lemma 1 *For $f(x,y) \in C^2([a,b] \times [c,d])$, such that f is not infinitely oscillating and $Z := \{(x,y) | f(x,y) = 0\}$ the roots of f form a one-dimensional manifold or a finite set of points, the following holds*

$$| ||f||_{L_1} - ||f||_{L^h_1} | \leq C_1 h^2$$

Proof We note that for a function which is only positive or only negative on $[a,b] \times [c,d]$, the discrete integral norm coincides with midpoint integration. We can encounter problems only at those points where f changes sign. Therefore we need handle this cases separately. We split the complete domain $[a,b] \times [c,d]$ into two parts

$$[a,b] \times [c,d] = ([a,b] \times [c,d] \setminus V_Z) \cup V_Z, \tag{30}$$

where V_Z is a h-neigbourhood of Z

$$\begin{aligned} V_Z := \cup_{(i,j)} V_{i,j}, \ V_{i,j} &:= [x_i - h_x/2, x_i + h_x/2] \times [y_j - h_y/2, y_j + h_y/2], \\ &(i,j) \in \{(i,j) | \exists (x,y) \in Z \cap V_{i,j}\}. \end{aligned} \tag{31}$$

Using the fact that $\int_a^c = \int_a^b + \int_b^c$ and the splitting (30) we have

$$\begin{aligned} &| ||f||_{L_1([a,b]\times[c,d])} - ||f||_{L^h([a,b]\times[c,d])} | \leq \\ &| ||f||_{L_1([a,b]\times[c,d]\setminus V_Z)} - ||f||_{L^h_1([a,b]\times[c,d]\setminus V_Z)} | + | ||f||_{L_1(V_Z)} - ||f||_{L^h_1(V_Z)} |. \end{aligned} \tag{32}$$

The first term of the right-hand side of (32) can be interpreted as an error of the midpoint integration rule

$$|||f||_{L_1([a,b]\times[c,d]\setminus V_Z)} - ||f||_{L_1^h([a,b]\times[c,d]\setminus V_Z)}| \leq D_1 h^2$$

Let now $(\xi, \eta) \in Z$ be the point in h-neighbourhood of (x_i, y_j), that is

$$(\xi, \eta) \in V_{i,j} \subset V_Z.$$

We need to estimate

$$\int_{V_{i,j}} |f(x_i, y_j| - |f(x,y)| dx dy.$$

Using a Taylor expansion around (ξ, η) we have

$$|\int_{V_{i,j}} |f(x_i, y_j| - |f(x,y)| dx dy| \leq D_2 h^3,$$

where D_2 incorporates first derivatives of f. The integral over V_Z can be represented as a sum of integrals over $V_{i,j}$ constituting V_Z

$$|||f||_{L_1(V_Z)} - ||f||_{L_1^h(V_Z)}| = \sum_{i,j} | \int_{V_{i,j}} |f(x_i, y_j| - |f(x,y)| dx dy|. \tag{33}$$

Expression (33) can be estimated by the number of $V_{i,j}$ times the maximum over all integrals. Noting that that the number of $V_{i,j}$ is at most $\int_Z 1 d\Gamma / h$, we have

$$|||f||_{L_1(V_Z)} - ||f||_{L_1^h(V_Z)}| \leq D_2 \int_Z 1 d\Gamma h^2$$

Finally,

$$|||f||_{L_1([a,b]\times[c,d])} - ||f||_{L_1^h([a,b]\times[c,d])}| \leq C_1 h^2,$$

with $C_1 = D_1 + D_2 \oint_Z 1 d\Gamma$. ■

Before we arrive at our desired property we first give another lemma for which we need a notation for the indices of points at the faces of Ω_H^l, viz.

$$I^g := \{(j,k) | (x_i^l, y_j^l, z_k^l) \in \Omega_H^l,$$
$$J^g := \{(i,k) | (x_i^l, y_j^l, z_k^l) \in \Omega_H^l,$$
$$K^g := \{(i,j) | (x_i^l, y_j^l, z_k^l) \in \Omega_H^l.$$

Let us denote the global fine mesh as Ω_h, and the solution computed on Ω_h as $(\hat{u}, \hat{v}, \hat{w})^T$. The following lemma estimates the distance between u^l and $\hat{h}$ at the faces of Ω_h^l.

Lemma 2 *If the analytical solution of* (3) *is twice continuously differentiable, then the following holds*

$$\max_{(i,j)\in K^g \cup K^l} (|u_{1,j,k}^l - \hat{u}_{1,j,k}|) \leq C_2 H^2$$

Proof The distance can be estimated as a maximum of two distances

$$\max_{(i,j)}(|u^l_{1,j,k} - \hat{u}_{1,j,k}|) \leq \max(\max_{(i,j)\in K^g}(|u^l_{1,j,k} - \hat{u}_{1,j,k}|), \max_{(i,j)\in K^l}(|u^l_{1,j,k} - \hat{u}_{1,j,k}|)).$$

We recall that $\hat{u}_{1,j,k}$ is the solution of the discrete problem on the fine mesh. Further, when $(j,k) \in K^g$, $u^l_{1,j,k}$ is a solution discrete problem on coarse mesh. Since they both approximate the same exact solution $\bar{u}$, we have for some constant D_3

$$\max_{(j,k)\in K^g} |u^l_{1,j,k} - \bar{u}_{1,j,k}| \leq D_3 H^2, \quad \max_{(j,k)\in K^g} |\hat{u}_{1,j,k} - \bar{u}_{1,j,k}| \leq D_3 h^2.$$

Therefore

$$\max_{(j,k)\in K^g} |u^l_{1,j,k} - \hat{u}_{1,j,k}| \leq 2D_3 H^2.$$

We now look for an error estimate at fine grid points not being coarse grid points at the same time (i.e the set K^l). The estimate can be obtained as follows

$$\max_{(j,k)\in K^l} |u^l_{1,j,k} - \hat{u}_{1,j,k}| \leq \max_{(j,k)\in K^l} |u^l_{1,j,k} - \bar{u}_{1,j,k}| + \max_{(j,k)\in K^l} |\bar{u}_{1,j,k} - \hat{u}_{1,j,k}|. \tag{34}$$

The second term on the right-hand side of (34) is an approximation error, since $\hat{u}_{1,j,k}$ is the discrete solution on the fine mesh and $\bar{u}_{1,j,k}$ is the exact solution, i.e.

$$\max_{(j,k)\in K^l} |\bar{u}_{1,j,k} - \hat{u}_{1,j,k}| \leq D_3 h^2.$$

In order to obtain a similar estimate for the first term of (32) we introduce a reference function u^s which is found by a cubic spline interpolation of the coarse grid solution (20)

$$u^s_H = \sum_{(j,k)\in K^g} u_{1,j,k} c^H_{j,k}(y,z),$$

where $c^H_{j,k}(y,z)$ is cubic B-spline cf. [4] Then applying the triangle inequality we have

$$\max_{(j,k)\in K^l} |u^l_{1,j,k} - \bar{u}_{1,j,k}| \leq \max_{(j,k)\in K^l} |u^l_{1,j,k} - u^s_{1,j,k}| + \max_{(j,k)\in K^l} |u^s_{1,j,k} - \bar{u}_{1,j,k}| \tag{35}$$

The first component on the right hand side is simply the linear approximation error. An estimate for the second component follows from the fact that u^s can be represented as

$$u^s_H = \sum_{(j,k)\in K_g} u_{1,j,k} c^H_{j,k}(y,z) = \sum_{(j,k)\in K_g} (\bar{u}_{1,j,k} + D_{j,k} H^2) c^H_{j,k}(y,z),$$

where

$$u_{1,j,k} = \bar{u}_{1,j,k} + D_{j,k} H^2, \text{ for some } D_{j,k}.$$

Hence,

$$\max_{(j,k)\in K^l} |u^s_{1,j,k} - \bar{u}_{1,j,k}| \leq || \sum_{(j,k)\in X_g} u_{1,j,k} c^H_{j,k}(x,y,z) - \bar{u}||_\infty =$$
$$||(\sum_{(j,k)\in X_g} \bar{u}_{1,j,k} c^H_{j,k}(y,z) - \bar{u}) + \sum_{(j,k)\in X_g} D_{j,k} H^2 c^H_{j,k}(x,y,z)||_\infty \leq$$
$$D_3 4H^2 + \max_{(j,k)\in K^g} (D_{j,k}) H^2 = D_5 H^2,$$

for some D_4, D_5. Here D_4 is the error constant of spline approximation. We note that if $\bar{u} \in C^3(\Omega)$, then

$$\max_{(j,k)\in K^l} |u^s_{1,j,k} - \bar{u}_{1,j,k}| \leq D_4 H^3 + max(D_{j,k}) H^2.$$

due to the fact that the spline approximation yields a better accuracy for a smoother function. Having proved (34), we finally proved the statement of the lemma, with $C_2 = \max(2D_1, D_5)$. ■

We can now find the required estimate for the perturbation parameter α.

Property 1 *If the analytical solution of* (3) *is twice continuously differentiable, then the following holds*

$$\alpha \leq \frac{6C_2 S H^2}{(1-\sigma^{-2})(C_3 - 12C_1 h^2)},$$

where C_1 *and* C_2 *are constants from lemma 1 and lemma 2, respectively,*

$$S := 2[(x^l_{N^l_x} - x^l_1)(y^l_{N^l_y} - y^l_1) + (x^l_{N^l_x} - x^l_1)(z^l_{N^l_z} - z^l_1) + (y^l_{N^l_y} - y^l_1)(z^l_{N^l_z} - z^l_1)]$$

and $\sigma := \max(\sigma_x, \sigma_y, \sigma_z)$

Proof The expression in the square brackets in (23) can be considered as the sum over all points minus the sum over only global points

$$\sum_{(j,k)\in I^l} h_y h_z (|u^l_{1,j,k}| + |u^l_{N^l_x,j,k}|) + \sum_{(i,k)\in J^l} h_x h_z (|v^l_{i,1,k}| + |v^l_{i,N^l_y,k}|) +$$
$$\sum_{(i,j)\in K^l} h_x h_y (|w^l_{i,j,1}| + |w^l_{i,j,N^l_z}|) =$$
$$\sum_{j,k=1}^{N^l_y,N^l_z} h_y h_z (|u^l_{1,j,k}| + |u^l_{N^l_x,j,k}|) + \sum_{i,k=1}^{N^l_x,N^l_z} h_x h_z (|v^l_{i,1,k}| + |v^l_{i,N^l_y,k}|) + \sum_{i,j=1}^{N^l_x,N^l_y} h_x h_y (|w^l_{i,j,1}| + |w^l_{i,j,N^l_z}|) -$$
$$\sum_{(j,k)\in I^g} h_y h_z (|u^l_{1,j,k}| + |u^l_{N^l_x,j,k}|) - \sum_{(i,k)\in J^g} h_x h_z (|v^l_{i,1,k}| + |v^l_{i,N^l_y,k}|) -$$
$$\sum_{(i,j)\in K^g} h_x h_y (|w^l_{i,j,1}| + |w^l_{i,j,N^l_z}|). \tag{36}$$

We denote as u_1^l a twice continuously differentiable function that attains values $u_{1,j,k}^l$ at points (x_1^l, y_j^l, z_k^l), e.g. a cubic spline. Using the notation of the discrete integral norm we can rewrite (36) as follows

$$\sum_{(j,k)\in X_l} h_y h_z(|u_{1,j,k}^l| + |u_{N_x^l,j,k}^l|) + \sum_{(i,k)\in Y_l} h_x h_z(|v_{i,1,k}^l| + |v_{i,N_y^l,k}^l|) +$$
$$\sum_{(i,j)\in Z_l} h_x h_y(|w_{i,j,1}^l| + |w_{i,j,N_z^l}^l|) =$$
$$||u_1||_{L_1^h} + ||u_N||_{L_1^h} + ||v_1||_{L_1^h} + ||v_M||_{L_1^h} + ||w_1||_{L_1^h} + ||w_L||_{L_1^h} -$$
$$\sigma_y^{-1}\sigma_z^{-1}(||u_1||_{L_1^H} + ||u_N||_{L_1^H}) - \sigma_x^{-1}\sigma_z^{-1}(||v_1||_{L_1^H} + ||v_M||_{L_1^H}) -$$
$$\sigma_x^{-1}\sigma_y^{-1}(||w_1||_{L_1^H} + ||w_L||_{L_1^H}) \geq$$
$$(1-\sigma^{-2})[(||u_1||_{L_1} + ||u_{N_x^l}||_{L_1} + ||v_1||_{L_1} + ||v_{N_x^l}||_{L_1} + ||w_1||_{L_1} + ||w_{N_x^l}||_{L_1}) - 12C_1h^2].$$

This estimate follows directly from lemma 1. We denote $||u_1||_{L_1} + ||u_{N_x^l}||_{L_1} + ||v_1||_{L_1} + ||v_{N_x^l}||_{L_1} + ||w_1||_{L_1} + ||w_{N_x^l}||_{L_1}$ as C_3. The next step is to estimate δ. Note that the solution $(\hat{u}, \hat{v}, \hat{w})^T$ on the global fine mesh Ω^h satisfies the consistency condition (21) exactly

$$\sum_{j,k=1}^{N_y^l,N_z^l} h_y h_z(\hat{u}_{1,j,k} - \hat{u}_{N_x^l,j,k}) + \sum_{i,k=1}^{N_x^l,N_z^l} h_x h_z(\hat{v}_{i,1,k} - \hat{v}_{i,N_y^l,k}) + \sum_{i,j=1}^{N_x^l,N_y^l} h_x h_y(\hat{w}_{i,j,1} - \hat{w}_{i,j,N_z^l}) = 0. \quad (37)$$

Subtracting (21) from (37) we find that

$$\delta = \sum_{j,k=1}^{N_y^l,N_z^l} h_y h_z(u_{1,j,k}^l - u_{N_x^l,j,k}^l) + \sum_{i,k=1}^{N_x^l,N_z^l} h_x h_z(v_{i,1,k}^l - v_{i,N_y^l,k}^l) +$$
$$\sum_{i,j=1}^{N_x^l,N_y^l} h_x h_y(w_{i,j,1}^l - w_{i,j,N_z^l}^l) - \sum_{j,k=1}^{N_y^l,N_z^l} h_y h_z(\hat{u}_{1,j,k} - \hat{u}_{N_x^l,j,k}) -$$
$$\sum_{i,k=1}^{N_x^l,N_z^l} h_x h_z(\hat{v}_{i,1,k} - \hat{v}_{i,N_y^l,k}) - \sum_{i,j=1}^{N_x^l,N_y^l} h_x h_y(\hat{w}_{i,j,1} - \hat{w}_{i,j,N_z^l}).$$

Rearranging terms and applying the triangle inequality we obtain

$$\delta \leq \sum_{j,k=1}^{N_y^l,N_z^l} h_y h_z|u_{1,j,k}^l - \hat{u}_{1,j,k}| + \sum_{j,k=1}^{N_y^l,N_z^l} h_y h_z|u_{N_x^l,j,k}^l - \hat{u}_{N_x^l,j,k}| +$$
$$\sum_{i,k=1}^{N_x^l,N_z^l} h_x h_z|v_{i,1,k}^l - \hat{v}_{i,1,k}| + \sum_{i,k=1}^{N_x^l,N_z^l} h_x h_z|v_{i,N_y^l,k}^l - \hat{v}_{i,N_y^l,k}| +$$
$$\sum_{i,j=1}^{N_x^l,N_y^l} h_x h_y|w_{i,j,1}^l - \hat{w}_{i,j,1}| + \sum_{i,j=1}^{N_x^l,N_y^l} h_x h_y|w_{i,j,N_z^l}^l - \hat{w}_{i,j,N_z^l}|.$$

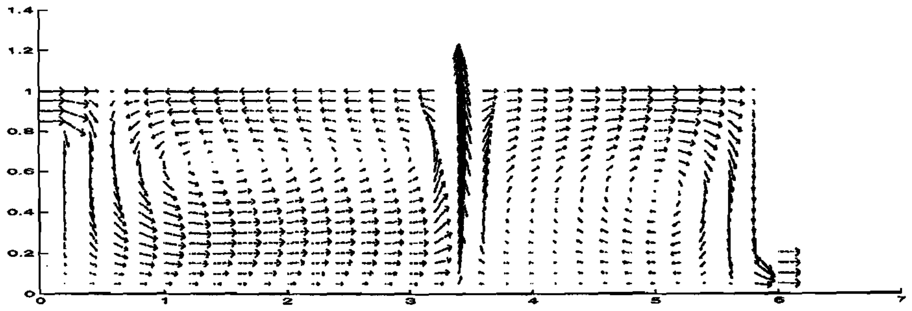

Figure 4: Typical pattern of the velocity field for bubbling

Using lemma 2 we have that

$$\delta \leq 6C_2SH^2. \tag{38}$$

■

6 Numerical Experiments

A good example to test our method for the glass oven is bubbling. Although this is a typically local phenomenon, it has a noticeable global effect. It works as follows. Air bubbles are injected at the bottom of the oven and on their way to the top attract air from the glass, thus making the glass medium more uniform. Another important feature of the bubbling is enforcing the so-called backflow, which is important for proper melting of all components. In figure 4 we plot a typical velocity field pattern when bubbling is used.

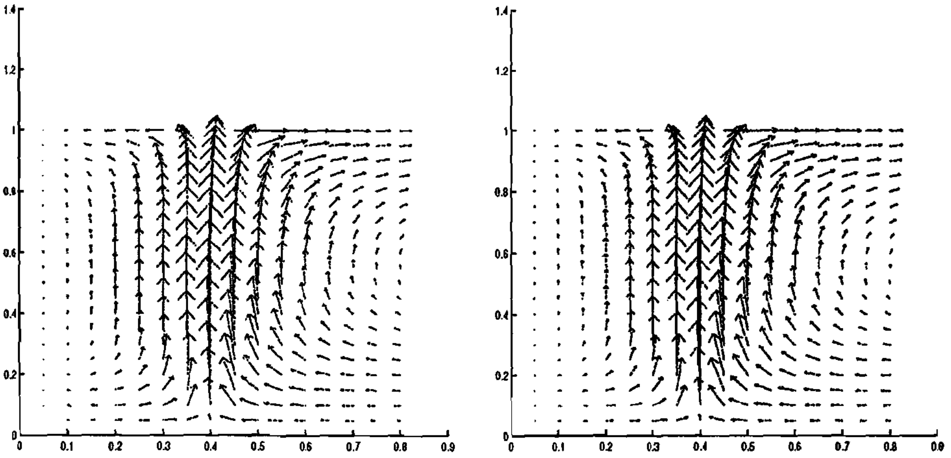

Figure 5: Velocity field in the vicinity of the bubbler, computed with non-uniform tensor refinement (left) and LDC (right), $x = 3.4$

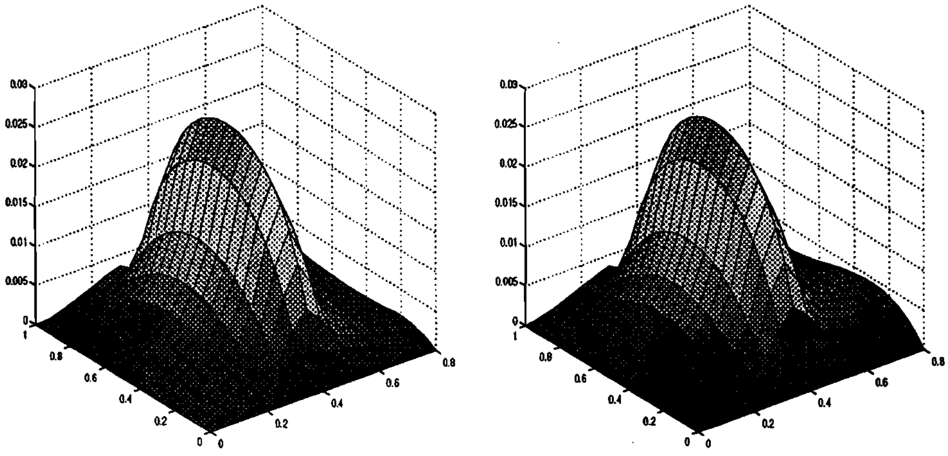

Figure 6: Absolute values of the velocity in the vicinity of the bubbler, computed with non-uniform tensor refinement (left) and LDC (right), $x = 3.4$

The bubbling is modelled as follows.

Let us assume that the initial volume of the bubble is V_0. Since the hydrostatic pressure decreases in going to the glass surface, the diameter of the bubble V_b increases. From the gas law we have

$$V_b = V_0 \frac{p_a + \rho g H}{p_a + \rho g(H - z)}, \tag{39}$$

where p_a is the ambient pressure, ρ the density of glass, g the gravitational constant, H the glass height and z the momentary glass height. We denote the total volume flow by Q. Then the distance between the bubbles is

$$d = \frac{V_0 v}{Q}, \tag{40}$$

where v is the bubble rising velocity, which is the sum of the undisturbed vertical glass velocity and the relative glass-bubble velocity. The force acting on the glass from one bubble is

$$F_b = -V_b \rho g. \tag{41}$$

Since per vertical meter, there are on average $1/d$ bubbles acting on the glass, the average force per meter height is

$$F = \frac{F_b}{d} = -\frac{V_b \rho g Q}{V_0 v}. \tag{42}$$

Let us denote the intersection of the bubble and the plane $z = z_k$ as B_k and the cell around the point (x_i, y_j) as $V_{i,j}$, i.e. $V_{i,j} := [x_i - h_x/2, x_i + h_x/2] \times [y_j - h_y/2, y_j + h_y]$. The body force is now corrected by

$$F^z_{i,j,k} = -\rho_{i,j,k} g + F \int\limits_{B_k \cap V_{i,j}} 1 dx dy / \int_{B_k} 1 dx dy. \tag{43}$$

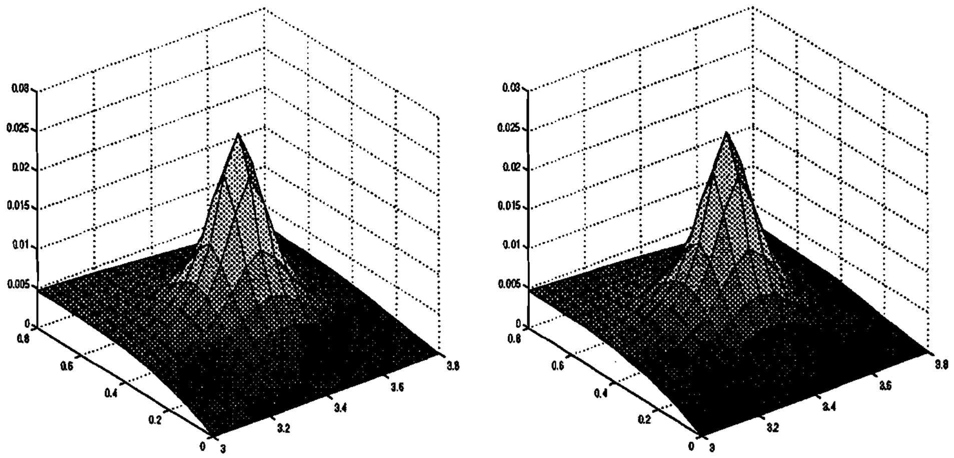

Figure 7: Absolute value of the velocity in the vicinity of the bubbler, computed with non-uniform tensor refinement (left) and LDC (right), $z = 0.5$

method	memory	time
tensor	36519	15120
LDC	6835	5213

Table 1: Computation cost for tensor refinement and LDC, oven with two bubblers

To put it another way, the bubbling is modelled by a local perturbation of the body force. The first example we consider is an oven of the size $6 \times 2 \times 1$, with two bubblers, located at the points $x = 3.4, y = 0.4$ and $x = 3.4, y = 1.6$, radius of both bubblers is equal to 0.06. Since the size of the bubbler is much smaller than the size of the oven, we need to either use uniform fine mesh everywhere, or refine locally. The uniform refinement is not feasible due to a huge overload. First we solve the problem refining in the vicinity of the bubblers by means of a non-uniform tensor grid. Then we solve the same problem but this time employing LDC and three uniform grids: the global coarse grid covering the whole oven and to local fine each covering one bubbler. Thus the local problems in LDC can be computed in parallel. The refinement factors are

$$\sigma_x = 4, \quad \sigma_y = 2, \sigma_z = 2.$$

We plot the results for both approaches in figures 5-7. Pictures on the left and the right sides are virtually identical. This means, that by using LDC we obtain the same results as when using a non-uniform tensor grid. However by using LDC we achieve a significant reduction of overload, see Table 1, namely approximately three times gain in storage and computational time.

The second example we consider is the same oven, but now with seven stirrers, equally distributed along the line $x = 3.4$. Again we compute the problem by means of two approaches: non-uniform tensor refinement and LDC. However, we use now only one local

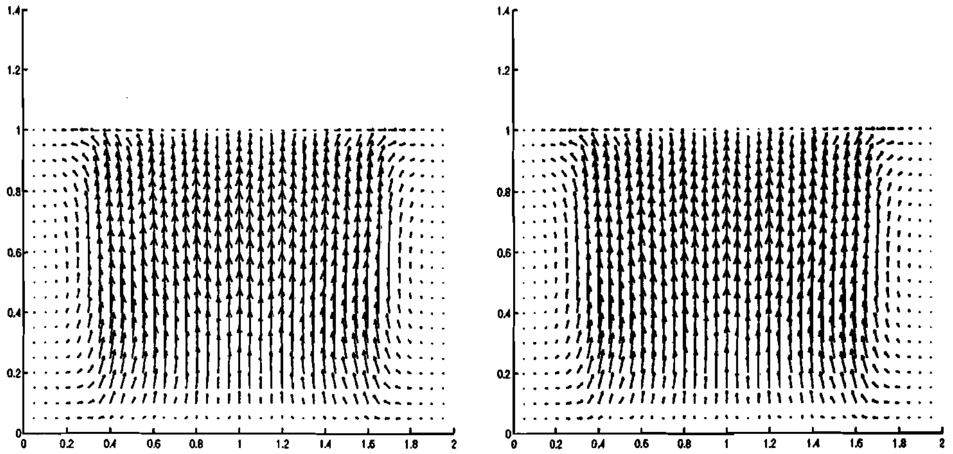

Figure 8: Velocity field in the vicinity of the bubblers, computed with non-uniform tensor refinement (left) and LDC (right), $x = 3.4$

method	memory	time
tensor	39231	17236
LDC	8029	6875

Table 2: Computation cost for tensor refinement and LDC, oven with 7 bubblers

grid for LDC. The results of the computation are depicted in figures (8)-(10). For this example LDC again has a considerable advantage in storage and time, see Table 2. However, since we used one larger local grid, the speed-up is slightly smaller. It can be explained by observing that doubling the number of grid points leads to more than two times increasing the computational time. We denote the solution obtained with LDC by $(u_{H,h}, v_{H,h}, w_{H,h})^T$ and the solution obtained with non-uniform tensor refinement by $(\hat{u}, \hat{v}, \hat{w})^T$. The distance between these solutions is measured by

$$e^u = \frac{||u_{H,h} - \hat{u}||_\infty}{||\hat{u}||_\infty}, \quad e^v = \frac{||v_{H,h} - \hat{v}||_\infty}{||\hat{v}||_\infty}, \quad e^w = \frac{||w_{H,h} - \hat{w}||_\infty}{||\hat{w}||_\infty}.$$

As can be seen from the table 3 the solutions differ about 10^{-2}, which corresponds to an interpolation error, that cannot be overcome. Therefore the result achieved is the best one we can expect from our method.

# bubblers	e^u	e^v	e^w
2	$2.34 \cdot 10^{-2}$	$2.01 \cdot 10^{-2}$	$9.85 \cdot 10^{-3}$
7	$2.17 \cdot 10^{-2}$	$1.85 \cdot 10^{-2}$	$8.76 \cdot 10^{-3}$

Table 3: Difference between solutions obtained by LDC and tensor refinement

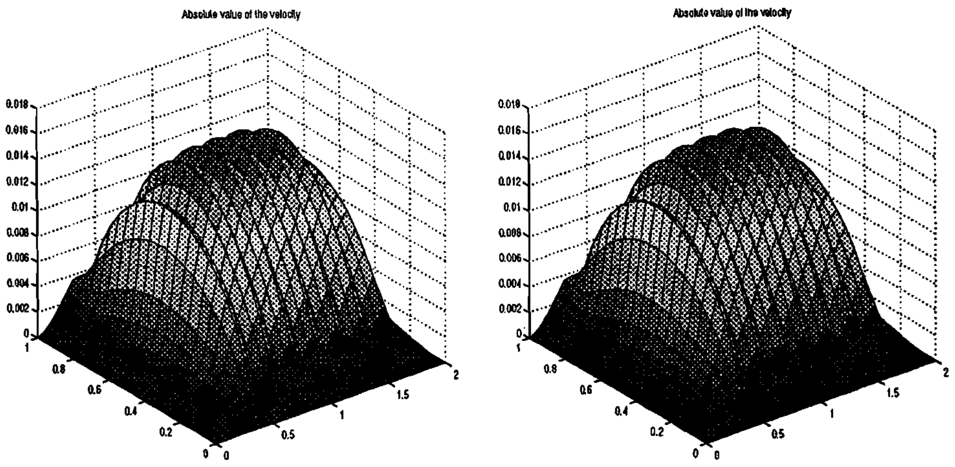

Figure 9: Absolute values of the velocity in the vicinity of the bubblers, computed with non-uniform tensor refinement (left) and LDC (right), $x = 3.4$

References

[1] M.J.H.Anthonissen, B. van 't Hof and A.A.Reusken, A finite volume scheme for solving elliptic boundary value problems on composite grids, *Computing* vol. 61, 1998, pp. 285-306.

[2] G.K.Batchelor, *An introduction to fluid dynamics*, Cambridge University Press, 1970.

[3] B.A.V.Bennett and M.D.Smooke, Local rectangular refinement with application to nonreacting and reacting fluid flow problems, *Journal of Computational Physics*, vol. 151, 1999, pp. 684-727.

[4] Ch.K.Chui, *Multivariate splines*, Society for Industrial and Applied Mathematics, Philadelphia, 1988.

[5] P.J.J.Ferket, *Solving boundary value problems on composite grids with an application to combustion*, Ph.D. Thesis, Eindhoven University of Technology, Eindhoven, 1996.

[6] P.J.J.Ferket and A.A.Reusken, Further analysis of the local defect correction method,*Computing* , vol. 56, 1996, pp. 117-139.

[7] J.H.Ferziger and M.Perić, *Computational methods for fluid dynamics*, Springer, 1996.

[8] L.Happel and H.Brenner, *Low Reynolds number hydrodynamics with special applications to particular media*, Prentice-Hall, 1965.

[9] W.Hackbusch, Local defect correction method and domain decomposition techniques, *Computing*, Suppl.5, 1984, pp. 89-113.

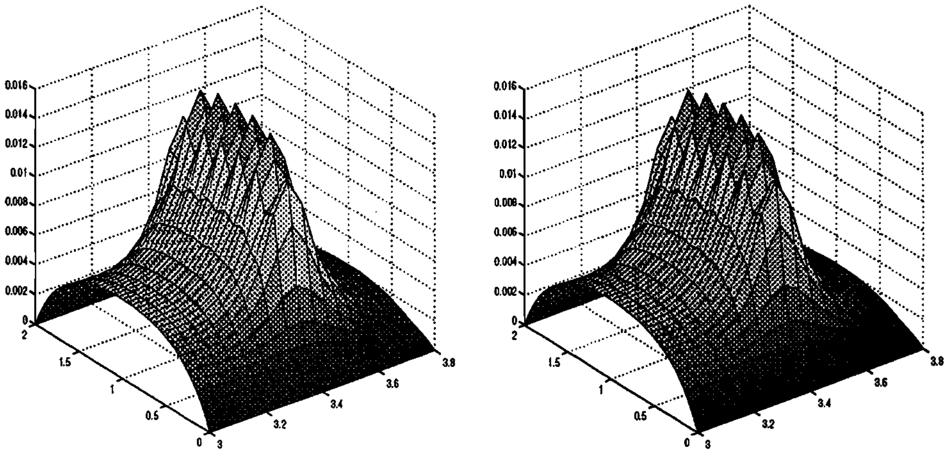

Figure 10: Absolute value of the velocity in the vicinity of the bubblers, computed with non-uniform tensor refinement (left) and LDC (right), $z = 0.5$

[10] R.I.Issa, Solution of the Implicitly Discretised Fluid Flow Equations by Operator-Splitting, *Journal of computational physics*, vol. 62, 1985, pp. 40-65.

[11] Ž.Lilek, S.Muzaferija, M.Perić and V.Seidl, An implicit finite-volume method using nonmatching blocks of structured grid, *Numerical Heat Transfer, Part B*, vol. 32, 1997, pp. 385-401.

[12] V.Nefedov, Simulation of glass flow in an oven, in Progress in Industrial Mathematics at ECMI98, B.G.Teubner, 1999, pp. 106-113.

[13] S.V.Patankar, *Numerical heat transfer and fluid flow*, Hemispere Publishing Corporation, 1980.

[14] A.Quarteroni and A.Valli, *Numerical approximation of partial differential equations*, Springer, 1994.

[15] B.Van 't Hof, J.H.M.Ten Thije Boonkamp and R.M.M.Mattheij, Pressure correction for laminar combustion simulation, *Combustion science and technology*, vol. 149, 1999, pp. 201-224.

[16] *NCNG Course on Glass Manufacture*, TNO Eindhoiven, Eindhoven, 1997.