INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS, VOL. 24, 1111–1128 (1997)

A COMPARATIVE STUDY OF PRESSURE CORRECTION AND BLOCK-IMPLICIT FINITE VOLUME ALGORITHMS ON PARALLEL COMPUTERS

M. KURRECK* AND S. WITTIG

Lehrstuhl und Institut für Thermische Strömungsmaschinen, Universität Karlsruhe (TH), D-76128 Karlsruhe, Germany

SUMMARY

A comparison of a new parallel block-implicit method and the parallel pressure correction procedure for the solution of the incompressible Navier–Stokes equations is presented. The block-implicit algorithm is based on a pressure equation. The system of non-linear equations is solved by Newton's method. For the solution of the linear algebraic systems the Bi-CGSTAB algorithm with incomplete lower–upper (ILU) decomposition of the matrix is applied. Domain decomposition serves as a strategy for the parallelization of the algorithms. Different algorithms for the parallel solution of the linear system of algebraic equations in conjunction with the pressure correction procedure are proposed. Three different flows are predicted with the parallel algorithms. Results and efficiency data of the block-implicit method are compared with the parallel version of the pressure correction algorithm. The block-implicit method is characterized by stable convergence behaviour, high numerical efficiency, insensitivity to relaxation parameters and high spatial accuracy. © 1997 by John Wiley & Sons, Ltd.

Int. J. Numer. Meth. Fluids, **24**: 1111–1128, 1997.

No. of Figures: 18. No. of Tables: 1. No. of Refs: 28.

KEY WORDS: pressure correction; block-implicit; finite volume; parallel

1. INTRODUCTION

The lack of an equation for the determination of the pressure distribution and the decoupling of the velocity and the pressure field are the major reasons for the large numerical effort required for the prediction of incompressible flows. Various algorithms, such as pressure correction procedures[1–4] and the artificial compressibility method,[5] often suffering from weak coupling of the velocity and the pressure field, have been developed. Furthermore, the convergence characteristics of these methods depend greatly on user-defined parameters.

In contrast with decoupled procedures, the fully implicit coupled approach is characterized by stability and insensitivity to user-defined parameters. Several researchers have developed algorithms for the coupled solution of the Navier–Stokes equations where the coupling of the equations has been carried out pointwise,[6] linewise[7] and fully implicitly.[8–10]

The parallelization of an algorithm with the domain decomposition approach further enhances the decoupling of the momentum and the continuity equation. In order to retain high parallel efficiency, the data transfer between the subdomains has to be reduced. Therefore the solutions in the

* Correspondence to: M. Kurreck.

CCC 0271–2091/97/111111–18 $17.50 *Received 9 November 1995*

subdomains are decoupled and the numerical efficiency of the algorithm decreases.[11,12] A means to stabilize the convergence behaviour of the pressure correction procedure for the prediction of two-dimensional laminar flows is to intensify the data transfer between the subdomains.[13]

The fully coupled solution of the Navier–Stokes equations in conjunction with domain decomposition also offers new opportunities to obtain a stable and efficient parallel algorithm. The development of a parallel block-implicit algorithm as well as the parallelization of the SIMPLEC pressure correction procedure and the application of both algorithms for the prediction of laminar and demanding turbulent flows are the aims of this study. The fully coupled algorithm and the parallel solution procedure are described in detail. For the parallel pressure correction procedure, three different methods for coupling the subdomains are outlined. Three different flows are predicted and results are compared with experimental data. The results of an efficiency analysis are presented and discussed.

2. DISCRETIZATION AND PRESSURE CORRECTION PROCEDURE

Before the fully implicit coupled algorithm is described in detail, a short overview of the governing equations, the discretization and the pressure correction algorithm is given. Three-dimensional steady flows are governed by the Navier–Stokes equations, turbulence is described by the standard k–ϵ model[14] and for the prediction of flows with heat release the eddy dissipation model[15] is applied. All transport equations can be written in the form

$$\mathrm{div}(\rho\vec{c}\phi) = \mathrm{div}(\Gamma_\phi \cdot \mathrm{grad}\phi) + S_\phi, \tag{1}$$

where ϕ denotes the transport variable, $\vec{c}$ is the velocity vector and the source term is denoted S_ϕ. The finite volume method is applied to discretize the partial differential equations. The computational domain is subdivided into regular control volumes and the transport equations are integrated over the volumes (see Figure 1).

The diffusive fluxes and the source terms are discretized by the central differencing scheme. In contrast, the convective-fluxes are discretized by the monotonized linear upwind (MLU) scheme of second-order accuracy developed by Noll.[16]

The pressure field is determined with the SIMPLEC algorithm[2] on non-staggered grids in conjunction with the interpolation of the cell face velocities proposed by Rhie and Chow.[17]

3. BLOCK-IMPLICIT SOLUTION OF NAVIER–STOKES EQUATIONS

3.1. Pressure equation

The idea of the fully coupled algorithm based on a non-staggered finite volume method is the introduction of a pressure equation. This equation is derived by inserting the momentum equations

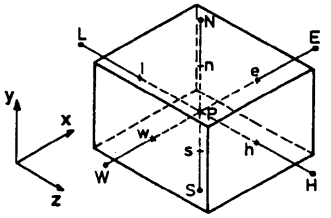

Figure 1. Control volume

into the continuity condition. The use of the discrete equations ensures the conservative form of the pressure equation. The decoupling of the velocity and the pressure field on a collocated grid is a well-known property of the finite volume discretization. A means to circumvent this problem is to evaluate the cell face velocities according to the momentum interpolation,[17] which is based on momentum balances on non- staggered and staggered grids:

$$U_{\mathrm{e}} = \overline{(u_{\mathrm{P}}, u_{\mathrm{E}})}_{\mathrm{e}} - \overline{\left[\frac{H^u_{\mathrm{P}}}{(a^u_{\mathrm{P}})_{\mathrm{P}}}, \frac{H^u_{\mathrm{E}}}{(a^u_{\mathrm{P}})_{\mathrm{E}}}\right]}_{\mathrm{e}} + \frac{A_{\mathrm{e}}}{\overline{[(a^u_{\mathrm{P}})_{\mathrm{P}}, (a^u_{\mathrm{P}})_{\mathrm{E}}]}_{\mathrm{e}}}(p_{\mathrm{P}} - p_{\mathrm{E}}). \tag{2}$$

H^j_i denotes the pressure source term of the jth Cartesian velocity component in control volume i. The overbar denotes the linear interpolation of the quantities at the cell face. All six cell face velocities are evaluated according to (2) and are inserted into the discretized continuity equation

$$(\rho UA)_{\mathrm{e}} - (\rho UA)_{\mathrm{w}} + (\rho VA)_{\mathrm{n}} - (\rho VA)_{\mathrm{s}} + (\rho WA)_{\mathrm{h}} - (\rho WA)_{\mathrm{l}} = 0. \tag{3}$$

After rearrangement the pressure equation in Cartesian or cylindrical co-ordinates yields

$$a^p_{\mathrm{P}} p_{\mathrm{P}} = \sum_{nb} a^p_{nb} p_{nb} = b^p_{\mathrm{P}}, \quad nb = \mathrm{EE, E, W, WW, NN, N, S, SS, HH, H, L, LL}, \tag{4}$$

with

$$\begin{aligned} b^p_{\mathrm{P}} = {} & (\rho A)_{\mathrm{w}} \overline{(u_{\mathrm{W}}, u_{\mathrm{P}})}_{\mathrm{w}} - (\rho A)_{\mathrm{e}} \overline{(u_{\mathrm{P}}, u_{\mathrm{E}})}_{\mathrm{e}} + (\rho A)_{\mathrm{s}} \overline{(v_{\mathrm{S}}, v_{\mathrm{P}})}_{\mathrm{s}} - (\rho A)_{\mathrm{n}} \overline{(v_{\mathrm{P}}, v_{\mathrm{N}})}_{\mathrm{n}} \\ & + (\rho A)_{\mathrm{l}} \overline{(w_{\mathrm{L}}, w_{\mathrm{P}})}_{\mathrm{l}} - (\rho A)_{\mathrm{h}} \overline{(w_{\mathrm{P}}, w_{\mathrm{H}})}_{\mathrm{h}}. \end{aligned} \tag{5}$$

The coefficients of (4) are given in Appendix II. The right-hand side of (4) is equivalent to the continuity equation. Therefore the pressure equation is a continuity equation modified by a fourth-order pressure term.

3.2. Solution of non-linear system

The non-linear coupled system of the three momentum equations and the pressure equation is solved by Newton's method in a block-implicit manner. In applying Newton's method, the linear system $\mathbf{J} \cdot \Delta\vec{X}^{(m+1)} = \vec{F}$, with $\Delta\vec{X}^{(m+1)} = [\Delta\vec{u}, \Delta\vec{v}, \Delta\vec{w}, \Delta\vec{p}]^{\mathrm{T}}$ and $\vec{F} = [\vec{f}^u, \vec{f}^v, \vec{f}^w, \vec{f}^p]^{\mathrm{T}}$, has to be solved. The right-hand side of the linear system is the vector of algebraic momentum and pressure equations

$$f^\phi = a^\phi_{\mathrm{P}} \phi_{\mathrm{P}} - \sum_{nb} a^\phi_{nb} \phi_{nb} - b^\phi_{\mathrm{P}}, \quad nb = \mathrm{E, W, N, S, H, L}, \quad \phi = u, v, w, \tag{6}$$

$$f^p = a^p_{\mathrm{P}} p_{\mathrm{P}} - \sum_{nb} a^p_{nb} p_{nb} - b^p_{\mathrm{P}}, \quad nb = \mathrm{EE, E, W, WW, NN, N, S, SS, HH, H, L, LL}. \tag{7}$$

The solution vector $\vec{X}$ is iteratively corrected according to $\vec{X}^{(m+1)} = \vec{X}^{(m)} + \beta_\phi \cdot \Delta\vec{X}^{(m+1)}$ until a certain convergence criterion is reached (see Section 4.4). The changes $\Delta\vec{X}$ can be damped by a factor β_ϕ between zero and one. The elements of the Jacobian matrix are derived in an analytical way. The Jacobian matrix is of the form

$$\mathbf{J} = \begin{bmatrix} \mathbf{J}^{u,u} & \mathbf{J}^{u,v} & \mathbf{J}^{u,w} & \mathbf{J}^{u,p} \\ \mathbf{J}^{v,u} & \mathbf{J}^{v,v} & \mathbf{J}^{v,w} & \mathbf{J}^{v,p} \\ \mathbf{J}^{w,u} & \mathbf{J}^{w,v} & \mathbf{J}^{w,w} & \mathbf{J}^{w,p} \\ \mathbf{J}^{p,u} & \mathbf{J}^{p,v} & \mathbf{J}^{p,w} & \mathbf{J}^{p,p} \end{bmatrix}. \tag{8}$$

One advantage of Newton's method is the separation of numerical and physical aspects. Therefore the elements of the Jacobian are modified such that the arising linear system can be efficiently solved by iterative methods. In this study the Jacobian is modified by the following two simplifications: the

coupling between the momentum equations is ignored and the bandwidth of the pressure matrix $\mathbf{J}^{p,p}$ is reduced from 13 to seven. This is achieved by neglecting the non-staggered pressure source terms of the momentum interpolation H (see equation (2)) in the Jacobian elements of the pressure equation. Furthermore, the elements of the Jacobian are calculated taking only the UPWIND scheme into account. The additional terms arising from the linearization of the second-order convection terms are neglected within the Jacobian. In other words, the MLU scheme of second-order accuracy is introduced via the well-known deferred correction approach.[18] The linear system $\tilde{\mathbf{J}} \cdot \Delta\vec{X} = \vec{F}$, with the Jacobian $\tilde{\mathbf{J}}$ shown in (9), can be efficiently solved by iterative methods and preserves the strong coupling between momentum and continuity equations:

$$\tilde{\mathbf{J}} = \begin{bmatrix} \tilde{\mathbf{J}}^{u,u} & 0 & 0 & \tilde{\mathbf{J}}^{u,p} \\ 0 & \tilde{\mathbf{J}}^{v,v} & 0 & \tilde{\mathbf{J}}^{v,p} \\ 0 & 0 & \tilde{\mathbf{J}}^{w,w} & \tilde{\mathbf{J}}^{w,p} \\ \tilde{\mathbf{J}}^{p,u} & \tilde{\mathbf{J}}^{p,v} & \tilde{\mathbf{J}}^{p,w} & \tilde{\mathbf{J}}^{p,p} \end{bmatrix}. \tag{9}$$

Other transport equations for the turbulent kinetic energy k, the dissipation rate ϵ, etc. are solved decoupled in a block-iterative manner.

3.3. Comparison of pressure correction and block-implicit algorithm

In this subsection the matrix structures of the pressure correction and the block-implicit algorithm are compared. Figure 2 shows a schematic illustration of both matrices. The numbers denote the non-zero elements within each row of a submatrix. Considering the pressure correction procedure, the submatrices are fully decoupled. In contrast, the three additional non-zero elements within the matrix of the block-implicit algorithm lead to the implicit coupling of the momentum and the pressure equation.

4. PARALLELIZATION AND SOLUTION OF LINEAR SYSTEM

4.1. Domain decomposition

With the new block-implicit method and the pressure correction procedure, two different algorithms for the solution of the incompressible Navier–Stokes equations are available. In this section the parallelization of both algorithms is described in detail. For the parallelization the domain decomposition method is applied (see Figure 3). Therefore the computational domain is split into a number of overlapping subdomains. Each of the processors manages the data and calculates the solution in one subdomain.

Within the iterative solution process, data are exchanged at the interior boundaries between the subdomains. One advantage of the domain decomposition approach is that only a few data have to be exchanged by local communication. On the other hand, the solutions within the subdomains are

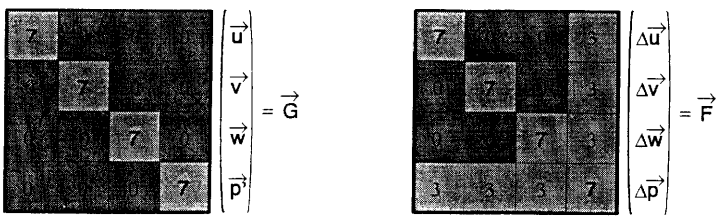

Figure 2. Schematic illustration of matrix structures: left, pressure correction procedure; right, block-implicit algorithm

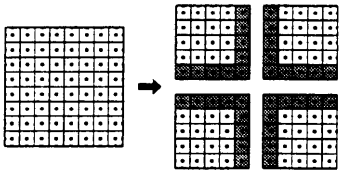

Figure 3. Domain decomposition

decoupled and the convergence characteristics of the parallel procedure may therefore deteriorate. A means for the stabilization of the convergence behaviour is to intensify the data exchange. This has been investigated in conjunction with the pressure correction algorithm. The different parallel solution procedures will be described in the subsection. With the development of the parallel version of the predescribed block-implicit algorithm the influence of the coupling between the Navier–Stokes equations on the performance of a parallel algorithm has been explored. An outline of the parallel block-implicit algorithm will be given in the next-but-one subsection.

4.2. Parallel block-iterative procedure

In applying the pressure correction algorithm and a block-iterative solution procedure in conjunction with the deferred correction approach, heptadiagonal matrices arise. Therefore linear algebraic systems $\mathbf{A}^{\phi} \cdot \vec{\phi} = \vec{b}$ have to be solved. The structure of the matrix $\mathbf{A}^{\phi}$ is given in Figure 4. The computational domain is decomposed once in each of the three co-ordinate directions. The broken lines denote the coefficients at the interior boundaries between the subdomains. The iterations performed to solve a linear system are called inner iterations. The complete sequence of solving the linear systems for all transport equations once is called an outer iteration. The three different strategies used in this study for the parallel solution of the block-iterative coupled linear systems will be described next.

The first type is the explicit coupling method after each outer iteration, called explicit outer iteration coupling (EOC). All data at the interior boundaries are exchanged after each outer iteration and each subsystem shown in Figure 4 is solved separately. The advantage of the EOC method is that

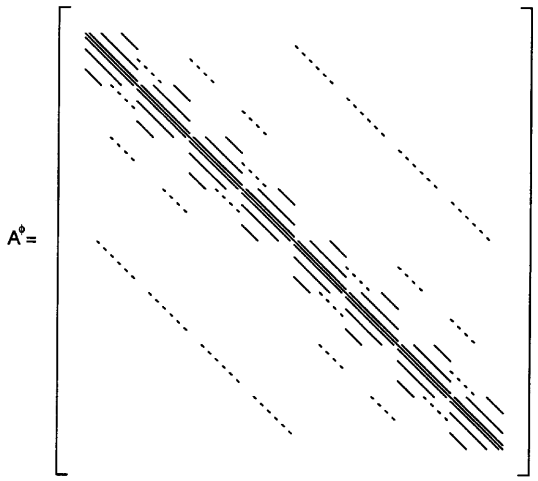

Figure 4. Structure of matrix $\mathbf{A}^{\phi}$ (eight subdomains)

relatively less time compared with the other two methods is needed for the data transfer, increasing the parallel efficiency of the parallel process. The disadvantage is the weak coupling between the subdomains, influencing convergence characteristics. In this study the incomplete lower–upper preconditioned conjugate gradient algorithm (ILU-CG)[19] has been used in conjunction with the EOC coupling method.

In applying the second coupling procedure, data transfer is performed within the solution algorithm of the system of algebraic equations after each inner iteration, called explicit inner iteration coupling (EIC). Thus a better coupling of the subdomains is achieved. For this coupling method the SIP algorithm is used.[20]

The third coupling method, called implicit inner iteration coupling (IIC), uses the largest amount of data exchange and achieves stronger coupling of the subdomains. For this procedure the ILU-CG algorithm of Noll and Wittig[19] is applied as the basic algorithm. Owing to its strong recursive nature, ILU preconditioning is not suitable for parallel computing. In the parallel version the ILU preconditioning is performed locally in each subdomain as a block-ILU decomposition and therefore in parallel. The coupling of the subsystems denoted by the broken lines in Figure 4 is neglected. In contrast, the CG acceleration step is performed globally taking into account all coefficients. Since this step consists of matrix–vector and vector–vector multiplications, most of the operations can be done in parallel. The final result of a vector–vector multiplication is evaluated by global communication, which is a limiting factor of the IIC coupling method.

Since EOC and IIC apply the same solution algorithm for the linear system of equations, the influence of the coupling method can be studied. In comparing EIC and IIC, the influence of different solution algorithms with approximately the same strong coupling procedure is stressed.

4.3. Parallel block-implicit procedure

For the parallel version of the block-implicit method the linear system of equations is solved by the ILU-preconditioned Bi-CGSTAB algorithm[21] with IIC coupling of the subdomains. The residuals are smoothed as proposed by Schönauer.[22] Several test calculations showed that an explicit coupling of the subdomains in conjunction with the block- implicit method is not suitable. A block-ILU decomposition of the Jacobian is performed in each subdomain in parallel. The structure of the Jacobian matrix $\tilde{\mathbf{J}}$ is shown in Figure 5 for the decomposition of the computational domain in two

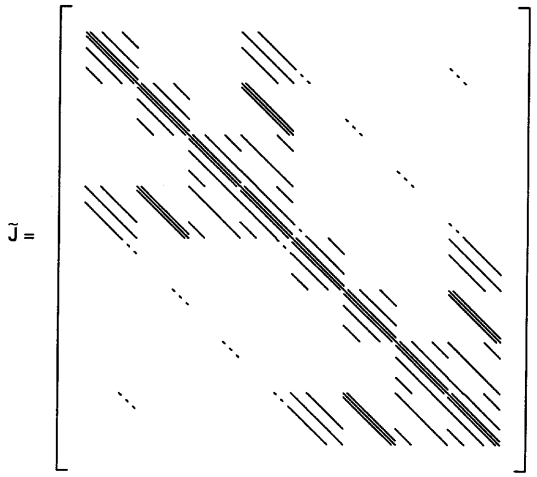

Figure 5. Structure of Jacobian $\tilde{\mathbf{J}}$ (two subdomains)

subdomains. The broken lines denote the coupling of the subsystems, which is neglected for the block-ILU decomposition of the matrix.

The solution steps within the Bi-CGSTAB algorithm are evaluated globally with the data of all subdomains:

$$
\begin{aligned}
&\vec{r}_0 = \vec{F}, \Delta\vec{X}_i^* = 0{\cdot}0, \rho_0 = \alpha = \omega_0 = 1{\cdot}0, \vec{v} = \vec{p} = 0{\cdot}0 \\
&\rho_i = (\vec{r}_0, \vec{r}_{i-1}^*) \\
&\beta = (\rho_i/\rho_{i-1})/(\alpha/\omega_{i-1}) \qquad \text{(global communication)} \\
&\vec{p}_i = \vec{r}_{i-1}^* + \beta(\vec{p}_{i-1} - \omega_{i-1}\vec{v}_{i-1}) \\
&\text{solve} \quad \mathbf{LU} \cdot \vec{y} = \vec{p}_i \\
&\text{exchange} \quad \vec{y} \qquad \text{(local communication)} \\
&\vec{v}_i = \tilde{\mathbf{J}}\vec{y} \\
&\alpha = \rho_i/(\vec{r}_0, \vec{v}_i) \qquad \text{(global communication)} \\
&\vec{s} = \vec{r}_i^* - \alpha\vec{v}_i \\
&\text{solve} \quad \mathbf{LU} \cdot \vec{z} = \vec{s} \qquad (10) \\
&\text{exchange} \quad \vec{z} \qquad \text{(local communication)} \\
&\vec{t} = \tilde{\mathbf{J}}\vec{z} \\
&\omega_i = (\mathbf{L}^{-1}\vec{t}, \mathbf{L}^{-1}\vec{s})/(\mathbf{L}^{-1}\vec{t}, \mathbf{L}^{-1}\vec{t}) \qquad \text{(global communication)} \\
&\Delta\vec{X}_i^* = \Delta\vec{X}_{i-1}^* + \alpha\vec{y} + \omega_i\vec{z} \\
&\vec{r}_i^* = \vec{s} - \omega_i\vec{t} \\
&\gamma = -\frac{\vec{r}_{i-1}^*(\vec{r}_{i-1} - \vec{r}_{i-1}^*)}{(\vec{r}_{i-1} - \vec{r}_{i-1}^*)(\vec{r}_{i-1}^* - \vec{r}_{i-1})} \qquad \text{(global communication)} \\
&\vec{r}_i = \gamma \cdot \vec{r}_{i-1} + (1-\gamma) \cdot \vec{r}_{i-1} \\
&\Delta\vec{X}_i^{(m+1)} = \gamma \cdot \Delta\vec{X}_{i-1} + (1-\gamma) \cdot \Delta\vec{X}_{i-1}^* \\
&\text{check} \quad \frac{\|\vec{r}_i\|_1}{r_{\text{Ref}}} \leqslant \epsilon_{\text{I}} \quad \text{or} \quad i \geqslant N_{\text{IM}} \qquad \text{(global communication)}
\end{aligned}
$$

4.4. *Convergence criteria*

In applying the pressure correction procedure, it is necessary to solve the system of algebraic equations until the residual of the inner interation falls below a specified value ϵ_{I}:

$$\frac{\|\vec{r}_i\|_1}{r_{\text{Ref}}} \leqslant \epsilon_{\text{I}}. \qquad (11)$$

This criterion is most important for the pressure correction equation in order to enforce the continuity condition. Approximately 10–20 inner iterations have to be carried out to satisfy (11). The solution of the other algebraic equations for the velocities, k and ϵ of the turbulence model, etc. requires only one to five inner iterations.

For the solution of the fully coupled system of the block-implicit algorithm a fixed number of inner iterations were performed. Ten iterations were made for laminar flows, whereas five iterations proved to be sufficient for turbulent flows.

A non-linear solution was considered to be converged when the normalized changes between two outer iterations were smaller than a specified value ϵ_A:

$$\frac{\|\Delta\vec{\phi}\|_\infty}{\phi_{\text{ref}}} \leqslant \epsilon_A. \tag{12}$$

The normalization factor ϕ_{ref} contains the relaxation parameter in order to account for the influence of relaxation on convergence.

4.5. *Efficiency analysis*

In order to evaluate the performance characteristics of the predescribed parallel methods, an efficiency analysis was made. Therefore the following relations were used:

$$\epsilon_t = \frac{T_1}{NP \cdot T_{NP}} = \frac{1}{1 + T_{NP}^{\text{com}}/T_{NP}^{\text{cal}}} \cdot \frac{a_1}{a_{NP}} = \epsilon_p \cdot \epsilon_n. \tag{13}$$

The total efficiency ϵ_t is subdivided into a parallel and a numerical part. The parallel efficiency is influenced by the ratio of the times requested for calculation, T_{NP}^{cal}, and communication, T_{NP}^{com}. In contrast, the numerical efficiency includes the ratio of the number of floating point operations per grid point required for the calculation of a converged solution with one and *NP* processors. In contrast with the work of Schreck and Peric,[23] the total and the parallel efficiency were evaluated after the times T_1, T_{NP} and T_{NP}^{com} were measured. The numerical efficiency is given by (13).

5. TEST CASES

In this section, results and efficiency data of the predescribed block- implicit parallel algorithm are presented and compared with the data of the parallel version of the SIMPLEC pressure correction algorithm. The parallel pressure correction algorithms in conjunction with the MLU scheme will be denoted by 'ptM' and '-EOC', '-EIC' or '-IIC' for the coupling of the subdomains. The parallel block-implicit method will be abbreviated by 'pmM-IIC'.

5.1. *Laminar flow over a backward-facing step*

The laminar flow over a backward-facing step served as a first test case for the parallel algorithms. The flow has been experimentally investigated by Armaly *et al.*[24] The Reynolds number based on the height of the inflow channel and the average main velocity $\bar{U}$ (0·533 m s^{-1}) at the step is 389 (see Figure 6). The computational domain extends 20·4 step heights *s* downstream of the expansion (78 × 38 (x/y) grid points). The predictions have been performed on the GCel 1024 Transputer Cluster with 1, 4/2, 4/4 and 8/4 (x/y) processors or subdomains respectively. The CPU speed and memory of one processor are 4·3 Mflops (peak performance) and 4 MB respectively. The single-processor calculations were done on a processor with 16 MB memory. The velocity profiles of the *x*-

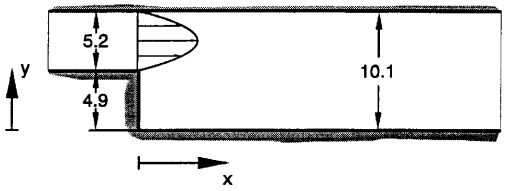

Figure 6. Backward-facing step[24]

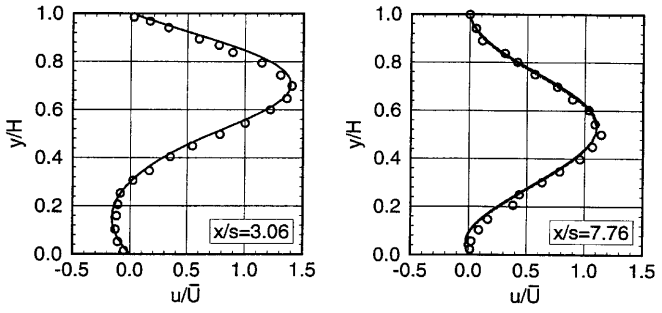

Figure 7. Velocity profiles (backward-facing step): ————, predicted; ○, experiment[24]

component predicted with the block-implicit algorithm are compared with the experimental data at two locations for all the processor configurations (see Figure 7). The velocity and the y-co-ordinate are divided by the average velocity at the expansion $\bar{U}$ and the channel height H respectively. The block-implicit algorithm gives high spatial accuracy independent of the number of processors and excellent agreement with the experimental data is achieved.

A detailed comparison of the times required to predict a converged solution is given in Figure 8. Except for the single-processor configuration, the block-implicit method requires approximately the same times as the pressure correction methods. Comparing the pressure correction procedures, the EIC coupling is characterized by the highest effort.

The parallel, numerical and total efficiencies versus the number of processors are plotted in Figure 9. Owing to the same ratio of times for communication and calculation, the parallel efficiencies of the block-implicit and the pressure correction algorithm with EOC coupling are approximately equal. It should be noted that the block-implicit algorithm requires the highest times for communication and calculation per outer iteration.

In contrast, the numerical efficiencies of the ptM-EOC and the pmM-ICC method differ. Owing to the stronger coupling of the momentum and the pressure equation, the block-implicit procedure gives the highest numerical efficiencies, whereas the ptM-EOC method yields the lowest values.

5.2. Mixing zone

As a second test case, the turbulent mixing of cold jets in a hot cross-flow was predicted using the parallel algorithms. This is a typical flow configuration for the mixing zone of a gas turbine combustor. The test section is shown in Figure 10 and experimental results have been published by us

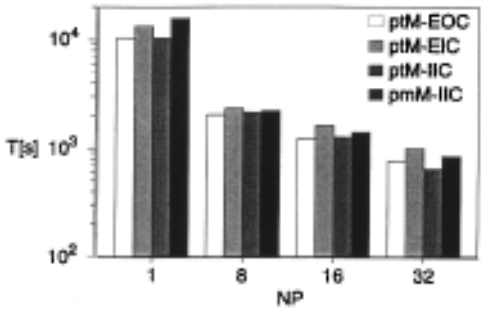

Figure 8. Comparison of times (backward-facing step)

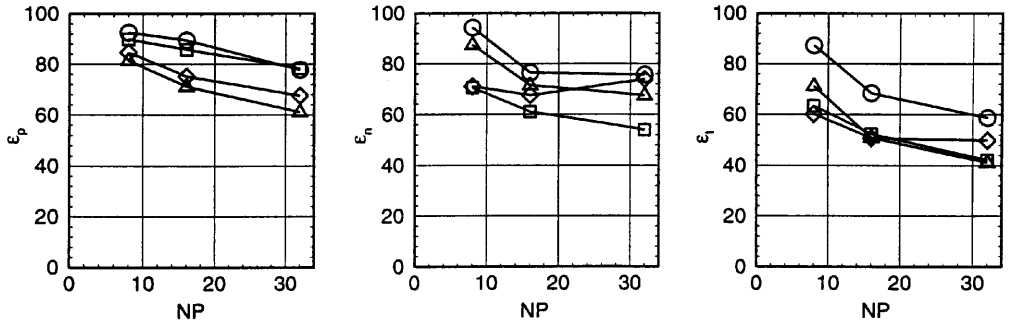

Figure 9. Parallel, numerical and total efficiencies (backward-facing step): □, ptM-EOC; △, ptM-EIC; ◇, ptM-IIC; ○, pmM-IIC

earlier.[25] At the top and bottom walls of the channel, five cold jets ($T_J = 308$ K) penetrate the mainstream ($\bar{T} = 460$ K). The jets enter the channel with velocities of 75·6 m s^{-1} (top) and 75·0 m s^{-1} (bottom), whereas the mainstream velocity $\bar{U}$ is 13·7 m s^{-1}. The height H of the channel is 100 mm and the diameter of the 20 mm spaced jets is 8 mm. Because of symmetry, the computational domain is located between the x–y plane at the centre of a hole and the subsequent x–y plane between two holes ($33 \times 30 \times 10$ ($x/y/z$) grid points).

The parallel computations were performed on the GCel 1024 Transputer Cluster with 1, 8/4/1, 8/8/1, 8/8/2 and 8/8/4 ($x/y/z$) processors. Therefore the grid was partitioned in two and three space directions. A comparison of the mainstream velocities and temperatures predicted with the ptM-EOC method at two locations within the channels is given in Figure 11 for all the processor configurations. Although a relatively coarse grid is applied, the numerical procedure is able to resolve the steep gradients at $x/H = 0{\cdot}4$ owing to the MLU scheme employed.

Figure 12 shows the total times of all methods. For small numbers of processors *NP* (32 and 64) the times of the different methods are similar. As the number of processors is doubled from 128 to 256, the times of the pressure correction algorithms increase owing to a deteriorated convergence behaviour. In contrast, the times of the block-implicit algorithm decrease for all processor configurations.

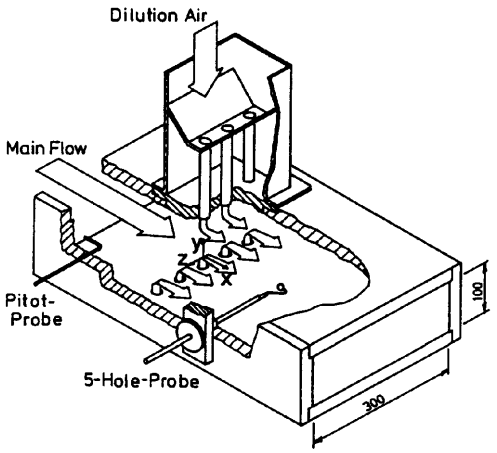

Figure 10. Test section: mixing zone[25]

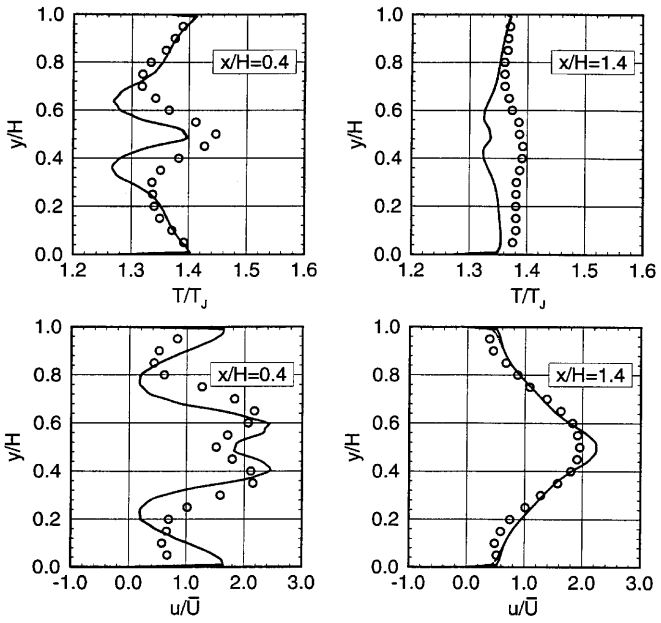

Figure 11. Velocity and temperature profiles (mixing zone): ————, predicted; ○, experiment[25]

Owing to the smaller amount of data transfer, the parallel efficiency of the pressure correction method is higher compared with the block-implicit procedure (see Figure 13). The numerical and total efficiencies versus the number of processors clearly indicate the stable convergence behaviour of the block-implicit algorithm. Moreover, the numerical efficiency of the pressure correction algorithm drops to 22 per cent for 256 processors.

The total memory per processor required for each solution algorithm and processor configuration is given in Table I. For the single-grid configuration the block-implicit algorithm requires about twice as much memory as the pressure correction algorithm. As the number of grid points per processor decreases, the differences in total memory between the two algorithms become smaller. This is due to the grid-node-independent part of the memory required to store the boundary conditions, etc. The data displayed in Table I demonstrate that memory is not a limiting factor of the block-implicit algorithm, especially on parallel computers. The data of ptm-IIC are not shown, because ptm-EOC and ptm-IIC have nearly the same memory requirements.

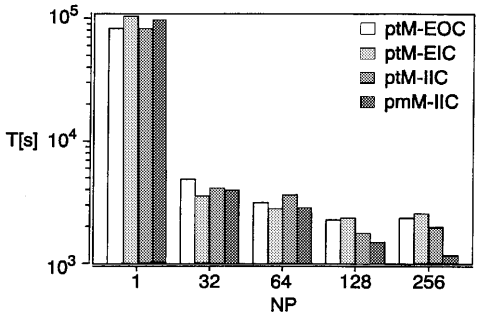

Figure 12. Comparison of times (mixing zone)

efficiency of the pressure correction algorithm drops down to 22 % for 256 processors.

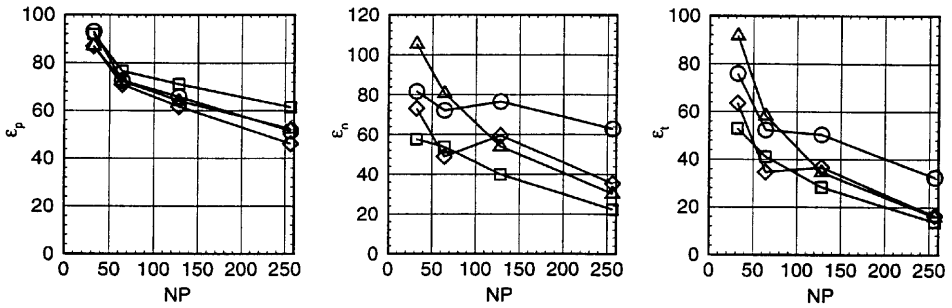

Figure 13. Parallel, numerical and total efficiencies (mixing zone): □, ptM-EOC; △, ptM-EIC; △, ptM-IIC; ○, pmM- IIC

Table I. Total memory per processor (mixing zone)

		$NP_x/NP_y/NP_z$				
		1/1/1	8/4/1	8/8/1	8/8/2	8/8/4
Total memory (MByte)	ptM-EOC	7·64	0·94	0·74	0·65	0·58
	ptM-EIC	7·52	0·93	0·73	0·64	0·58
	pmM-IIC	13·86	1·38	1·00	0·84	0·75

Finally, total times for three different grids with $33 \times 30 \times 10$, $64 \times 56 \times 18$ and $126 \times 108 \times 34$ grid points are shown in Figure 14. Comparing with subsequent grids, the number of points is doubled in each direction. This is a typical grid sequence for a grid dependence study. The times for the single-processor configuration of the two largest grids are estimated and compared with the times required by eight and 64 processors. All the data shown in Figure 14 were measured on the GC/PowerPlus parallel system. Each processor has a CPU speed of 80 Mflops (peak performance) and 32 MB memory. With 64 processors the overall time could be reduced to 11·6 h, whereas with only one processor 20·6 days would be required. These data show that grids with half-a-million points can be treated with the block-implicit algorithm.

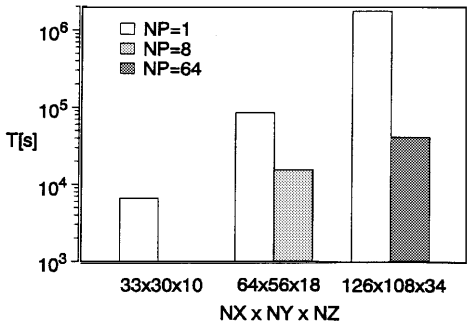

Figure 14. Times for pmM-IIC algorithm (mixing zone)

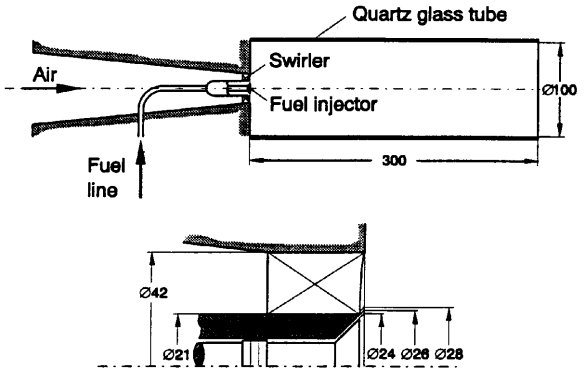

Figure 15. Swirl-stabilized combustor[26]

5.3. *Swirl-stabilized combustor*

In order to demonstrate their applicability to complex fluid dynamics, the flow in a propane-fuelled swirl-stabilized combustor was predicted with the parallel algorithms. The experimental set-up is shown in Figure 15.[26] Air enters the tubular combustor with diameter D through a swirler with an angle of 45°. Gaseous propane fuel is introduced through the centre of the swirler via an annular slot as a conical jet with a cone angle of 90°. The air and propane temperatures are 313 and 294 K respectively and the corresponding flow rates are 30·4 and 1·284 g s^{-1}.

The calculations of the flow within the combustor were performed on the CG/PowerPlus parallel system. The grid ($3 \times 52 \times 54$ ($\varphi/r/z$) grid points) was partitioned in the radial and axial directions using 1, 1/2/2, 1/2/4, 1/2/8 and 1/4/8 ($\varphi/r/z$) processors. The predicted temperature distribution is shown in Figure 16. The maximum temperatures of about 2200 K can be found at the outlet of the chamber. Near the fuel nozzle the temperatures are relatively low owing to fuel-rich conditions. A large number of experimental and numerical data sets have been compared and analysed and in general good qualitative agreement could be accomplished.[27] The flow in the combustor was also investigated by us earlier using detailed PDF models for combustion.[28]

The total times required for the calculation of the flow are shown in Figure 17. The pressure correction algorithm with EOC coupling consumes the lowest times for all processor configurations except for the single-processor calculation. This is due to the fact that this parallel method satisfies

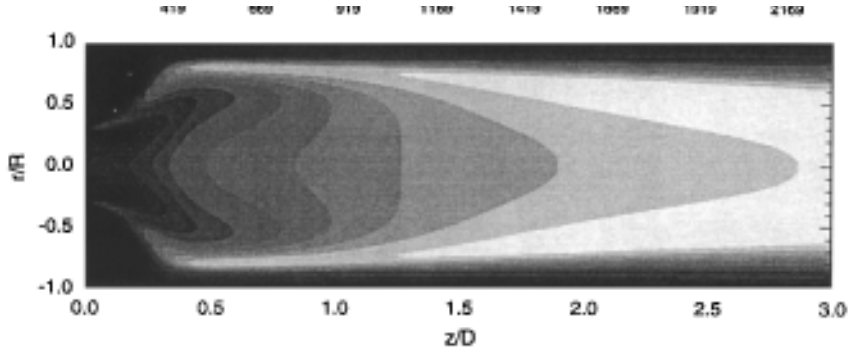

Figure 16. Temperature distribution (swirl-stabilized combustor)

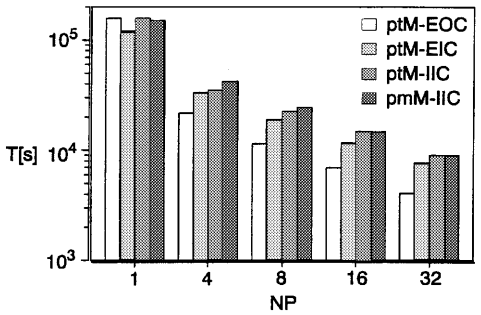

Figure 17. Comparison of times (swirl-stabilized combustor)

the convergence criterion after half the number of iterations compared with the single-processor calculation. This behaviour can be explained by noting that the EOC coupling is equivalent to a relaxation of the variables at the interior boundaries. Therefore the density is also relaxed, which has an important impact on the convergence behaviour of flows with heat release.

The parallel efficiencies of the pressure correction methods are at the same level (see Figure 18). As the number of processors is increased to 16 and 32, the parallel efficiency of the block-implicit method decreases faster compared with the pressure correction procedures. The processors of the GC/PowerPlus system are about 20 times faster than those of the Transputer Cluster. On the other hand, the network of the GC/PowerPlus system is slow compared with the Transputer Cluster. Therefore the enhanced effort for calculation and communication of the block-implicit method leads to low parallel efficiencies.

Since the ptM-EOC algorithm requires only half the number of outward iterations compared with the single-processor calculation, its numerical efficiencies are in the range of 200 per cent. The numerical efficiencies of the other algorithms are slightly dependent on the number of processors used. However, the block-implicit method is again insensitive to relaxation parameters, even for the prediction of complex flows with heat release.

7. CONCLUSIONS

A comparison of a block-implicit method and the SIMPLEC pressure correction procedure on parallel computing systems has been presented. The block-implicit method is characterized by strong coupling between the momentum and the pressure equation. This leads to stable convergence

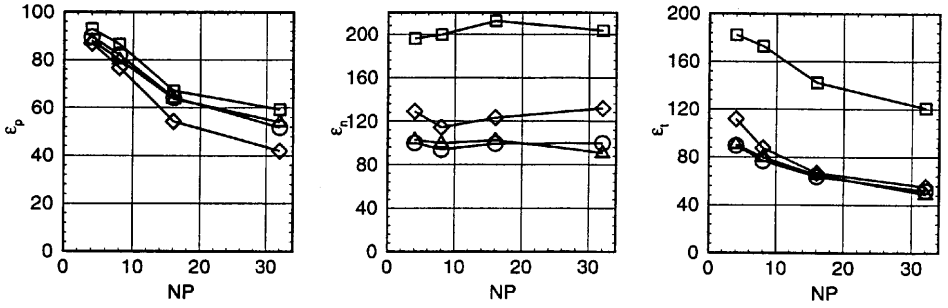

Figure 18. Parallel, numerical and total efficiencies (swirl- stabilized combustor): □, ptM-EOC; △, ptM-EIC; ◇, ptM-IIC; ○, pmM-IIC

behaviour of the presented method. In contrast with the pressure correction procedure, the relaxation or damping factors do not have to be adjusted for different processor configurations. The laminar flow over a backward-facing step and the flow within a mixing zone have been carried out with a single set of parameters ($\beta_p = 1{\cdot}0$, $\beta_c = 0{\cdot}6$). The flow within the swirl-stabilized combustor has been predicted with $\beta_p = 1{\cdot}0$ and $\beta_c = 1{\cdot}0$.

As far as the coupling of the solutions within the subdomains is concerned, the block-implicit method works well only with an implicit coupled Bi-CGSTAB solver. On the other hand, the linear systems of the pressure correction procedure can be solved with an explicit or an implicit coupling of the subdomains. A recommendation for the best coupling method in conjunction with the pressure correction procedure cannot be given, since the performances of the different methods depend on the type of flow. Taking all experiences into account, the parallel block-implicit method gives the best overall performance and is therefore strongly recommended.

ACKNOWLEDGEMENTS

This work was supported by a grant from SFB 167 (High Intensity Combustors) of the Deutsche Forschungsgemeinschaft.

APPENDIX I: NOMENCLATURE

a	coefficient of algebraic equation*
a	number of floating point operations per grid point
A	side area of control volume (m^2)
b	right-hand-side of the algebraic equation*
$\vec{c}$	velocity vector (m s^{-1})
$\vec{F}$	vector of algebraic equations*
H	pressure source term (N)
$\mathbf{J}$	Jacobian matrix*
NP	number of processors
p	static pressure (N m^{-2})
S	source term*
T	total CPU time (s)
u, v, w	velocity components (m s^{-1})
U, V, W	velocity components at cell face (m s^{-1})
x, y, z	spatial co-ordinates (m)
$\vec{X}$	solution vector*

Greek letters

β	damping factor
Γ	diffusion coefficient*
ϵ	turbulence energy dissipation rate (m^2 s^{-3})
ϵ	efficiency
ϵ	convergence criterion
ρ	density (kg m^{-3})
ϕ	flow quantity*

* Equation-dependent.

Superscripts

cal	calculation
com	communication
u, *v*, *w*	momentum equation
p	pressure equation
,ϕ	differentiation with respect to ϕ

Subscripts

A	outer iteration
c	velocities
E, W, N, S, H, L	six nodes adjacent to P
I	inner iteration
n	numerical
NP	number of processors
p	parallel
p	pressure
P	grid point
t	total
1	one processor

APPENDIX II: COEFFICIENTS OF PRESSURE EQUATION

$$
\begin{aligned}
a_{\mathrm{P}}^{p} &= (B_{\mathrm{w,P}} - B_{\mathrm{e,P}}) \cdot (\xi_{x,\mathrm{w}} + \xi_{x,\mathrm{e}} - 1) - B_{\mathrm{e,E}} \cdot (1 - \xi_{x,\mathrm{e}}) - B_{\mathrm{w,W}} \cdot \xi_{x,\mathrm{W}} \\
&\quad + (B_{\mathrm{s,P}} - B_{\mathrm{n,P}}) \cdot (\xi_{y,\mathrm{s}} + \xi_{y,\mathrm{n}} - 1) - B_{\mathrm{n,N}} \cdot (1 - \xi_{y,\mathrm{n}}) - B_{\mathrm{s,S}} \cdot \xi_{y,\mathrm{S}} \\
&\quad + (B_{\mathrm{l,P}} - B_{\mathrm{h,P}}) \cdot (\xi_{x,\mathrm{l}} + \xi_{z,\mathrm{h}} - 1) - B_{\mathrm{h,H}} \cdot (1 - \xi_{z,\mathrm{h}}) - B_{\mathrm{l,L}} \cdot \xi_{z,\mathrm{L}} \\
&\quad + \frac{(\rho A)_{\mathrm{e}}}{\overline{[(a_{\mathrm{P}}^{u})_{\mathrm{P}}, (a_{\mathrm{P}}^{u})_{\mathrm{E}}]}_{\mathrm{e}}} + \frac{(\rho A)_{\mathrm{w}}}{\overline{[(a_{\mathrm{P}}^{u})_{\mathrm{W}}, (a_{\mathrm{P}}^{u})_{\mathrm{P}}]}_{\mathrm{w}}} + \frac{(\rho A)_{\mathrm{n}}}{\overline{[(a_{\mathrm{P}}^{v})_{\mathrm{P}}, (a_{\mathrm{P}}^{v})_{\mathrm{N}}]}_{\mathrm{n}}} \\
&\quad + \frac{(\rho A)_{\mathrm{s}}}{\overline{[(a_{\mathrm{P}}^{v})_{\mathrm{S}}, (a_{\mathrm{P}}^{v})_{\mathrm{P}}]}_{\mathrm{s}}} + \frac{(\rho A)_{\mathrm{h}}}{\overline{[(a_{\mathrm{P}}^{w})_{\mathrm{P}}, (a_{\mathrm{P}}^{w})_{\mathrm{H}}]}_{\mathrm{h}}} + \frac{(\rho A)_{\mathrm{l}}}{\overline{[(a_{\mathrm{P}}^{w})_{\mathrm{L}}, (a_{\mathrm{P}}^{w})_{\mathrm{P}}]}_{\mathrm{l}}}, \\
a_{\mathrm{WW}}^{p} &= -B_{\mathrm{w,W}} \cdot (1 - \xi_{w,\mathrm{ww}}), \\
a_{\mathrm{W}}^{p} &= -B_{\mathrm{w,W}} \cdot (\xi_{x,\mathrm{ww}} + \xi_{x,\mathrm{w}} - 1) + (B_{\mathrm{e,P}} - B_{\mathrm{w,P}}) \cdot (1 - \xi_{x,\mathrm{w}}) + \frac{(\rho A)_{\mathrm{w}}}{\overline{[(a_{\mathrm{P}}^{u})_{\mathrm{W}}, (a_{\mathrm{P}}^{u})_{\mathrm{P}}]}_{\mathrm{w}}}, \\
a_{\mathrm{EE}}^{p} &= -B_{\mathrm{e,E}} \cdot \xi_{x,\mathrm{ee}}, \\
a_{\mathrm{E}}^{p} &= (B_{\mathrm{w,P}} - B_{\mathrm{e,P}}) \cdot \xi_{x,\mathrm{e}} + B_{\mathrm{e,E}} \cdot (\xi_{x,\mathrm{e}} + \xi_{x,\mathrm{ee}} - 1) + \frac{(\rho A)_{\mathrm{e}}}{\overline{[(a_{\mathrm{P}}^{u})_{\mathrm{P}}, (a_{\mathrm{P}}^{u})_{\mathrm{E}}]}_{\mathrm{e}}}, \\
a_{\mathrm{SS}}^{p} &= -B_{\mathrm{s,S}} \cdot (1 - \xi_{y,\mathrm{ss}}),
\end{aligned}
$$

$$a_{\mathrm{S}}^{p} = -B_{\mathrm{s,S}} \cdot (\xi_{y,\mathrm{ss}} + \xi_{y,\mathrm{s}} - 1) + (B_{\mathrm{n,P}} - B_{\mathrm{s,P}}) \cdot (1 - \xi_{y,\mathrm{s}}) + \frac{(\rho A)_{\mathrm{s}}}{\overline{[(a_{\mathrm{P}}^{v})_{\mathrm{S}}, (a_{\mathrm{P}}^{v})_{\mathrm{P}}]}_{\mathrm{s}}},$$

$$a_{\mathrm{NN}}^{p} = -B_{\mathrm{n,N}} \cdot \xi_{y,\mathrm{nn}},$$

$$a_{\mathrm{N}}^{p} = (B_{\mathrm{s,P}} - B_{\mathrm{n,P}}) \cdot \xi_{y,\mathrm{n}} + B_{\mathrm{n,N}} \cdot (\xi_{y,\mathrm{n}} + \xi_{y,\mathrm{nn}} - 1) + \frac{(\rho A)_{\mathrm{n}}}{\overline{[(a_{\mathrm{P}}^{v})_{\mathrm{P}}, (a_{\mathrm{P}}^{v})_{\mathrm{N}}]}_{\mathrm{n}}},$$

$$a_{\mathrm{LL}}^{p} = -B_{\mathrm{l,L}} \cdot (1 - \xi_{x,\mathrm{ll}}),$$

$$a_{\mathrm{L}}^{p} = -B_{\mathrm{l,L}} \cdot (\xi_{x,\mathrm{ll}} + \xi_{x,\mathrm{l}} - 1) + (B_{\mathrm{h,P}} - B_{\mathrm{l,P}}) \cdot (1 - \xi_{x,\mathrm{l}}) + \frac{(\rho A)_{\mathrm{l}}}{\overline{[(a_{\mathrm{P}}^{w})_{\mathrm{L}}, (a_{\mathrm{P}}^{w})_{\mathrm{P}}]}_{\mathrm{l}}},$$

$$a_{\mathrm{HH}}^{p} = -B_{\mathrm{h,H}} \cdot \xi_{x,\mathrm{hh}},$$

$$a_{\mathrm{H}}^{p} = (B_{\mathrm{l,P}} - B_{\mathrm{h,P}}) \cdot \xi_{z,\mathrm{h}} + B_{\mathrm{h,H}} \cdot (\xi_{z,\mathrm{h}} + \xi_{z,\mathrm{hh}} - 1) + \frac{(\rho A)_{\mathrm{h}}}{\overline{[(a_{\mathrm{P}}^{w})_{\mathrm{P}}, (a_{\mathrm{P}}^{w})_{\mathrm{H}}]}_{\mathrm{h}}},$$

$$B_{\mathrm{e,P}} = (\rho A)_{\mathrm{e}} \cdot (1 - \xi_{x,\mathrm{e}}) \cdot \frac{A_{\mathrm{P}}}{(a_{\mathrm{P}}^{u})_{\mathrm{P}}}, \qquad B_{\mathrm{e,E}} = (\rho A)_{\mathrm{e}} \cdot \xi_{x,\mathrm{e}} \cdot \frac{A_{\mathrm{E}}}{(a_{\mathrm{P}}^{u})_{\mathrm{E}}},$$

$$B_{\mathrm{w,W}} = (\rho A)_{\mathrm{w}} \cdot (1 - \xi_{x,\mathrm{w}}) \cdot \frac{A_{\mathrm{W}}}{(a_{\mathrm{P}}^{u})_{\mathrm{W}}}, \qquad B_{\mathrm{w,P}} = (\rho A)_{\mathrm{w}} \cdot \xi_{x,\mathrm{w}} \cdot \frac{A_{\mathrm{P}}}{(a_{\mathrm{P}}^{u})_{\mathrm{P}}},$$

$$B_{\mathrm{n,P}} = (\rho A)_{\mathrm{n}} \cdot (1 - \xi_{y,\mathrm{n}}) \cdot \frac{A_{\mathrm{P}}}{(a_{\mathrm{P}}^{v})_{\mathrm{P}}}, \qquad B_{\mathrm{n,N}} = (\rho A)_{\mathrm{n}} \cdot \xi_{y,\mathrm{n}} \cdot \frac{A_{\mathrm{N}}}{(a_{\mathrm{P}}^{v})_{\mathrm{N}}},$$

$$B_{\mathrm{s,S}} = (\rho A)_{\mathrm{s}} \cdot (1 - \xi_{y,\mathrm{s}}) \cdot \frac{A_{\mathrm{S}}}{(a_{\mathrm{P}}^{v})_{\mathrm{S}}}, \qquad B_{\mathrm{s,P}} = (\rho A)_{\mathrm{s}} \cdot \xi_{y,\mathrm{s}} \cdot \frac{A_{\mathrm{P}}}{(a_{\mathrm{P}}^{v})_{\mathrm{P}}},$$

$$B_{\mathrm{h,P}} = (\rho A)_{\mathrm{h}} \cdot (1 - \xi_{z,\mathrm{h}}) \cdot \frac{A_{\mathrm{P}}}{(a_{\mathrm{P}}^{w})_{\mathrm{P}}}, \qquad B_{\mathrm{h,H}} = (\rho A)_{\mathrm{h}} \cdot \xi_{z,\mathrm{h}} \cdot \frac{A_{\mathrm{H}}}{(a_{\mathrm{P}}^{w})_{\mathrm{H}}},$$

$$B_{\mathrm{l,L}} = (\rho A)_{\mathrm{l}} \cdot (1 - \xi_{z,\mathrm{l}}) \cdot \frac{A_{\mathrm{L}}}{(a_{\mathrm{P}}^{w})_{\mathrm{L}}}, \qquad B_{\mathrm{l,P}} = (\rho A)_{\mathrm{l}} \cdot \xi_{z,\mathrm{l}} \cdot \frac{A_{\mathrm{P}}}{(a_{\mathrm{P}}^{w})_{\mathrm{P}}},$$

$$\xi_{x,\mathrm{e}} = \frac{x_{\mathrm{e}} - x_{\mathrm{P}}}{x_{\mathrm{E}} - x_{\mathrm{P}}}, \qquad \xi_{x,\mathrm{w}} = \frac{x_{\mathrm{w}} - x_{\mathrm{P}}}{x_{\mathrm{W}} - x_{\mathrm{P}}}, \qquad \xi_{y,\mathrm{n}} = \frac{y_{\mathrm{n}} - y_{\mathrm{P}}}{y_{\mathrm{N}} - y_{\mathrm{P}}},$$

$$\xi_{y,\mathrm{s}} = \frac{y_{\mathrm{s}} - y_{\mathrm{P}}}{y_{\mathrm{S}} - y_{\mathrm{P}}}, \qquad \xi_{z,\mathrm{h}} = \frac{z_{\mathrm{h}} - z_{\mathrm{P}}}{z_{\mathrm{H}} - z_{\mathrm{P}}}, \qquad \xi_{z,\mathrm{l}} = \frac{z_{\mathrm{l}} - z_{\mathrm{P}}}{z_{\mathrm{L}} - z_{\mathrm{P}}}.$$

REFERENCES

1. S. Patankar, *Numerical Heat Transfer and Fluid Flow*, Hemisphere, Washington, DC, 1980.
2. J. P. Van Doormal and G. D. Raithby, 'Enhancement of the SIMPLE method for predicting incompressible fluid flows', *Numer. Heat Transfer*, **7**, 147–163 (1984).
3. J. P. Van Doormal and G. D. Raithby, 'An evaluation of the segregated approach for predicting incompressible fluid flows', *ASME Paper 85-HT-9*, 1985.
4. R. I. Issa, 'Solution of the implicitly discretized fluid flow equations by operator-splitting', *J. Comput. Phys.*, **62**, 40– 65 (1985).
5. A. J. Chorin, 'A numerical method for solving incompressible viscous flow problems', *J. Comput. Phys.*, **2**, 12–26 (1967).
6. S. P. Vanka, 'A calculation procedure for three-dimensional steady recirculating flows using multigrid methods', *Comput. Methods Appl. Mech. Eng.*, **55**, 321–338 (1986).
7. P. F. Galpin, J. P. Van Doormal and G. D. Raithby, 'Solution of the incompressible mass and momentum equations by application of a coupled equations line solver', *Int. j. numer. methods fluids*, **5**, 615–625 (1985).
8. S. P. Vanka, 'Block-implicit calculation of steady turbulent recirculating flows', *Int. J. Heat Mass Transfer*, **28**, 2093–2103 (1985).
9. S. S. Clift and P. A. Forsyth, 'Linear and non-linear iterative methods for the incompressible Navier–Stokes equations', *Int. j. numer. methods fluids*, **18**, 229–256 (1994).

10. J. Simoneau and A. Pollard, 'Finite volume methods for laminar and turbulent flows using a penalty function approach', *Int. j. numer. methods fluids*, **18**, 733–746 (1994).
11. M. E. Braaten, 'Solution of viscous fluid flows on a distributed memory concurrent computer', *Int. j. numer. methods fluids*, **10**, 889–905 (1990).
12. M. E. Braaten, 'Development of a parallel computational fluid dynamics algorithm on a hypercube computer', *Int. j. numer. methods fluids*, **12**, 947–963 (1991).
13. M. Peric, M. Schäfer and E. Schreck, 'Computation of fluid flow with a parallel multi-grid solver', *Proc. Conf. on Parallel Computational Fluid Dynamics '91*, Edited by K. G. Reinsch *et al*., Elsevier Science Publishers B.V., pp. 297–312, 1992.
14. B. E. Launder and D. B. Spalding, 'The numerical computation of turbulent flows', *Comput. Methods Appl. Mech. Eng.*, **3**, (1974).
15. B. F. Magnussen and B. H. Hjertager, 'On mathematical modeling of turbulent combustion with special emphasis on soot formation and combustion', *Proc. 16th Int. Symp. on Combustion*, Pittsburgh, 1976, pp. 719–729.
16. B. Noll, 'Evaluation of a bounded high-resolution scheme for combustor flow computations', *AIAA J.*, **30**, 64–69 (1992).
17. C. M. Rhie and W. L. Chow, 'Numerical study of the turbulent flow past an airfoil with trailing edge separation', *AIAA J.*, **21**, 1525–1532 (1983).
18. P. K. Khosla and S. G. Rubin, 'A diagonally dominant second-order accurate implicit scheme', *Comput. Fluids*, **2**, 207–209 (1974).
19. B. Noll and S. Wittig, 'Generalized conjugate gradient method for the efficient solution of three-dimensional fluid flow problems', *Numer. Heat Transfer B*, **20**, 207–221 (1991).
20. H. L. Stone, 'Iterative solution of implicit approximations of multidimensional partial differential equations', *SIAM J. Numer. Anal.*, **5**, 530–558 (1968).
21. H. A. Van der Vorst, 'Bi-CGSTAB: a fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems', *SIAM J. Sci. Stat. Comput.*, **13**, 631–644 (1992).
22. W. Schönauer, *Scientific Computing on Vector Computers*, North-Holland, Amsterdam, 1987.
23. E. Schreck and M. Peric, 'Computation of fluid flow with a parallel multigrid solver', *Int. j. numer. methods fluids*, **16**, 303–327 (1993).
24. B. F. Armaly, F. Durst, J. C. F. Pereira and B. Schönung, 'Experimental and theoretical investigation of backward-facing step flow', *J. Fluid Mech.*, **127**, 473–496 (1983).
25. S. Wittig, O. Elbahar and B. Noll, 'Temperature profile development in turbulent mixing of coolant jets with a confined hot crossflow', *ASME J. Eng. Gas Turb. Power*, **106**, 193–197 (1984).
26. J. Wilhelmi, 'Axisymmetric swirl stabilized combustion', *Ph.D. Thesis*, University of London, 1984.
27. M. Kurreck, 'Entwicklung grundlegender numerischer Methoden zur Berechnung gasturbinentypischer Strömungen mit Parallelrechnern', *Dissertation*, Institut für Thermische Strömungsmaschinen, Universität Karlsruhe (TH), 1995.
28. M. Scheurlen, 'Über den Einsatz von Monte-Carlo-Verfahren zur Berechnung von Wahrscheinlichkeitsdichtefunktionen in reagierenden Brennkammerströmungen', *Dissertation*, Institut für Thermische Strömungsmachinen, Universität Karlsruhe (TH), 1992.

