DELFT UNIVERSITY OF TECHNOLOGY

REPORT 02-01

The Krylov accelerated SIMPLE(R) method for incompressible flow

C. Vuik and A. Saghir

ISSN 1389-6520

Reports of the Department of Applied Mathematical Analysis

Delft 2002

[The Krylov accelerated SIMPLE(R) method for incompressible flow]

[C. Vuik and A. Saghir]

September 10, 2002

Abstract

Numerical modeling of the melting and combustion process is an important tool in gaining understanding of the physical and chemical phenomena which occur in a gas- or oil-fired glass melting furnace. The incompressible Navier-Stokes equations are used to model the gas flow in the furnace. The discrete Navier-Stokes equations are solved by the SIMPLE(R) method. In our applications many SIMPLE(R) iterations are necessary to obtain an accurate solution. In this paper Krylov accelerated versions are proposed: GCR-SIMPLE(R). The properties of these methods are investigated for a simple two-dimensional flow. Thereafter the efficiency of the methods is compared for three-dimensional flows in industrial glass melting furnaces.

Keywords: SIMPLE(R) method, Krylov acceleration, efficiency, incompressible Navier-Stokes

1 Introduction

The increasing demand in quality, production efficiency and environmental issues drive glass producers in optimizing their melting furnaces. The quality demand is so high and the melting behavior so complex that a complete understanding of all important physical and chemical phenomena during the melting process is required to make advances. At the TNO Institute of Applied Physics a CFD simulation model for gas- and oil-fired glass melting furnaces has been developed. This is a complete model for glass melting furnaces, describing the combustion space and glass bath, and predicting the effects on melting performance and glass quality. The model is successfully used by the glass industry and furnace manufacturers for product quality improvement, optimization of furnace designs and trouble-shooting.

The simulation of a complete glass melting furnace often results in large computation times. One of the reasons for this is that the model uses the so-called SIMPLE(R) pressure-correction method to solve the incompressible Navier-Stokes equations. It is well known

that the SIMPLE(R) method often requires many iterations. To reduce the large computation times of the SIMPLE(R) method, a Krylov subspace acceleration of the SIMPLE(R) method has been developed. The GCR (Generalized Conjugate Residuals) method has been used for this purpose, since GCR can be applied to the non-symmetric matrices which result from discretization of the Navier-Stokes equations. The new method presented in this paper is called GCR-SIMPLE(R).

2 Description of the mathematical model

We compute numerical solutions of the steady state Navier-Stokes equations:

$$\begin{aligned} -\nu\Delta\mathbf{u} + \mathbf{u}\cdot\mathrm{grad}\mathbf{u} + \mathrm{grad}p &= \mathbf{f}, \\ -\mathrm{div}\mathbf{u} &= 0, \end{aligned}$$

combined with appropriate boundary conditions. The vector field $\mathbf{u}$ represents the velocity, p represents the pressure and ν is the viscosity. The Stokes equations are the Navier-Stokes equations without the non-linear term. After discretization of the incompressible Stokes equations one obtains the following linear system:

$$\begin{pmatrix} \mathbf{Q} & \mathbf{G} \\ \mathbf{G}^T & \mathbf{0} \end{pmatrix} \begin{pmatrix} u \\ p \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \end{pmatrix}, \tag{1}$$

where u is the algebraic vector of velocity components and the algebraic vector p contains the pressure unknowns. In the remainder of this paper, this system is abbreviated as $\mathbf{A}x = b$.

Discretization of the continuity equation leads to a zero block on the main diagonal of A, so the resulting linear system is indefinite and symmetric for the Stokes equations. This leads to serious problems when linear solvers are used. Various methods are known to overcome these difficulties: the pressure-matrix method [4], Uzawa method [17, 21, 6, 8], SIMPLE-type methods[14, 10], penalty method [2], pressure correction method [20], PISO method [1], etc. For an overview of these methods we refer to [16] Section 9.6. In CFD packages, a popular method is the SIMPLE method proposed by Patankar and Spalding [15] or one of its variants SIMPLER [14], SIMPLEST [26], or SIMPLEC [19].

In many applications the SIMPLE method needs many iterations before an accurate solution is reached. Various authors consider a multigrid acceleration of the SIMPLE method [24, 11, 3]. In this paper we consider a Krylov subspace acceleration of the SIMPLE(R) method [23]. The reason for this is that Krylov methods have only a small amount of overhead costs and are easy to implement in an existing CFD package. Although the discretized Stokes equation leads to a symmetric coefficient matrix, we use a Krylov subspace method suitable for non-symmetric matrices, because we also apply the resulting method to the discrete Navier-Stokes equations, where a non-symmetric coefficient matrix occurs. Recently, saddle point preconditioners for Krylov subspace methods have been proposed to solve the discretized Navier-Stokes equations. For a survey we refer to [9].

3 SIMPLE type methods as distributive iterative methods

In order to combine the SIMPLE-type methods with a Krylov solver we first write these methods as a distributive iterative method [26, 24].

3.1 SIMPLE method

The diagonal of the matrix $\mathbf{Q}$ is denoted by $\mathbf{D}$ and $\mathbf{R} \equiv -\mathbf{G}^T\mathbf{D}^{-1}\mathbf{G}$. The SIMPLE method as proposed by Patankar [14] is given by the following algorithm:

SIMPLE algorithm

1. Choose an initial estimate p^*.
2. Solve $\mathbf{Q}u^* = b_1 - \mathbf{G}p^*$.
3. Solve $\mathbf{R}\delta p = b_2 - \mathbf{G}^T u^*$.
4. Compute $u = u^* - \mathbf{D}^{-1}\mathbf{G}\delta p$ and $p := p^* + \delta p$.
5. If not converged take $p^* = p$ and go to 2.

The solutions of the systems given in 2 and 3 are obtained by a small number of iterations with a Block Gauss-Seidel method (TDMA solver [14]).

The SIMPLE method can also be seen as a distributive iterative method. Instead of solving the system $\mathbf{A}x = b$ the system $\mathbf{A}\mathbf{B}_\mathbf{R}y = b, x = \mathbf{B}_\mathbf{R}y$ will be solved. Choosing $\mathbf{B}_\mathbf{R}$ and $\mathbf{M}_\mathbf{R}$ as:

$$\mathbf{B}_\mathbf{R} = \begin{pmatrix} \mathbf{I} & -\mathbf{D}^{-1}\mathbf{G} \\ \mathbf{0} & \mathbf{I} \end{pmatrix}, \quad \mathbf{M}_\mathbf{R} = \begin{pmatrix} \mathbf{Q} & \mathbf{0} \\ \mathbf{G}^T & \mathbf{R} \end{pmatrix}, \tag{2}$$

and using the splitting $\mathbf{A}\mathbf{B}_\mathbf{R} = \mathbf{M}_\mathbf{R} - \mathbf{N}_\mathbf{R}$ the following iteration is obtained (SIMPLE method) [24]:

$$x^{k+1} = x^k + \mathbf{B}_\mathbf{R}{\mathbf{M}_\mathbf{R}}^{-1}(b - \mathbf{A}x^k), \quad k = 1, 2, ..., niter.$$

In the SIMPLE method the matrix $\mathbf{A}$ is multiplied from the right with $\mathbf{B}_\mathbf{R}$ and the resulting matrix is split. We can also use a multiplication from the left with $\mathbf{B}_\mathbf{L}$ and use the splitting $\mathbf{M}_\mathbf{L}$ where the matrices are given by

$$\mathbf{B}_\mathbf{L} = \begin{pmatrix} \mathbf{I} & \mathbf{0} \\ -\mathbf{G}^T\mathbf{D}^{-1} & \mathbf{I} \end{pmatrix}, \quad \mathbf{M}_\mathbf{L} = \begin{pmatrix} \mathbf{Q} & \mathbf{G} \\ \mathbf{0} & \mathbf{R} \end{pmatrix}. \tag{3}$$

Using the splitting $\mathbf{B}_\mathbf{L}\mathbf{A} = \mathbf{M}_\mathbf{L} - \mathbf{N}_\mathbf{L}$ we can use the iteration

$$x^{k+1} = x^k + {\mathbf{M}_\mathbf{L}}^{-1}\mathbf{B}_\mathbf{L}(b - \mathbf{A}x^k), \quad k = 1, 2, ..., niter.$$

This is a new SIMPLE type method, which we denote by SIMPLEL.

3.2 SIMPLER method

First the SIMPLER method is explained. Thereafter the SIMPLER method is written as a distributive iterative solver. Suppose the velocity vector u is known. Then an easy calculation shows that p is a solution of the system:

$$\mathbf{R}p = b_2 - \mathbf{G}^T\mathbf{D}^{-1}((\mathbf{D}-\mathbf{Q})u + b_1).$$

This idea is used in the SIMPLER method. When u^k is known, p^{k+1} and u^{k+1} are calculated as follows:

SIMPLER algorithm

1. Solve $\mathbf{R}p^* = b_2 - \mathbf{G}^T\mathbf{D}^{-1}((\mathbf{D}-\mathbf{Q})u^k + b_1)$.
2. Solve $\mathbf{Q}u^* = b_1 - \mathbf{G}p^*$.
3. Solve $\mathbf{R}\delta p = b_2 - \mathbf{G}^T u^*$.
4. Compute $u^{k+1} = u^* - \mathbf{D}^{-1}\mathbf{G}\delta p$ and $p^{k+1} = p^* + \delta p$.

One iteration of the SIMPLER algorithm is approximately 1.3 times as expensive as one SIMPLE iteration. Steps 2, 3, and 4 of both methods are comparable.

Using (3) it is easy to check that $\mathbf{N_L}$ is given by

$$\mathbf{N_L} = \begin{pmatrix} \mathbf{0} & \mathbf{0} \\ -\mathbf{G}^T\mathbf{D}^{-1}(\mathbf{D}-\mathbf{Q}) & \mathbf{0} \end{pmatrix}. \tag{4}$$

Now the first and second step of the SIMPLER algorithm can be written as:

$$\mathbf{M_L}\begin{pmatrix} u^* \\ p^* \end{pmatrix} = \mathbf{N_L}\begin{pmatrix} u^k \\ p^k \end{pmatrix} + \mathbf{B_L}\begin{pmatrix} b_1 \\ b_2 \end{pmatrix}. \tag{5}$$

This is equivalent to

$$\mathbf{M_L}\begin{pmatrix} u^* \\ p^* \end{pmatrix} = \mathbf{M_L}\begin{pmatrix} u^k \\ p^k \end{pmatrix} + \mathbf{B_L}\begin{pmatrix} b_1 \\ b_2 \end{pmatrix} - (\mathbf{M_L}-\mathbf{N_L})\begin{pmatrix} u^k \\ p^k \end{pmatrix}. \tag{6}$$

Multiplying this equation with ${\mathbf{M_L}}^{-1}$ leads to

$$\begin{pmatrix} u^* \\ p^* \end{pmatrix} = \begin{pmatrix} u^k \\ p^k \end{pmatrix} + {\mathbf{M_L}}^{-1}\mathbf{B_L}\left(\begin{pmatrix} b_1 \\ b_2 \end{pmatrix} - \mathbf{A}\begin{pmatrix} u^k \\ p^k \end{pmatrix}\right). \tag{7}$$

After this step a SIMPLE iteration follows. So the new solution is given by

$$\begin{pmatrix} u^{k+1} \\ p^{k+1} \end{pmatrix} = \begin{pmatrix} u^* \\ p^* \end{pmatrix} + \mathbf{B_R}{\mathbf{M_R}}^{-1}\left(\begin{pmatrix} b_1 \\ b_2 \end{pmatrix} - \mathbf{A}\begin{pmatrix} u^* \\ p^* \end{pmatrix}\right). \tag{8}$$

Combining both steps shows that one SIMPLER iteration can be denoted by

$$\begin{pmatrix} u^{k+1} \\ p^{k+1} \end{pmatrix} = \begin{pmatrix} u^k \\ p^k \end{pmatrix} + \mathbf{P_S} \left(\begin{pmatrix} b_1 \\ b_2 \end{pmatrix} - \mathbf{A} \begin{pmatrix} u^k \\ p^k \end{pmatrix} \right), \tag{9}$$

where $\mathbf{P_S}$ is given by

$$\mathbf{P_S} = \mathbf{B_R}{\mathbf{M_R}}^{-1} - \mathbf{B_R}{\mathbf{M_R}}^{-1} A {\mathbf{M_L}}^{-1}\mathbf{B_L} + {\mathbf{M_L}}^{-1}\mathbf{B_L}.$$

This expression can also be written as

$$\mathbf{P_S} = \mathbf{B_R}{\mathbf{M_R}}^{-1}{\mathbf{B_L}}^{-1} \begin{pmatrix} \mathbf{Q} & \mathbf{0} \\ \mathbf{0} & 2\mathbf{R} \end{pmatrix} {\mathbf{B_R}}^{-1}{\mathbf{M_L}}^{-1}\mathbf{B_L}.$$

Summarizing we note that the SIMPLER method can be written as a distributive iterative method:

$$x^{k+1} = x^k + \mathbf{B}_R\mathbf{M}_R^{-1}\mathbf{B}_L^{-1}\mathbf{T}\mathbf{B}_R^{-1}\mathbf{M}_L^{-1}\mathbf{B}_L(b - \mathbf{A}x^k),$$

where $\mathbf{T}$ is the block diagonal part of the matrix $\mathbf{M}_L + \mathbf{M}_R - \mathbf{A}$. Note that the SIMPLER method is closely related to the Symmetric Block Gauss-Seidel method.

Furthermore, when $\mathbf{Q}$ is symmetric it follows from the definition that $\mathbf{B}_L = \mathbf{B}_R^T$ and $\mathbf{M}_L = \mathbf{M}_R^T$, which implies that $\mathbf{P_S}$ is symmetric. So for the incompressible Stokes equation the MINRES method combined with the SIMPLER method as preconditioner can be used.

4 A saddle point preconditioner

In this section we describe one of the saddle point preconditioners proposed by Elman [7]. Thereafter we write the iteration matrices of this preconditioner and the SIMPLE preconditioner in the same way. This enables us to give a theoretical comparison of both preconditioners. In Section 6.2 these preconditioners are compared from numerical point of view.

In [7] saddle point preconditioners of the following form are considered:

$$\mathbf{P_E} = \begin{pmatrix} \mathbf{Q} & \mathbf{G} \\ \mathbf{0} & -\mathbf{X} \end{pmatrix}^{-1}. \tag{10}$$

It is easy to show that

$$\mathbf{A}\mathbf{P_E} = \begin{pmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{G}^T\mathbf{Q}^{-1} & \mathbf{G}^T\mathbf{Q}^{-1}\mathbf{G}\mathbf{X}^{-1} \end{pmatrix},$$

so that the eigenvalues are:

$$\{1\} \cup \sigma(\mathbf{G}^T\mathbf{Q}^{-1}\mathbf{G}\mathbf{X}^{-1}).$$

In [7] various choices for the matrix $\mathbf{X}$ are considered. It appears that a balance should be found between fast convergence, $\sigma(\mathbf{G}^T\mathbf{Q}^{-1}\mathbf{G}\mathbf{X}^{-1}) \approx \{1\}$, and the amount of work to apply the preconditioner to a vector.

For the SIMPLE preconditioner we can compute $\mathbf{M_R}^{-1}$:

$$\mathbf{M_R}^{-1} = \begin{pmatrix} \mathbf{Q}^{-1} & \mathbf{0} \\ -\mathbf{R}^{-1}\mathbf{G}^T\mathbf{Q}^{-1} & \mathbf{R}^{-1} \end{pmatrix}.$$

So the iteration matrix $\mathbf{A}\mathbf{B_R}\mathbf{M_R}^{-1}$ can be written as:

$$\begin{pmatrix} \mathbf{I} - (\mathbf{I} - \mathbf{Q}\mathbf{D}^{-1})\mathbf{G}\mathbf{R}^{-1}\mathbf{G}^T\mathbf{Q}^{-1} & (\mathbf{I} - \mathbf{Q}\mathbf{D}^{-1})\mathbf{G}\mathbf{R}^{-1} \\ \mathbf{0} & \mathbf{I} \end{pmatrix},$$

which implies that the eigenvalues of $\mathbf{A}\mathbf{B_R}\mathbf{M_R}^{-1}$ are:

$$\{1\} \cup \sigma(\mathbf{I} - (\mathbf{I} - \mathbf{Q}\mathbf{D}^{-1})\mathbf{G}\mathbf{R}^{-1}\mathbf{G}^T\mathbf{Q}^{-1}).$$

We note that the Elman preconditioner converges in one iteration when $\mathbf{X} = \mathbf{G}^T\mathbf{Q}^{-1}\mathbf{G}$, whereas the SIMPLE preconditioner converges in one iteration when $\mathbf{D} = \mathbf{Q}$. In general $\mathbf{X}$ is an approximation of $\mathbf{G}^T\mathbf{Q}^{-1}\mathbf{G}$ and $\mathbf{D}$ is an approximation of $\mathbf{Q}$. Comparing both preconditioners, it appears that for the SIMPLE preconditioner the iteration matrix converges to the identity matrix for $\mathbf{D} \to \mathbf{Q}$, whereas the iteration matrix for the Elman preconditioner converges to a block lower triangular matrix (so the iteration matrix is non-normal) for $\mathbf{X} \to \mathbf{G}^T\mathbf{Q}^{-1}\mathbf{G}$.

5 The GCR acceleration method

In this section a Krylov acceleration of the preconditioners given in Section 3 and 4 is given. Many Krylov subspace methods are known to solve non-symmetric linear systems. We choose the GCR method [5] because the method is robust, minimizes the residual and allows a variable preconditioner [18, 22]. This final property is very important, since in practice the inverse of the matrices occurring in the preconditioners are only computed approximately. So, the preconditioner $\mathbf{P}^k$ is a different operator in every iteration.

GCR algorithm

$r^0 = b - \mathbf{A}x^0$

for $k = 0, 1, ..., ngcr$

$s^{k+1} = \mathbf{P}^k r^k$

$v^{k+1} = \mathbf{A}s^{k+1}$

for $i = 0, 1, ..., k$

$v^{k+1} = v^{k+1} - (v^{k+1}, v^i)v^i$

$s^{k+1} = s^{k+1} - (v^{k+1}, v^i)s^i$

end for

$v^{k+1} = v^{k+1}/\|v^{k+1}\|_2$
$s^{k+1} = s^{k+1}/\|v^{k+1}\|_2$
$x^{k+1} = x^k + (r^k, v^{k+1})s^{k+1}$
$r^{k+1} = r^k - (r^k, v^{k+1})v^{k+1}$

end for

For $\mathbf{P}^k$ one can choose $\mathbf{B_R}{\mathbf{M_R}}^{-1}$ (GCR-SIMPLE), $\mathbf{P_S}$ (GCR-SIMPLER), or $\mathbf{P_E}$ (GCR-Elman). Due to the modified Gram-Schmidt orthogonalization, the amount of work and memory increases when the number of iterations grows. To bound these quantities the method is truncated or restarted after a small number of iterations. Comparing the amount of work with that of the SIMPLE method, we note that if the GCR-SIMPLE method is restarted after $ngcr$ iterations, then $ngcr^2$ vector-updates and $ngcr^2/2$ inner-products extra are required. Furthermore, an additional $2ngcr$ vectors should be stored in memory. When $ngcr$ is small these costs are negligible.

For the SIMPLE(R) preconditioners it appears that a diagonal scaling is beneficial for the convergence of the GCR-SIMPLE(R) method. To implement this the following adaptations are made: compute $\mathbf{D}_{AB} = \mathrm{diag}(\mathbf{AB_R})$ and use $r^0 = \mathbf{D}_{AB}^{-1}(b - \mathbf{A}x^0)$, $s^{k+1} = \mathbf{B}\mathbf{M}_k^{-1}\mathbf{D}_{AB}r^k$, and $v^{k+1} = \mathbf{D}_{AB}^{-1}\mathbf{A}s^{k+1}$.

The discretization of the Navier-Stokes equations gives a non-linear system due to the convection terms. The discretization equations for the velocities can be written as follows:

$$\mathbf{Q}(u)u + \mathbf{G}p = b_1. \tag{11}$$

Various methods can be chosen to linearize $\mathbf{Q}(u)$, like the Newton-Raphson method or the Picard iteration method. We have used the Picard iteration method where $\mathbf{Q}(u^{k+1})$ is approximated by $\mathbf{Q}(u^k)$. A non-symmetric linear system is obtained with the same structure as the discretized Stokes equations. Now, the GCR-SIMPLER algorithm for the Navier-Stokes equations can be summarized as follows:

x^0 guessed value

for $k = 0, 1, 2, \ldots, niter$

solve $\mathbf{A}(x^k)x^{k+1} = b$ with GCR-SIMPLER

end for

The other preconditioners can be used in the same way. During each iteration we do not need to solve this equation until convergence because the matrix $\mathbf{A}$ is defined using an approximation of x^k. This has the advantage that a small value of $ngcr$ can be chosen which leads to low memory requirements. The optimal value of $ngcr$ can be different for each problem.

6 Numerical experiments

In this section attention will be given to the application of the various methods to solve the incompressible Navier-Stokes equations. We will first investigate the properties of SIMPLE(R) and GCR-SIMPLE(R) for a 2D Navier-Stokes flow between two flat plates. Thereafter we compare the GCR method using the preconditioners proposed in Section 3 and 4 for a Poiseuille flow and a backward facing step problem. Finally, in order to compare the efficiency for more realistic test problems, the Ford Nashville float glass furnace is used.

In the measurements the following quantities are used:

- *CPUtime*: execution time of a used method measured in seconds on an HP-735 in Section 6.1 and on an HP-J210 in Section 6.3,
- *residu*: absolute sum of residuals for a given variable,
- *niter*: number of iterations.

6.1 Numerical properties of the SIMPLE type methods

In this section we present some results obtained when applying the SIMPLER and the GCR-SIMPLER method to the flow between two flat plates with distance $D = 10\ cm$ and length $L = 500\ cm$. For this test problem an equidistant grid will be used.

To apply the SIMPLER and the GCR-SIMPLER methods we first define some default values of parameters used in these methods. For both methods the termination criterion is: stop when the sum of the absolute residuals of each variable is less than or equal to 10^{-6}. The relaxation factor for the pressure is always 1. The SIMPLER method will be used with relaxation factors equal to 0.8 for the velocities u_1 and u_2. For the GCR-SIMPLER method *ngcr* is taken equal to 3 and the relaxation factors for the velocities u_1 and u_2 are equal to 1. The default TDMA solver is PLANE TDMA. For a motivation of these values we refer to [23].

First we investigate the dependence of GCR-SIMPLER on the value of *ngcr*. The results are given in Table 1. When *ngcr* increases the number of GCR-SIMPLER iterations decreases, but every iteration becomes more expensive. On the 40×20 grid we see that the CPU time is more or less the same for all values of *ngcr*. For the 40×40 grid there are larger differences. The choice $ngcr = 14$ leads to a minimal amount of CPU time; however, many vectors should be stored in memory. Therefore the value $ngcr = 3$ is a good compromise. When convergence problems occur for the GCR-SIMPLER method it helps to increase the value of *ngcr*.
In Table 2 the results are given for various grid sizes. Both methods need more iterations when the grid size increases. For a small grid size the CPU times are comparable, whereas for a large grid size GCR-SIMPLER needs less CPU time than the SIMPLER method. For the 20×20 grid the aspect ratio of the finite volumes is equal to 50.

ngcr	Grid 40×20		Grid 40×40	
	niter	CPU time	*niter*	CPU time
2	43	10.3	96	39.9
3	33	9.9	67	35.6
4	30	10.4	59	37.4
6	21	9.8	38	33.4
8	17	9.9	31	35
14	11	10.6	14	27.5

Table 1: Results of the GCR-SIMPLER method for various values of *ngcr*

Grid size	SIMPLER		GCR-SIMPLER	
	niter	CPU time	*niter*	CPU time
20×20	61	5.2	29	5.9
40×20	139	16.9	33	9.9
80×20	303	68.5	80	40.2

Table 2: Results for various grid sizes

6.2 Comparison of the various preconditioners

In this section we present convergence results for the preconditioned GCR method, without restarting or truncation, applied to some test problems. These experiments are done in Matlab. For the discretization we use the Matlab codes [1] given in [25]. In our SIMPLE(R) preconditioners we always take $\mathbf{D} = diag(\mathbf{Q})$. From [7] it appears that $\mathbf{X} = \frac{h^d}{\nu}\mathbf{I}$, for a uniform grid of width h in d dimensions, is a good preconditioner as long as $\nu \approx 1$. Since this preconditioner is easy to use, we take $\mathbf{X} = \gamma\mathbf{I}$, where γ is chosen such that the convergence of GCR-Elman is optimal. We cannot use $\mathbf{X} = \frac{h^d}{\nu}\mathbf{I}$ because the step-size in x and y-direction is different. We only report number of iterations to observe the convergence behavior. All inverse matrices in the preconditioners are computed exactly. In real applications approximate inverses should be used in order to save flops and memory.

Channel flow

Our first test problem in this section is a flow in a channel with width 2 and the length is varied. At the inflow boundary we use a parabolic profile, whereas a no-slip boundary condition is used at the side walls. As outflow boundary conditions we use homogeneous Neumann conditions for both velocity components. For the discretization we use a staggered grid arrangement [25, 12]. The Dirichlet boundary conditions are included as extra equations in the linear system. We start the GCR method with $x^0 = 0$ and stop if $\frac{\|r^k\|}{\|b\|} \leq \epsilon$. As a default we take $\epsilon = 10^{-6}$.

[1] ta.twi.tudelft.nl/users/wesselin/cfdbook.html

In many channel flows the length of the channel is much larger than the width of the channel. From our experiments we know that a number of preconditioners break down or have a bad convergence behavior when stretched cells occurs. Therefore we vary the length of the channel two orders of magnitude and report the results for the proposed preconditioners in Table 3. These results are obtained for the Stokes equations on a 16×16 grid. Note that the number of iterations increases for the ILU and Elman preconditioners for increasing length, whereas the number of iterations is constant or decreasing for the SIMPLE(R) preconditioners.

Length	2	20	200
ILU	57	62	91
SIMPLE	18	22	9
SIMPLER	9	11	6
Elman	12	22	31

Table 3: Number of iterations of the preconditioned GCR method for the Stokes equations

We have also used the Oseen equations:

$$\begin{aligned} -\nu\Delta\mathbf{u} + \mathbf{w}\cdot\mathrm{grad}\mathbf{u} + \mathrm{grad}p &= \mathbf{f}, \\ -\mathrm{div}\mathbf{u} &= 0, \end{aligned}$$

where $\mathbf{w}$ is a given velocity field. These equations are used during a Picard iteration to solve the non-linear Navier-Stokes equations. We take $\mathbf{w}$ equal to the velocity of the Poiseuille flow. From Table 4 and 5 it appears that the number of iterations increases for decreasing ν. The behavior for increasing length is comparable for the Stokes and Oseen equations.

Length	2	20	200
ILU	57	63	94
SIMPLE	19	24	11
SIMPLER	9	13	7
Elman	12	25	39

Table 4: Number of iterations of the preconditioned GCR method for the Oseen equations for $\nu = 1$

Finally, solving the Navier-Stokes equations with a Picard iteration the intermediate Stokes and Oseen equations are solved with a relative large ϵ. Therefore we also give the results for $\epsilon = 10^{-2}$ in Table 6. It appears that for a low accuracy there is a large growth in the number of iterations for the ILU and Elman preconditioners for increasing length of the channel.

Length	2	20	200
ILU	53	91	101
SIMPLE	20	30	17
SIMPLER	8	16	12
Elman	26	49	53

Table 5: Number of iterations of the preconditioned GCR method for the Oseen equations for $\nu = 0.1$

Length	2	20	200
ILU	23	54	81
SIMPLE	6	9	4
SIMPLER	5	9	7
Elman	2	32	43

Table 6: Number of iterations of the preconditioned GCR method for the Oseen equations for $\nu = 0.1$ and $\epsilon = 10^{-2}$

Backward facing step

Our second test problem is a backward facing step problem. The length of the channel is 25 and the width is 2. At the inflow boundary we have a parabolic profile at the upper half part and a no-slip condition at the lower half part of the boundary. In the Oseen equations we take $\mathbf{w}$ equal to zero in the lower part of the channel and equal to the Poiseuille flow velocities in the upper part of the channel. The horizontal velocity contours of $\mathbf{w}$ and $\mathbf{u}$ are given in Figure 1 and 2.

The iteration counts are given in Table 7. Note that the SIMPLER preconditioner is insensitive to the choice of ν. For the other preconditioners the number of iterations increases for decreasing ν. This behavior is also reported in [7] for the Elman preconditioner.

	Stokes	Oseen	
Preconditioner		$\nu = 0.25$	$\nu = 0.025$
ILU	77	77	136
SIMPLE	47	52	66
SIMPLER	16	17	18
Elman	34	41	105

Table 7: Number of iterations of the preconditioned GCR method for the backward facing step problem

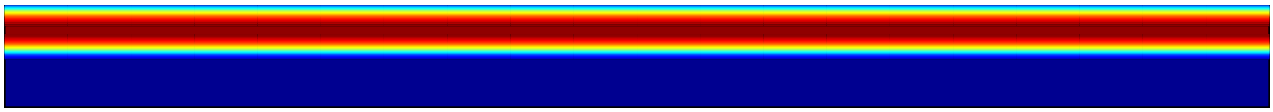

Figure 1: Contour plot of horizontal component of $\mathbf{w}$ used in the backward facing step problem

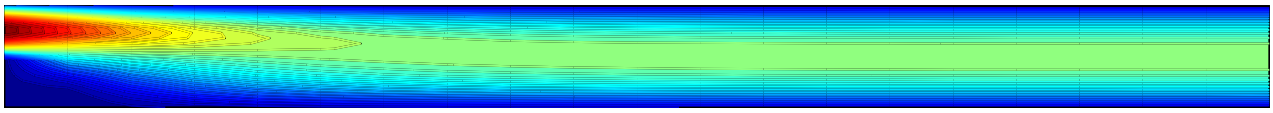

Figure 2: Contour plot of the horizontal component of the solution $\mathbf{u}$ of the backward facing step problem

6.3 The Ford Nashville furnace

In this section the SIMPLER and the GCR-SIMPLER method are used to simulate the combustion chamber of the Ford furnace [13]. In the model for this furnace, the combustion of natural gas is described by the conserved scalar approach to high temperature, non-premixed combustion and the chemistry is described with a one-step global reaction. The geometry of the Ford furnace is sketched in Figure 3. The internal length, width and maximum height of the combustion chamber are $34.7 \times 10.1 \times 2.3\ m$. The same convergence

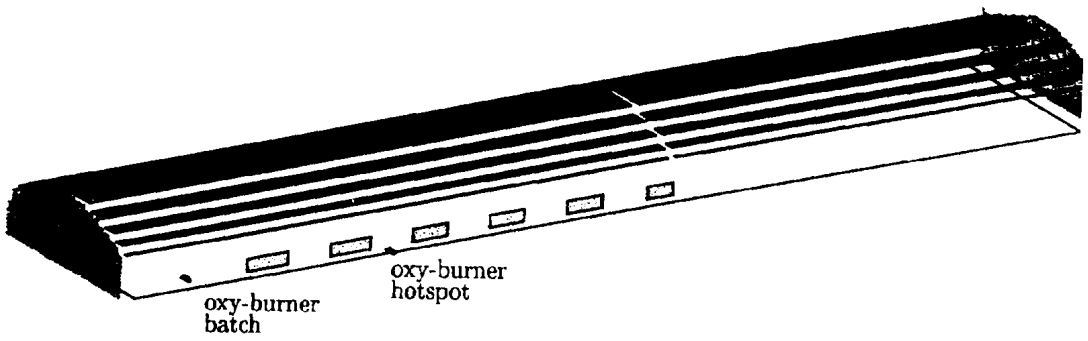

Figure 3: Geometry of the Ford float glass furnace

criterion is used for each method. In this problem the iteration process is stopped when the absolute sum of the residuals of each variable is less than or equal to 10^{-4}. The finite volume grid consists of $130 \times 40 \times 40 = 208000$ points. The same relaxation factors are used for both methods. In the SIMPLER and the GCR-SIMPLER method the same SPACE TDMA solver is used. The first simulation has been done using the GCR-SIMPLER method. The results are: $niter = 3390$, $CPUtime \approx 3.3$ days. Using the SIMPLER method the simulation has been stopped after 7.5 days, because the maximum number of iterations has been reached. We see again a large decrease in CPU time when the Krylov

acceleration is used. The temperature contours in a plane just above the glass surface are shown in Figure 4. A comparison of the computed and measured quantities for this furnace is given in [13]. For more results concerning the convergence behavior of the various methods applied to the IFRF furnace we refer to [23].

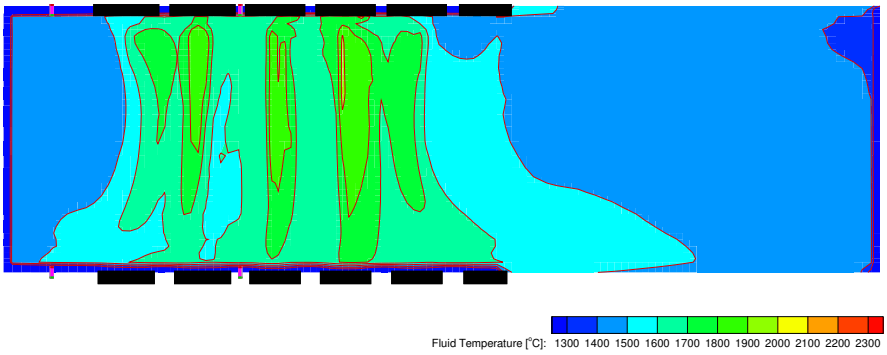

Figure 4: The temperature contours of the Ford float furnace using the GCR-SIMPLER method

6.4 Memory

Using the GCR-SIMPLER method instead of the SIMPLER method leads to more memory. In Table 8 the memory requirements are given for various problems. For a three-dimensional problem the increase is approximately 50 %. When $ngcr$ is increased the CPU time may decrease but the memory requirements increase.

problem		SIMPLER	GCR-SIMPLER ($ngcr = 3$)
Flat plates	120×120	31	39
IFRF furnace	$42 \times 37 \times 27$	52	78
Ford furnace	$130 \times 40 \times 40$	202	333

Table 8: Memory requirements for various problems measured in Megabytes

7 Conclusions

An efficient method to simulate glass-melting furnaces is considered. In this method the incompressible Navier-Stokes equations are used. SIMPLE-type methods are very popular

to solve the discretized incompressible Navier-Stokes equations. In this paper SIMPLE and SIMPLER are rewritten as classical distributive iteration methods for linear systems. Two other preconditioners are considered ILU and Elman [7]. As Krylov acceleration the GCR method is used.

Firstly the dependence of the GCR-SIMPLE(R) methods on grid-size and $ngcr$ is investigated by numerical experiments. These insights are used to propose a number of default parameters ($ngcr = 3$, TDMA solver, outlet boundary condition, etc.) for these methods.

Secondly the convergence behavior of the ILU, SIMPLE(R), and Elman preconditioner are compared for two test problems: a channel flow and a backward facing step flow. It appears that GCR-SIMPLER is only weakly dependent on the stretching of the grid and the viscosity. Points of current research are: inclusion of more advanced preconditioners proposed by [7] and comparison of the memory requirements and CPU time of the proposed preconditioners for practical problems.

Finally, the efficiency of SIMPLER and GCR-SIMPLER is compared using a simulation of an industrial furnace. For these simulations, where the grid has high aspect ratios, the GCR-SIMPLER method appears to be three times as fast as the SIMPLER method. Furthermore, larger relaxation factors can be used for GCR-SIMPLER, which leads to a still higher efficiency. After convergence the quality of the computed results (velocity, pressure, turbulence quantities, etc) is comparable. GCR-SIMPLER requires more memory than the SIMPLER method.

References

[1] I.E. Barten. Comparison of SIMPLE- and PISO-type algorithms for transient flows. *Int J. Num. Meth. in Fluids*, 26:459–483, 1998.

[2] M. Bercovier. Perturbation of a mixed variational problem, applications to mixed finite element methods. *R.A.I.R.O. Anal. Numér.*, 12:211–236, 1978.

[3] D. Braess and D. Sarazin. An efficient smoother for the Stokes problem. *App. Num. Math.*, 23:3–19, 1997.

[4] J. Cahouet and J.P. Chabard. Some fast 3-D finite element solvers for the generalized Stokes problem. *Int. J. Num. Methods Fluids*, 8:869–895, 1988.

[5] S.C. Eisenstat, H.C. Elman, and M.H. Schultz. Variational iterative methods for nonsymmetric systems of linear equations. *SIAM J. Num. Anal.*, 20:345–357, 1983.

[6] H.C. Elman. Multigrid and Krylov subspace methods for the discrete Navier-Stokes equations. *Int. J. Num. Meth. Fluids*, 22:755–770, 1996.

[7] H.C. Elman. Preconditioning for the steady-state Navier-Stokes equations with low viscosity. *SIAM J. Sci. Comput.*, 20:1299–1316, 1999.

[8] H.C. Elman and G.H. Golub. Inexact and preconditioned Uzawa algorithms for saddle-point problems. *SIAM J. Num. Anal.*, 31:1645–1661, 1994.

[9] H.C. Elman, D.J. Silvester, and A.J. Wathen. Performance and analysis of saddle point preconditioners for the discrete steady-state Navier-Stokes equations. *Numer. Math.*, 90:665–688, 2002.

[10] J.H. Ferziger and M. Peric. *Computational methods for fluid dynamics (second edition).* Springer, Berlin, 1999.

[11] T. Gjesdal and M.E.H. Lossius. Comparison of pressure correction smoothers for multigrid solution of incompressible flow. *Int. J. Num. Meth. Fluids*, 25:393–405, 1997.

[12] F.H. Harlow and J.E. Welch. Numerical calculation of time-dependent viscous incompressible flow of fluid with a free surface. *The Physics of Fluids*, 8:2182–2189, 1965.

[13] A.M. Lankhorst, G.P. Boerstoel, H.P.H. Muysenberg, and K.K. Koram. Complete simulation of the glass tank and combustion chamber of the former Ford Nashville float furnace including melter, refiner and working end. In *Fourth Int. Seminar on Mathematical Simulation in the Glass Melting*, 1997.

[14] S.V. Patankar. *Numerical heat transfer and fluid flow.* McGraw-Hill, New York, 1980.

[15] S.V. Patankar and D.B. Spalding. A calculation procedure for heat and mass transfer in three-dimensional parabolic flows. *Int. J. Heat Mass Transfer*, 15:1787–1806, 1972.

[16] A. Quarteroni and A. Valli. *Numerical approximation of partial differential equations.* Springer series in Computational Mathematics 23. Springer Verlag, Berlin, 1994.

[17] R. Temam. *Navier-Stokes equations. Theory and Numerical Analysis, 3rd edition.* North-Holland, Amsterdam, 1984.

[18] H.A. van der Vorst and C. Vuik. GMRESR: a family of nested GMRES methods. *Num. Lin. Alg. Appl.*, 1:369–386, 1994.

[19] J.P. Van Doormaal and G.D. Raithby. Enhancements of the SIMPLE method for predicting incompressible fluid flows. *Num. Heat Transfer*, 7:147–163, 1984.

[20] J. van Kan. A second-order accurate pressure-correction scheme for viscous incompressible flow. *SIAM J. Sci. Stat. Comput.*, 7:870–891, 1986.

[21] C. Vincent and R. Boyer. A preconditioned conjugate gradient Uzawa-type method for the solution of the Stokes problem by mixed Q1-P0 stabilized finite elements. *Int. J. Num. Meth. Fluids*, 14:289–298, 1992.

[22] C. Vuik. New insights in GMRES-like methods with variable preconditioners. *J. Comp. Appl. Math.*, 61:189–204, 1995.

[23] C. Vuik, A. Saghir, and G.P. Boerstoel. The Krylov accelerated SIMPLE(R) method for flow problems in industrial furnaces. *Int. J. for Num. Meth. Fluids*, 33:1027–1040, 2000.

[24] P. Wesseling. *An introduction to multigrid methods*. Wiley, Chichester, 1992. available at www.mgnet.org/mgnet-books-wesseling.html.

[25] P. Wesseling. *Principles of Computational Fluid Dynamics*. Springer Series in Computational Mathematics, Vol.29. Springer, Heidelberg, 2001.

[26] G. Wittum. Multi-grid methods for Stokes and Navier-Stokes equations with transforming smoothers: Algorithms and numerical results. *Numer. Math.*, 54:543–563, 1989.