JOURNAL OF COMPUTATIONAL PHYSICS 116, 135–142 (1995)

A Fast Schur Complement Method for the Spectral Element Discretization of the Incompressible Navier–Stokes Equations

W. COUZY AND M. O. DEVILLE*

Applied Mechanics, Catholic University of Louvain, Louvain-la-Neuve, Belgium

Received June 7, 1993; revised April 4, 1994

The weak formulation of the incompressible Navier–Stokes equations in three space dimensions is discretized with spectral element approximations and Gauss–Lobatto–Legendre quadratures. The Uzawa algorithm is applied to decouple the velocities from the pressure. The equation that results for the pressure is solved by an iterative method. Within each pressure iteration, a Helmholtz operator has to be inverted. This can efficiently be done by separating the equations for the interior nodes from the equations at the interfaces, according to the Schur method. Fast diagonalization techniques are applied to the interior variables of the spectral elements. Several ways to deal with the resulting interface problem are discussed. Finally, a comparison is made with a more classical method.

1. INTRODUCTION

In the past years, the spectral element discretization of the 3D Navier–Stokes equations has received considerable attention [6, 8, 11, 13, 18]. The advantages of this method are numerous. The high degrees of the approximating polynomials combined with high-order quadrature rules yield accurate solutions. In comparison with more classical discretization methods, a low number of degrees of freedom is needed for a prescribed level of accuracy. The clustering of the gridpoints close to the boundary, which is typical for many spectral methods, makes the method attractive for flows dominated by boundary-layer dynamics. The decomposition of the domain into several subdomains (the spectral elements) ensures geometrical flexibility and a natural implementation on parallel computers [6, 8].

There are many ways to uncouple the velocities from the pressure. Karniadakis *et al.* [11] proposed a high-order splitting method, where the pressure is computed by a Poisson equation with compatible boundary conditions. Another way of dealing with this problem is the Uzawa technique [1] which is in fact a Gaussian elimination method by block. An advantage of this approach is that the resulting system, which consists of four positive (semi-) definite symmetric systems (one for the pressure and three for the velocities), is equivalent to the original coupled set of equations. Hence, the system is determined by velocity boundary conditions only and no additional conditions for the pressure are needed. Usually, the four systems are solved by the preconditioned conjugate gradient method (PCGM), an efficient iterative method for symmetric systems of equations. One of the attractive properties of the PCGM is that the matrix system, which would take $O(K_e N^6)$ memory positions, is never built up explicitly (K_e corresponds to the number of subdomains and N is the polynomial degree in one space dimension). Moreover, tensor producs reduce the bulk of the work, which consists in the computation of matrix-vector products, to $O(K_e N^4)$. A disadvantage of the Uzawa method, however, is the high cost required to compute the pressure. Since the pressure matrix contains the inverse of a Helmholtz operator, a classical implementation requires two *nested* PCGMs, resulting in large computation times. One way to deal with this problem is operator splitting [15]. This method seems to work very well in practice, although until now the consistency has not been proven for increasing order of the time scheme.

This paper deals with another approach, first proposed by Patera [14], that reduces rigorously the time to invert the Helmholtz operator and, hence, the time to compute the pressure. To this end, the Schur complement method is used to separate the Helmholtz equations at the interior nodes of each element from those at the inter-element interfaces. This leads to a set of independent subproblems which is, in general, easier to solve than the original global problem: Iterative methods tend to converge faster due to the locally reduced number of variables and the absence of inter-element coupling. Direct methods can also be considered to invert the local Helmholtz operators. In this paper, we will restrict ourselves to geometries consisting of non-deformed spectral elements. In this case, a fast direct method [12] based on tensor products of the eigenvalue/eigenvector decomposition of the one-dimensional operators is applied to the interior nodes. This direct method, which we will often refer to as the diagonalization method, is very fast; the inverse is computed at the price of two PCGM iterations. It is not surprising that the fast diagonalization method (FDM) is often used in the context of spectral methods (see, for example,

* Current address: Swiss Federal Institute of Technology, Lausanne, Switzerland.

0021-9991/95 $6.00

[14, 17]), where iterative methods tend to converge slowly due to a large value of N and ill-conditioned matrices. The dimension of the corresponding Schur matrix is less by one than the dimension of the original system, since it involves the interface variables only. Therefore, we might consider classical direct methods as well as iterative methods. In the latter case, the conjugate gradient method can be efficiently preconditioned by block Jacobi. Moreover, an impressive improvement on vector computers can be obtained when the Schur matrix is constructed explicitly. In this way, we avoid the evaluation of tensor products, which is inefficient in terms of vectorization. Independently of the solution method for the Schur matrix, we found that the new algorithm is an order of magnitude faster than the classical one.

The outline of this paper is as follows. First, in Section 2, we briefly present the spectral discretization of the Navier–Stokes equations. In Section 3 we discuss the FDM and in Section 4 we treat the Schur method for a particular problem. Section 5 will deal with two test problems and comment on the parallelization of the method.

2. DERIVATION OF THE DISCRETE EQUATIONS

The 3D incompressible Navier–Stokes equations are discretized by the spectral element method. For more details about the material of this section, we refer the reader to the article of Maday and Patera [13] and to the lecture series by Rønquist [18]. The Navier–Stokes problem is formulated as follows: Find velocities $\mathbf{u}$ and pressure p in a domain $\Omega \subset \mathcal{R}^3$ such that

$$\frac{\partial \mathbf{u}}{\partial t} - \mathrm{Re}^{-1}\,\Delta \mathbf{u} + \mathbf{u}\cdot\nabla\mathbf{u} + \nabla p = \mathbf{b}, \tag{1}$$

$$-\mathrm{div}\,\mathbf{u} = 0, \tag{2}$$

with $t \in [0, T_{\mathrm{end}}]$. At the boundary $\partial\Omega$ of the domain Ω, we impose Dirichlet boundary conditions for the velocity:

$$\mathbf{u} = \mathbf{g} \quad \text{on } \partial\Omega. \tag{3}$$

Here, $\mathrm{Re} = UL/\nu$ is the Reynolds number based on a characteristic velocity, length, and kinematic viscosity. Furthermore, $\mathbf{b}$ is a force vector and $\mathbf{g}$ contains the Dirichlet boundary conditions. The extension to Neumann or mixed boundary conditions is straightforward. As a starting point for the spectral element discretization, we use the variational equivalences of (1)–(2): Find $(\mathbf{u}, p)$ in $X_g \times M$ such that $\forall \mathbf{w} \in X_0$, $\forall q \in M$,

$$\left(\frac{\partial \mathbf{u}}{\partial t}, \mathbf{w}\right) + \mathrm{Re}^{-1}(\nabla\mathbf{u}, \nabla\mathbf{w}) + (\mathbf{u}\cdot\nabla\mathbf{u}, \mathbf{w}) - (p, \mathrm{div}\,\mathbf{w}) = (\mathbf{b}, \mathbf{w}) \tag{4}$$

$$-(q, \mathrm{div}\,\mathbf{u}) = 0, \tag{5}$$

where

$$\forall \phi, \psi \in \mathcal{L}^2(\Omega) \quad (\phi, \psi) = \int_\Omega \phi(\mathbf{x})\psi(\mathbf{x})\,d\mathbf{x}, \quad \mathbf{x} \in \Omega. \tag{6}$$

The space $\mathcal{L}^2(\Omega)$ is the space of all square integrable functions over Ω. The space X_g for the velocity, the space X_0 for the test functions, and the space M for the pressure are defined as follows:

$$X_g = \{v \in |\mathcal{H}^1(\overline{\Omega})|^3,\ v \text{ satisfies boundary conditions}\} \tag{7}$$

$$X_0 = \{v \in |\mathcal{H}^1(\overline{\Omega})|^3,\ v \text{ vanishes at } \partial\Omega\} \tag{8}$$

$$M = \mathcal{L}^2_0(\Omega) = \{\phi \in \mathcal{L}^2(\Omega);\ \int_\Omega \phi(x)\,dx = 0\}. \tag{9}$$

Expression (9) should be interpreted as an averaging procedure for the pressure. $\mathcal{H}^1(\Omega)$ is the space of all square integrable functions whose first-order derivatives are also square integrable over Ω.

The first step in the discretizaton process is to subdivide the domain $\overline{\Omega} = \partial\Omega\,\Omega$ into K_e non-overlapping rectilinear elements $\overline{\Omega}_k$ such that the intersection of two or more neighboring elements is either a face, an edge, or a vertex. In order to simplify the notation, we assume that the number of nodes N is equal in each direction and on each element. Of course, this does not affect the general concept. Next, we have to define the discrete polynomial subspaces $X_{g,h} \subset X_g$ and $M_h \subset M$, in which the velocities and pressure will be approximated, respectively. In order to avoid spurious pressure modes, Maday and Patera [13], and Bernardi and Maday [2] proposed the use of the subspaces

$$X_{g,h} = X_g \cap \mathcal{P}^3_{N,K_e}(\Omega) \tag{10}$$

$$M_h = M \cap \mathcal{P}_{N-2,K_e}(\Omega), \tag{11}$$

with $\mathcal{P}_{N,K_e} = \{\phi \in \mathcal{L}^2(\Omega);\ \phi_{|\Omega_k}$ is a polynomial of degree less than or equal to $N\}$. Consequently, the space $X_{0,h}$ is defined as

$$X_{0,h} = X_0 \cap \mathcal{P}^3_{N,K_e}(\Omega). \tag{12}$$

The choice for the spaces (10)–(11) implies the introduction of staggered grids. In our case, the velocities will be approximated on a Gauss–Lobatto–Legendre grid, whereas the pressure will be approximated on a Gauss–Legendre grid. Furthermore, the velocities are continuous along the element boundaries (while the pressure is *not* necessarily continuous).

The spectral element discretization proceeds by the application over each subdomain of two Gaussian integration rules, corresponding to the two grids mentioned above. We should remark that the three-dimensional rules are obtained by tensor products of the one-dimensional formulas.

The final step consists in the discretization in time. In this

paper we confine ourselves to a simple time scheme. We choose backward Euler for the viscous term, where the convective terms at time t_{n+1} are approximated by an explicit time-integration method. The third-order explicit Adams–Bashforth scheme has been applied for its advantageous stability characteristics; the overall order is one. We write the discrete equations immediately in matrix notation, where we use the same symbols as in [13, 18],

$$\frac{1}{\Delta t} B\mathbf{u}_i^{n+1} + \mathrm{Re}^{-1} A\mathbf{u}_i^{n+1} - D_i^{\mathrm{T}} p^{n+1} = B\mathbf{f}_i^{n+1}, \quad i = 1, 2, 3. \tag{13}$$

$$- D_i \mathbf{u}_i^{n+1} = 0. \tag{14}$$

Here, B is the diagonal mass matrix, A is the discrete Laplace operator, D_i is the discrete divergence operator, and the superscript T indicates the transpose. The right-hand-side vector $\mathbf{f}^{n+1}$ contains the volume force $\mathbf{b}^{n+1}$ and the explicit terms.

The Uzawa algorithm is applied to uncouple the velocities from the pressure. This technique was originally designed for finite element computations, but is nowadays used in spectral element computations as well (see [13, 18]). The attractive property of the Uzawa method is that the uncoupled system, which consists of four positive (semi-)definite, symmetric sets of equations, is equivalent to the original system. Starting from the discrete equations (13)–(14), the Uzawa algorithm is obtained by multiplication of the momentum equations by $D_i H^{-1}$, with

$$H = (\mathrm{Re}^{-1} A + \Delta t^{-1} B). \tag{15}$$

We obtain

$$-D_i H^{-1} D_i^{\mathrm{T}} p^{n+1} = D_i H^{-1} B\mathbf{f}_i^{n+1} \tag{16}$$

$$H\mathbf{u}_i^{n+1} = D_i^{\mathrm{T}} p^{n+1} + B\mathbf{f}_i^{n+1}, \quad i = 1, 2, 3. \tag{17}$$

Equation (16) is solved by the PCGM. Clearly, the Helmholtz operator H has to be inverted within each PCGM-iteration. This can efficiently be done by the Schur complement method in combination with the FDM, which will be discussed in the following sections.

3. THE FAST DIAGONALIZATION METHOD

For a tensorizable and separable operator it is possible to explicitly construct an inverse having a similar tensor product structure. Under certain conditions, which we will discuss later, H is such an operator. According to Lynch *et al.* [12], we can write

$$H^{-1} = P_z \otimes P_y \otimes P_x (I \otimes I \otimes \Lambda_x + I \otimes \Lambda_y \otimes I + \Lambda_z \otimes I \otimes I)^{-1} P_z^{-1} \otimes P_y^{-1} \otimes P_x^{-1}. \tag{18}$$

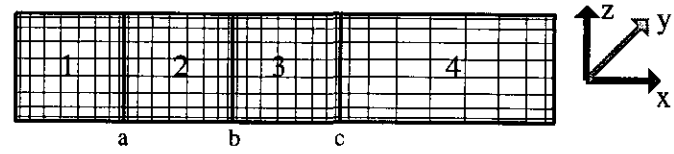

FIG. 1. Geometry consisting of four spectral elements. Projection in y-direction ($y = 1$).

Here, $A \otimes B$ denotes the tensor product of A and B, and Λ_x, Λ_y, and Λ_z are the diagonal matrices of eigenvalues arising from the (1D) generalized eigenvalue problems $Hu = \lambda u$ in x-, y-, and z-directions. The matrices P_x, P_y, and P_z are the corresponding matrices containing the eigenvectors. The interest in this method becomes clear when the cost of an evaluation of $H^{-1}\mathbf{v}$ is compared to the matrix-vector multiplication $H\mathbf{v}$, the basic ingredient of any iterative method. Using (18), $H^{-1}\mathbf{v}$ requires $6N^4$ multiplications, whereas $H\mathbf{v}$ takes $3N^4$ multiplications. So the inverse is computed at the price of two matrix-vector products.

Many authors have used this fast diagonalization technique in the context of spectral methods, e.g., [9, 10]. Streett and Hussaini [17] describe the application of this method to the Uzawa technique. All these papers are based on mono-domain computations. On the one hand this restriction seems obvious, since the condition that the operator should be separable, does not allow non-rectangular geometries. If the interface and boundary variables, however, are eliminated, we can use the FDM for the interior nodes of each spectral element, *provided that these elements are rectangular.* Note that, in fact, H is not a separable operator due to a multiplication by the weights. Therefore, we solve the equivalent problem $(HB^{-1})B\mathbf{u} = \mathbf{f}$ instead. To avoid complex notation, this diagonal shift is not explicitly represented in this paper. In the next paragraph we will describe the Schur complement method which separates the interior variables from the interface variables. In this way, we can take full advantage of the FDM.

4. SCHUR COMPLEMENT METHOD

4.1. *Construction of the Schur Complement*

The use of the Schur method is a common practice in modern numerical mechanics. The reduction of the problem to a set of subproblems often leads to memory savings and faster algorithms. Moreover, these subproblems can be solved independently, leading to a high degree of parallelism. Another argument is that different solvers can be applied to the interior and to the interface variables. In our case, the latter reason is the most important, although we will also discuss a parallel implementation in Section 5.3.

In order to explain the Schur method, we consider a domain as depicted in Fig. 1. The parallelepipedic domain $\overline{\Omega}$ is decomposed into four spectral elements $\overline{\Omega}_1, \overline{\Omega}_2, \overline{\Omega}_3$, and $\overline{\Omega}_4$. The first

three elements are of size $[0, 1] \times [0, 2] \times [0, 1]$, but the fourth element is two times as large in the x-direction. The interfaces Γ_a, Γ_b, and Γ_c are defined as $\Gamma_a = \overline{\Omega}_1 \cap \overline{\Omega}_2$, $\Gamma_b = \overline{\Omega}_2 \cap \overline{\Omega}_3$, and $\Gamma_c = \overline{\Omega}_3 \cap \overline{\Omega}_4$. Boundary variables are assumed to be eliminated. We introduce the following notations for the unknowns u and the right-hand side f: f_1, $u_1 \in \Omega_1$, f_2, $u_2 \in \Omega_2$, f_3, $u_3 \in \Omega_3$, f_4, $u_4 \in \Omega_4$, f_a, $u_a \in \Gamma_a$, f_b, $u_b \in \Gamma_b$, and f_c, $u_c \in \Gamma_c$. The Helmholtz equation $H\mathrm{u} = f$ can be written as

$$\begin{pmatrix} H_{11} & 0 & 0 & 0 & H_{1a} & 0 & 0 \\ 0 & H_{22} & 0 & 0 & H_{2a} & H_{2b} & 0 \\ 0 & 0 & H_{33} & 0 & 0 & H_{3b} & H_{3c} \\ 0 & 0 & 0 & H_{44} & 0 & 0 & H_{4c} \\ H_{a1} & H_{a2} & 0 & 0 & H_{aa} & H_{ab} & 0 \\ 0 & H_{b2} & H_{b3} & 0 & H_{ba} & H_{bb} & H_{bc} \\ 0 & 0 & H_{c3} & H_{c4} & 0 & H_{cb} & H_{cc} \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_a \\ u_b \\ u_c \end{pmatrix} = \begin{pmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \\ f_a \\ f_b \\ f_c \end{pmatrix}. \tag{19}$$

The following notations have been used:

H_{ii}, discretization of the Helmholtz operator on subdomain i (internal nodes)

$H_{i\alpha}$, coupling between the unknowns for subdomain i and interface α

$H_{\alpha i}$, coupling between the unknowns for interface α and subdomain i

$H_{\alpha\beta}$, coupling between the unknowns for interface α and interface β.

We have that $H_{ii} \in \mathcal{R}^{N^3 \times N^3}$, $H_{\alpha i} \in \mathcal{R}^{N^2 \times N^3}$, and $H_{\alpha\beta} \in \mathcal{R}^{N^2 \times N^2}$, $i \in \{1, 2, 3, 4\}$, $\alpha, \beta \in \{a, b, c\}$. Moreover, it can be shown that $H_{i\alpha} = H_{\alpha i}^{\mathrm{T}}$ and $H_{\alpha\beta} = H_{\beta\alpha}^{\mathrm{T}}$. Elimination of the variables at the interior nodes leads to the following system, often called the Schur complement:

$$\begin{aligned} &(H_{aa} - H_{a1}H_{11}^{-1}H_{1a} - H_{a2}H_{22}^{-1}H_{2a})u_a + (H_{ab} - H_{a2}H_{22}^{-1}H_{2b})u_b \\ &\qquad = f_a - H_{a1}H_{11}^{-1}f_1 - H_{a2}H_{22}^{-1}f_2 \\ &(H_{bb} - H_{b2}H_{22}^{-1}H_{2b} - H_{b3}H_{33}^{-1}H_{3b})u_b + (H_{ba} - H_{b2}H_{22}^{-1}H_{2a})u_a \\ &\qquad + (H_{bc} - H_{b3}H_{33}^{-1}H_{3c})u_c = f_b - H_{b2}H_{22}^{-1}f_2 - H_{b3}H_{33}^{-1}f_3 \\ &(H_{cc} - H_{c3}H_{33}^{-1}H_{3c} - H_{c4}H_{44}^{-1}H_{4c})u_c + (H_{cb} - H_{c3}H_{33}^{-1}H_{3b})u_b \\ &\qquad = f_c - H_{c3}H_{33}^{-1}f_3 - H_{c4}H_{44}^{-1}f_4. \end{aligned} \tag{20}$$

System (20) shows that the unknowns of the three interfaces, although they do not have any node in common, are coupled in a direct way. Moreover, the appearance of the matrices $H_{\alpha\beta}$ $(\alpha \neq \beta)$ is due to the high-order approximations and is absent in classical discretization methods, as the finite element or the finite difference technique. Schematically, the Schur complement of our example has the form

$$\begin{pmatrix} \square & \square & 0 \\ \square & \square & \square \\ 0 & \square & \square \end{pmatrix} \begin{pmatrix} u_a \\ u_b \\ u_c \end{pmatrix} = \text{r.h.s.}, \tag{21}$$

where the square box $\square$ represents an $N^2 \times N^2$ full block matrix. The FDM is used to evaluate the expressions $H_{ii}^{-1}f_i$ in the right-hand side of (20) and to compute the variables at the interior nodes:

$$\begin{aligned} H_{11}u_1 &= -H_{1a}u_a + f_1 \\ H_{22}u_2 &= -H_{2a}u_a - H_{2b}u_b + f_2 \\ H_{33}u_3 &= -H_{3b}u_b - H_{3c}u_c + f_3 \\ H_{44}u_4 &= -H_{4c}u_c + f_4. \end{aligned} \tag{22}$$

In case of a more general (curvy) geometry, the Schur complement method can still be useful. Although the FDM does not apply anymore, the inverse of the Helmholtz operators can be computed by a classical direct method or by an iterative method preconditioned by finite elements [5]. The advantage is that these computations are decoupled per element.

The Schur complement problem (20), involving the interface variables, can be solved either by a direct or by an iterative method. In the case of a direct method, it is clear that the implicit, tensorized form in which the Schur matrix (20) is written, should be replaced by an explicit formulation. It is also interesting to construct the matrix explicitly when an iterative method, like PCGM, is used. This becomes clear when we take a closer look at the block-matrices that form the Schur matrix. Using expression (18) to evaluate H_{11}^{-1} and H_{22}^{-1}, the number of operations to multiply one of the block-matrices, for example, $(H_{aa} - H_{a1}H_{11}^{-1}H_{1a} - H_{a2}H_{22}^{-1}H_{2a})$, is $17N^4$. When this block is constructed explicitly, the operation count of a block-vector multiplication is reduced to N^4. Moreover, since the explicit formulation does not contain tensor products, block-vector products can efficiently be computed on vector computers. It can be shown that the price to construct these blocks is $O(N^5)$. This computation is done once and for all in a preprocessing stage, so that its cost will be amortized.

4.2. *Preconditioning*

Let us first discuss the preconditioning of the original Helmholtz operator H. The condition number of this operator depends on Δt. For small values of Δt, the operator $H = (\mathrm{Re}^{-1}A + \Delta t^{-1}B)$ tends to the diagonal matrix B, containing the Gauss–Lobatto–Legendre weights. For larger values of Δt, the Laplacian A becomes dominant. The matrix $B^{-1}A$ is ill-conditioned, since its condition number is proportional to N^4 [18]. A precon-

ditioner that works well for small and large values of Δt is the diagonal of the Helmholtz matrix H. This preconditioner is used when we compare the new method to the classical one. Alternatives could be a preconditioner based on the incomplete Choleski [13] or the finite element method.

In this paper, however, we are not concerned with the preconditioning of the Helmholtz operator, but with the associated Schur complement. Although the condition number of the Schur matrix is smaller than that of the original system, preconditioning is still essential. Many preconditioners have been proposed in the literature, e.g., [3, 4]. In this paper, we will consider two preconditioners; the inverse of the diagonal of (20) and the block diagonal matrix, which has as entries the inverse of the diagonal blocks of (21). The first preconditioner is easy to construct and its cost per multiplication is negligible, but its efficiency is low, as will be shown later. The effect of the block diagonal preconditioner, also called block Jacobi, is impressive. A question, however, is whether the price to construct this preconditioner (K_e matrices of dimension $N^2 \times N^2$ have to be inverted) is not too high, especially when compared to the inversion of the complete system (20). For this simple, four-element problem this might indeed be the case, but it is interesting to compare the cost for a larger number of interfaces. Let us define the number of interfaces by K_i. Taking into account that the block matrices on the diagonal are symmetric, computing the preconditioner requires $K_i N^6/6$ operations, whereas the inversion of the complete Schur complement requires $K_i^3 N^6/6$ operations. Hence, the iterative method becomes attractive for larger K_i. The construction of the preconditioner is performed once and for all and the cost is spread out over hundreds of pressure iterations.

In order to give some heuristics about the spectrum of the Schur complement and of the Schur complement premultiplied by the diagonal and the block-diagonal preconditioners, we computed the condition numbers of these three operators for different values of K_i, N, and $\Delta t \mathrm{Re}^{-1}$. Table I represents the results for a geometry with dimensions $[0, 1] \times [0, 4] \times [0, 1]$, with respectively 4, 8, and 16 elements in the y-direction and one in the remaining directions. For this particular mesh, the Schur complement is tridiagonal by block. We remark that the results representing a Navier–Stokes simulation ($\Delta t = 0.01$, $\mathrm{Re} = 10$) give rise to condition numbers close to one. Moreover, they seem to be more or less independent of the number of interfaces. For $\Delta t = 0.25$, $\mathrm{Re} = 1$, we see that the condition number of the Schur complement, for a fixed value of K_i, is of order $O(N^2)$. The diagonal preconditioner seems to reduce the condition number to $O(N)$ and the block-diagonal preconditioner yields a condition number that is independent of N. There is no obvious relation between the spectrum and K_i, other than that the condition number grows as K_i increases.

Although the construction of an efficient preconditioner for the pressure operator (16) is beyond the scope of this paper, we make the following remarks: for large values of Δt, this operator is well conditioned and can be preconditioned by a diagonal matrix containing the Gauss–Legendre weights. The problem is much more difficult for small values of Δt. In a recent paper, Rønquist [16] proposes a preconditioner based on the decomposition of the pressure system into two pressure systems. Here, we confine ourselves to the simple diagonal preconditioner based on the Gauss–Legendre weights.

5. COMPARISON OF DIFFERENT METHODS

5.1. *Numerical Results for a Simple Test Problem*

The new algorithm, based on the Schur method and the direct inversion is compared to the classical one, where the complete Helmholtz equation is solved by the PCGM. The difference between a direct and an iterative method in solving the Schur complement is examined. Moreover, in the case of an iterative solver, the performance of the preconditioners is investigated. The tests have been run on a Convex C3820 vector computer. Every program has been compiled with and without vector optimization. The routines that perform the multiplication by the Schur complement or by the inverse of the Schur complement (in the case of an iterative or direct method, respectively) are written in BLAS.

As a test problem, we have taken the Stokes flow, in which the non-linear terms in (1) are neglected. The geometry is given in Fig. 1. The degree of the approximating polynomials is equal to 10 for the velocities and 8 for the pressure, $K_e = 4$ and $K_i = 3$. The analytical solution is given by

$$\mathbf{u}(x_1, x_2, x_3) = \left(-\cos\left(\frac{\pi}{2}x_1\right)\sin\left(\frac{\pi}{2}x_3\right), 0, \sin\left(\frac{\pi}{2}x_1\right)\cos\left(\frac{\pi}{2}x_3\right)\right)^{\mathrm{T}} \tag{23}$$

$$p(x_1, x_2, x_3) = -\pi \sin\left(\frac{\pi}{2}x_1\right)\sin\left(\frac{\pi}{2}x_3\right).$$

At $t = 0$, the fluid is at rest and the boundary conditions match the analytical solution. After 10 time steps ($\Delta t = 0.5$) a steady solution is obtained. Four different methods are compared: The classical iterative (CI) method, the direct Schur complement (DS) method, the iterative Schur complement method preconditioned by the diagonal (ISD), and the iterative Schur complement method preconditioned by the block-diagonal (ISB). Table II shows the results of the runs without vector optimization. The accuracy of the four methods is of the same order. We found a maximum error of 3×10^{-5} for the pressure and of 8×10^{-7} for the velocities. The speed of the three methods which are based on the Schur method (DS, ISD, and ISB) is much higher than that of the classical method. Furthermore, we note that the direct inversion of the Schur complement (DS) is faster than in the iterative methods (ISB and ISD). Finally, preconditioning with the block-diagonal matrix is preferred to

TABLE I

Condition Number of the Schur Complement (CS), the Schur Complement Premultiplied by the Diagonal Preconditioner ($D^{-1}CS$), and the Schur Complement Premultiplied by the Block-Diagonal Preconditioner ($(BD)^{-1}CS$) for Different Values of K_i and N

		$\Delta t = 0.25$, Stokes			$\Delta t = 0.01$, Re = 10		
K_i	N	CS	$D^{-1}CS$	$(BD)^{-1}CS$	CS	$D^{-1}CS$	$(BD)^{-1}CS$
3	5	2.66	2.45	1.02	1.10	1.09	1.00
3	7	4.61	3.67	1.02	1.29	1.22	1.00
3	9	7.36	4.92	1.02	1.57	1.36	1.00
3	11	10.94	6.19	1.02	1.98	1.54	1.00
7	5	2.88	2.68	1.38	1.08	1.08	1.00
7	7	5.13	4.14	1.38	1.21	1.16	1.00
7	9	8.27	5.63	1.38	1.47	1.30	1.00
7	11	12.34	7.14	1.38	1.89	1.50	1.00
15	5	4.84	4.60	3.29	1.07	1.06	1.00
15	7	7.93	6.50	3.29	1.19	1.15	1.00
15	9	12.68	8.67	3.29	1.46	1.30	1.00

the diagonal preconditioner. We found that, for any Schur method, the preprocessing time (construction of the Schur complement and preconditioner, eigenvalue decomposition of mono-dimensional operators necessary for fast diagonalization) is less than 2 s.

The results for the vectorized programs can be found in Table III. We note that, as expected, the Schur methods benefit more from vectorization than the classical method. The difference between the methods DS, ISD, and ISB has almost completely disappeared. This can be explained as follows: The time for the computation of the interfaces has become negligible with respect to the computation of the interior nodes, which, due to the tensor products, does not vectorize well. An analysis of the distribution of the cpu time over the different subroutines reveals that, for this particular problem, the ISB method spent a considerable amount of time (about 47%) to compute the interior nodes by FDM, whereas the interface routines took only 5%.

5.2. *Numerical Results on an Eight-Element Geometry*

In this paragraph we consider a more complex geometry, consisting of the cube $[0, 1]^3$ with a hole in the center $[0.4, 0.6] \times [0, 1] \times [0.4, 0.6]$. The number of spectral elements is eight ($K_i = K_e = 8$) and the polynomial degree is 10. Figure 2 shows a projection of the cube in the y-direction. Homogeneous Dirichlet boundary conditions are applied everywhere, except at the top plane ($z = 1$), where $\mathbf{u}_1 = 16xy(x-1)(y-1)$. The Schur complement that results after elimination of the interior nodes is somewhat more complex than that of the four-element matrix (20). We will refrain from giving the full system and give, as an example, the equations at the interface a,

$$(H_{aa} - H_{a1}H_{11}^{-1}H_{a1}^{T} - H_{a2}H_{22}^{-1}H_{a2}^{T})u_a + (H_{ab} - H_{a2}H_{22}^{-1}H_{b2}^{T})u_b - H_{a1}H_{11}^{-1}H_{h1}^{T}u_h = f_a - H_{a1}H_{11}^{-1}f_1 - H_{a2}H_{22}^{-1}f_2. \quad (24)$$

The equations at the other interfaces are given by similar expressions, since the geometry is symmetric with respect to the interfaces. Schematically, the Schur complement can be written in the form

$$\begin{pmatrix} \square & \square & 0 & 0 & 0 & 0 & 0 & \square \\ \square & \square & \square & 0 & 0 & 0 & 0 & 0 \\ 0 & \square & \square & \square & 0 & 0 & 0 & 0 \\ 0 & 0 & \square & \square & \square & 0 & 0 & 0 \\ 0 & 0 & 0 & \square & \square & \square & 0 & 0 \\ 0 & 0 & 0 & 0 & \square & \square & \square & 0 \\ 0 & 0 & 0 & 0 & 0 & \square & \square & \square \\ \square & 0 & 0 & 0 & 0 & 0 & \square & \square \end{pmatrix} \begin{pmatrix} u_a \\ u_b \\ u_c \\ u_d \\ u_e \\ u_f \\ u_g \\ u_h \end{pmatrix} = \text{r.h.s.} \quad (25)$$

System (25) illustrates that for large K_i direct inversion of the

TABLE II

Timings in Seconds after 10 Time Steps for the Test Problem without Vector Optimization

Method	CI	DS	ISD	ISB
Seconds	2637	191	330	223

TABLE III

Timings in Seconds after 10 Time Steps for the Test Problem with Vector Optimization

Method	CI	DS	ISD	ISB
Seconds	1565	105	114	111

Schur complement is not a good idea (see discussion in Section 4.2). Therefore, an iterative method (ISB) is preferred.

Table IV compares the cpu time of the ISB and CI methods for the first time step of a Stokes flow and a Navier–Stokes flow (Re = 100, where the characteristic velocity U equals one). Again, the new algorithm is much faster than the previous one. The large differences between the computation times for the Stokes and Navier–Stokes problems can be explained by the fact that for small values of Δt more iterations will be needed to compute the pressure, whereas the Helmholtz operator is well conditioned, resulting in a low number of internal iterations, as is illustrated by the last line of Table IV.

5.3. *Closing Remarks*

In Fischer *et al.* [6] and in Fischer and Patera [8], the parallelization of the classical CI method is discussed. Large computational kernels, like the computation of gradients and the multiplication by the Helmholtz operator H can be performed in parallel on each spectral element.

The parallel efficiency of our method will not differ very much from that of the original method, at least when computers based on shared memory are considered; the additional computational kernels can naturally be parallelized. The work to construct the Schur complement method and its preconditioner can be divided over different processors. Clearly, the computation of the interior nodes by FDM can be done independently on each element. Finally, when an iterative method is used to solve

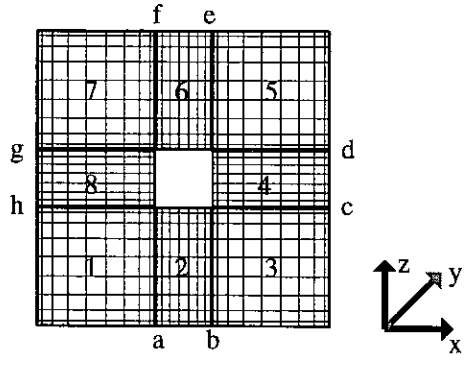

FIG. 2. Geometry consisting of eight spectral elements. Projection in y-direction (y = 0.05). The sets of interior nodes are enumerated 1 ... 8; the interfaces a ... h.

TABLE IV

Timings in Seconds after One Time Step for the Eight-Element Problem on a Convex 3820 using Vector Optimization

	Stokes Δt = 0.5		Navier–Stokes Re = 100, Δt = 0.005	
Method	CI	ISB	CI	ISB
Seconds	702	40	670	140
Average number of iterations	87	18	17	6

Note. The average number of iterations (tolerance 10^{-12}) is given per Helmholtz solve (CI) or per solve of the Schur complement (ISB).

the Schur complement system, the matrix-vector multiplication is split in block matrix-vector operations. In Table V we give the results on the parallel Alliant FX/8 for the Stokes and Navier–Stokes problems, defined in the previous paragraph. The parallel efficiency is about 85% for the Stokes and 80% for the Navier–Stokes.

On distributed memory computers, where communication is an issue, the Schur method might have a disadvantage compared to the original CI method. According to Fischer *et al.* [6], the latter method only requires the communication of scalars (to assemble the dot product) and points at the interfaces (for the direct stiffness). The interfaces are only exchanged between neighboring elements. In our case, the FDM and the construction of the preconditioner are communication free. The construction of the Schur complement matrix requires some communication, but this is done only once in a preprocessing stage. The difficulty is related to the implementation of the iterative method to solve the Schur complement. Since the Schur matrix is in general not block diagonal (see, for example, (21), (25)), the interface variables have to be exchanged between processors. In the case of system (25), for instance, u_b and u_h have to be sent to the processor that computes the first row. The time to send these $O(N^2)$ messages can be relatively large with respect to the fast block matrix-vector multiplication, which is

TABLE V

Timings in Seconds after the First Time Step for the Test Problem on an Alliant FX/8

	Stokes Δt = 0.5		Navier–Stokes Re = 100, Δt = 0.005	
Mode	Vec	Vec + par	Vec	Vec + par
No. of proc.	1	4	1	4
Seconds	544	160	1694	532

Note. The method used is ISB.

of order $O(N^4)$. Preliminary results show that the performance of the iterative solver depends heavily on the architecture of the parallel distributed memory computer.

The algorithm we investigated in this paper leads to an important acceleration of the Uzawa algorithm applied to the discrete Navier–Stokes equations. Since the conditions for the FDM are very restrictive, the present method can only be applied to geometries consisting of non-deformed spectral elements. This is a serious limitation. However, since the Schur method decouples the global Helmholtz operator H in K_e local operators H_{ii}, classical direct methods can also be considered. Once the inverses H_{ii}^{-1} (or the LL^T decomposition) have been computed, the Schur complement matrix can be constructed, yielding no extra computational effort to solve the Schur complement problem, apart from a higher preprocessing cost. The decoupled problems for the interior nodes ($H_{ii}u_i =$ r.h.s.) can be solved either by multiplication by the local inverse (or by back substitution) or by an iterative method preconditioned by finite elements [5]. Fischer and Rønquist [7] already showed in the context of preconditioning of the pressure operator that the construction of local inverses by a classical direct method is feasible and advantageous in terms of computation times. Both the parallelization on distributed memory machines and the implementation of deformed geometries will be the subject of future research.

ACKNOWLEDGMENT

The above text presents research results of the Belgian Incentive Program "Information Technology—Computer Science of the Future," initiated by the SPPS (Services du Premier Ministre. Programmation de la Politique Scientifique). The scientific responsibility is assumed by the authors.

REFERENCES

1. K. Arrow, L. Hurwicz, and H. Uzawa, *Studies in Nonlinear Programming* (Stanford Univ. Press, Stanford, 1968).
2. C. Bernardi and Y. Maday, *Intern. J. Numer. Methods Fluids* **8,** 537 (1988).
3. P. E. Bjørstad and O. B. Widlund, *SIAM J. Numer. Anal.* **23**(6), 1097 (1986).
4. T. F. Chan and D. Goovaerts, *Appl. Numer. Math.* (special issue on spectral multi-domain methods) **6,** 53 (1989).
5. M. O. Deville and E. H. Mund, *SIAM J. Sci. Stat. Comput.* **12,** 311 (1990).
6. P. F. Fischer, E. Rønquist, D. Dewey, and A. T. Patera, in *First Int. Symp. on Domain Decomposition Methods for Partial Differential Equations, SIAM, 1987,* p. 397.
7. P. F. Fischer and E. Rønquist, in *Proceedings, 2nd International Conference on Spectral and High-Order Methods, Montpellier, France, 1992.*
8. P. F. Fischer and A. T. Patera. *J. Comput. Phys.* **92,** 380 (1991).
9. D. B. Haidvogel and T. Zang, *J. Comput. Phys.* **30,** 167 (1979).
10. P. Haldenwang, G. Labrosse, S. A. Abboudi, and M. O. Deville, *J. Comput. Phys.* **55,** 115 (1984).
11. G. E. M. Karniadakis, M. Israeli, and S. A. Orszag, *J. Comput. Phys.* **97,** 414 (1991).
12. R. E. Lynch, J. R. Rice, and D. H. Thomas, *Numer. Math.* **6,** 185 (1964).
13. Y. Maday and A. T. Patera, in *State-of-the-Art Surveys on Computational Mechanics,* edited by A. K. Noor and J. T. Oden (ASME, New York, 1989), p. 71.
14. A. T. Patera, *J. Comput. Phys.* **65**(2), 474 (1986).
15. Y. Maday, A. T. Patera, and E. M. Rønquist, *J. Sci. Comput.* **5**(4), 263 (1990).
16. E. M. Rønquist, in *Proceedings, Fifth Conference on Domain Decomposition Methods for Partial Differential Equations, Norfolk, USA, 1991,* edited by D. E. Keyes *et al.,* 545.
17. C. Streett and M. Y. Hussaini, AIAA Paper 87-1444 (unpublished).
18. E. M. Rønquist, "Spectral Element Methods for the Unsteady Navier–Stokes Equations," *Lecture Series on Computational Fluid Dynamics,* Von Karman Institute for Fluid Dynamics, 1991-01.