Journal of Computational Physics **175,** 438–453 (2002)
doi:10.1006/jcph.2001.6907, available online at http://www.idealibrary.com on IDEAL®

Algebraic Splitting for Incompressible Navier–Stokes Equations

Martin Ofstad Henriksen and Jens Holmen

Department of Structural Engineering, NTNU, N-7491 Trondheim, Norway
E-mail: martin.henriksen@bygg.ntnu.no; jens@math.ntnu.no

Received October 11, 2000; revised July 11, 2001

Fully discretized incompressible Navier–Stokes equations are solved by splitting the algebraic system with an approximate factorization. This splitting affects the temporal convergence order of velocity and pressure. The splitting error is proportional to the pressure variable, and a simple analysis shows that the original convergence order of the time-integration scheme can be retained by solving for incremental pressure. The combination of splitting and incremental pressure is shown to be equivalent to an error-correcting method using the full pressure. In numerical experiments employing a third-order time-integration scheme and various orders for the pressure increment, the splitting error is shown to control the convergence order, and the full order of the scheme is recaptured for both velocity and pressure. The difference between perturbing the momentum or the continuity equation is also explored.

Key Words: Navier–Stokes equations; block LU factorization; time splitting; splitting error; fractional step method; incremental pressure; order of convergence.

1. INTRODUCTION

The Navier–Stokes equations for an incompressible Newtonian fluid are

$$\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla)\mathbf{u} = -\nabla p + \nu \Delta \mathbf{u} + \mathbf{f}, \tag{1}$$

$$\nabla \cdot \mathbf{u} = 0, \tag{2}$$

where the velocity $\mathbf{u} = \mathbf{u}(\mathbf{x}, t)$ varies over the open domain $\Omega \in \mathbb{R}^d$ (d is 2 or 3), and $\mathbf{f}$ is an external force. The scalar field p represents the pressure divided by the density, and the constant ν is the kinematic viscosity. At $t = t_0$ initial values are assumed; $\mathbf{u}(\mathbf{x}, t_0) = \mathbf{u}_0$ and $p(\mathbf{x}, t) = p_0$, where $\mathbf{u}_0$ and p_0 satisfy Eqs. (1) and (2) and the boundary conditions.

0021-9991/02 $35.00

We only consider Dirichlet conditions (this does not limit the applicability of the algebraic splitting method to other types and combinations of boundary conditions),

$$\mathbf{u} = \mathbf{w} \quad \text{for } \mathbf{x} \in \Gamma, \quad \text{and} \quad \int_\Gamma \mathbf{w} \cdot \mathbf{n} = 0,$$

where $\mathbf{w}$ is a known function. The pressure is well defined modulo an additive constant, and a requirement like

$$\int_\Omega p = 0$$

is needed for uniqueness.

The algebraic equations obtained when discretizing the problem are often virtually impossible to solve directly, and different strategies to reduce the number of central processing unit cycles and necessary computer memory have been developed. One possibility is to rewrite the partial differential equations (PDEs) to separate the variables (pressure and d velocity components). Perhaps the most successful PDE-modifying technique is the projection method based on the work of Chorin [2] and Temam [17] with a Poisson equation for the explicit calculation of the pressure. However, due to the requirements of the underlying Helmholtz–Hodge theorem [13, Sect. 7.2], the projection method in general corrupts the boundary conditions. In certain cases (e.g., when the forces on the boundaries are required), errors in this area of the domain may have a large impact on the quality of the computation. The problem can be avoided by a clever choice of bondary conditions in the intermediate steps of the projection method (see, e.g., [8]) or by the use of algebraic splitting.

Algebraic splitting as a tool for solving the Navier–Stokes equations was introduced by Dukowicz and Dvinsky [3] and further explored by Perot [11] and Quarteroni *et al.* [14]. This technique might be seen as the matrix equivalent of the fractional step or projection method, and in certain cases the equation systems that are solved are identical; see [14]. It is only concerned with block matrices, and hence can be used with any time-integration scheme and any method for spatial discretization. Boundary conditions are applied as for the fully coupled equations. Thus, the problems of nonphysical boundary conditions and numerical boundary layers inherent in the projection method are resolved. It is possible to a priori control the splitting error; it may, in principle, have any order. This makes algebraic splitting as accurate as solving the fully coupled equations.

In Refs. [11, 14], the splitting is done by approximate LU factorization. The temporal order of the splitting error is determined by a (truncated) von Neumann series, which approximates the inverse of a block matrix—see Section 4. In [3], a different type of matrix splitting is used, which is limited to second-order accuracy. There, it is suggested that higher order methods can be achieved by increasing the order of the pressure increment. The present paper shows that higher order methods can be successfully constructed by combining higher order pressure increments as in [3] and the approximate LU-factorization of [14]. This requires far fewer computations than the computing terms of the von Neumann series, and the presented method is believed to be useful in computational fluid dynamics.

Also new in this paper is an alternative interpretation of the use of incremental pressure. It is shown that it is equivalent to a particular error-correction of the nonincremental scheme and that certain schemes can easily be solved in their nonincremental form and still obtain a higher order. Such a variant of the method may be advantageous since it avoids accumulating rounding errors in the pressure.

When using a high-order pressure increment, the Schur-complement of the approximate LU factorization typically involves the mass matrix. In the context of the finite-element method, one wants to lump the mass matrix to increase the computational efficiency. As is seen in Section 4, a lumped mass matrix formally reduces the order of the factorization. This practical detail seems to be overlooked. The reason for this might be, as demonstrated in the numerical examples herein, that the error introduced is much smaller than the splitting error and hence does not limit the convergence.

In the next sections the algebraic splitting method and its incremental pressure versions are presented. We briefly discuss the proper choice of matrix approximations and then show that solving for incremental pressure reduces the error introduced by the splitting. The results are illustrated numerically by solving the equations with a third-order time-integration scheme.

2. ALGEBRAIC SPLITTING

We apply some suitable time discretization to the PDE system of Eqs. (1) and (2). Let t_n be the nth point in time, $\delta t = t_{n+1} - t_n$, and $\mathbf{u}^n \approx \mathbf{u}(t_n)$. As an example, the backward Euler scheme with linearized convection renders the semidiscretized system

$$\frac{\mathbf{u}^{n+1} - \mathbf{u}^n}{\delta t} + \nabla p^{n+1} = -(\mathbf{u}^n \cdot \nabla)\mathbf{u}^{n+1} + \nu \Delta \mathbf{u}^{n+1} + \mathbf{f}^{n+1}, \tag{3}$$

$$\nabla \cdot \mathbf{u}^{n+1} = 0. \tag{4}$$

A number of methods may be used for spatial discretization—we focus on the Galerkin finite-element method (FEM). Note that if, for example, finite differences are used instead, some of the following discussion becomes irrelevant, as the mass matrix is diagonal in that case. When discretizing Eqs. (3) and (4) with FEM, the elements for the velocities and the pressure should be chosen such that the "inf–sup" criterion of mixed FEMs (see, e.g., [15]) is fulfilled.

The discretized system assumes the form

$$\begin{bmatrix} C & G \\ D & 0 \end{bmatrix} \begin{bmatrix} \mathbf{u}^{n+1} \\ p^{n+1} \end{bmatrix} = \begin{bmatrix} \mathbf{r}_u^n \\ 0 \end{bmatrix} + \begin{bmatrix} \mathbf{b}_u^{n+1} \\ b_p^{n+1} \end{bmatrix}, \tag{5}$$

where uppercase letters are used for (block) matrices, and lowercase letters are used for vectors. Bold symbols like $\mathbf{u}^{n+1}$ signify that the entries are vectors, whereas the entries of p^{n+1} are scalars. $\mathbf{r}_u^n$ contains the external force and the velocity terms from previous time steps. Boundary conditions are enforced on the system; the vector $[\mathbf{b}_u^{n+1}, b_p^{n+1}]^{\mathrm{T}}$ reflects the necessary modifications to the right-hand side.

The backward Euler scheme in Eqs. (3) and (4) makes C equal to the sum of the consistent mass matrix $\delta t^{-1} M$, the linearized convection matrix $K(\mathbf{u}^n)$, and the diffusion matrix νL. The right-hand side accounts for everything known at time step $n+1$, like $\delta t^{-1} M \mathbf{u}^n$ and boundary conditions. The matrices D and G represent the diffusion and the gradient operator, respectively. These are generally nonsquare.

The large system can be made somewhat more manageable by block LU factorization. Following the notation used in [14] we write:

$$A = \begin{bmatrix} C & G \\ D & 0 \end{bmatrix} = \begin{bmatrix} C & 0 \\ D & -DC^{-1}G \end{bmatrix} \begin{bmatrix} I & C^{-1}G \\ 0 & I \end{bmatrix}.$$

To solve Eq. (5), we put

$$\begin{bmatrix} \tilde{\mathbf{u}}^{n+1} \\ \tilde{p}^{n+1} \end{bmatrix} = \begin{bmatrix} I & C^{-1} \\ 0 & I \end{bmatrix} \begin{bmatrix} \mathbf{u}^{n+1} \\ p^{n+1} \end{bmatrix}.$$

Since $p^{n+1} = I\tilde{p}^{n+1}$, forward and backward substitution now amounts to the three-step procedure

$$\begin{aligned} &1. && C\tilde{\mathbf{u}}^{n+1} = \mathbf{r}_u^n + \mathbf{b}_u^{n+1}, \\ &2. && DC^{-1}Gp^{n+1} = D\tilde{\mathbf{u}}^{n+1} + b_p^{n+1}, \\ &3. && C\mathbf{u}^{n+1} = C\tilde{\mathbf{u}}^{n+1} - Gp^{n+1}. \end{aligned} \tag{6}$$

In contrast to the projection method, $\tilde{\mathbf{u}}^{n+1}$ is not assigned any physical meaning and should be considered a purely computational convenience.

Computing the large and dense matrix C^{-1} is expensive, and an implicit method must update C and C^{-1} at every time step. Therefore the consistent operator $DC^{-1}G$, though it may have satisfying sparseness, will not be formed in any implementation for large problems. One may circumvent the inversion of C and keep the product in factored form by applying the Uzawa algorithm or the pressure matrix method (see [15, Chap. 9.6]). These techniques introduce extra inner iterations.

Substituting a proper approximation for C^{-1} can reduce the cost of the inner iterations or obliviate them. Following [14], we introduce H_1 and H_2, two (possibly identical) approximations of C^{-1}. Thus, the solution procedure above becomes

$$\begin{aligned} &1. && C\tilde{\mathbf{u}}^{n+1} = \mathbf{r}_u^n + \mathbf{b}_u^{n+1} \quad \text{(unchanged)}, \\ &2. && DH_1Gp^{n+1} = D\tilde{\mathbf{u}}^{n+1} + b_p^{n+1}, \\ &3. && \mathbf{u}^{n+1} = \tilde{\mathbf{u}}^{n+1} - H_2Gp^{n+1}. \end{aligned} \tag{7}$$

This system is equivalent to the system obtained by substituting for A the product of an *approximative* LU factorization, as

$$A \approx \hat{A} = \begin{bmatrix} C & CH_2G \\ D & D(H_2 - H_1)G \end{bmatrix} = \begin{bmatrix} C & 0 \\ D & -DH_1G \end{bmatrix} \begin{bmatrix} I & H_2G \\ 0 & I \end{bmatrix}. \tag{8}$$

Choosing $H_1 = H_2$ gives a solenoidal solution, while $H_2 = C^{-1}$ leaves the momentum equation unaltered. The latter was chosen in [14], and the former in [11]. It is not clear which alternative is the better; it probably depends on the specific problem. In the following, both alternatives are examined.

All boundary conditions must be imposed on the original, unfactorized algebraic system (5). No auxiliary or unphysical boundary conditions are needed—especially not for the pressure.

3. INCREMENTAL VERSION

A variant of the classical projection method, perhaps going back to Hirt and Cook [7], produces the pressure increment, instead of the pressure itself.

An incremental version of the algebraic splitting method is derived by subtracting $[Gp^n, 0]^{\mathrm{T}}$ from both sides of Eq. (5), setting $\delta p^{n+1} = p^{n+1} - P^n$, and then introducing the approximations H_1 and H_2. Only the solution vector and the right-hand side are changed, while the matrix $\hat{A}$ remains, as in the nonincremental case.

Let δ^l denote the lth-order backward difference operator, $\delta^l p^{n+1} = \delta^{l-1} p^{n+1} - \delta^{l-1} p^n$. This allows us to express the incremental algebraic splitting method of general order l as

$$\begin{bmatrix} C & 0 \\ D & -DH_1G \end{bmatrix} \begin{bmatrix} I & H_2G \\ 0 & I \end{bmatrix} \begin{bmatrix} \mathbf{u}^{n+1} \\ \delta^l p^{n+1} \end{bmatrix} = \begin{bmatrix} \mathbf{r}_u^n - G\sigma^l p^n \\ 0 \end{bmatrix} + \begin{bmatrix} \mathbf{b}_u^{n+1} \\ b_p^{n+1} \end{bmatrix}. \tag{9}$$

The vector $\sigma^l p^n = p^{n+1} - \delta^l p^{n+1}$ is computed only from the pressures at previous time steps. If we define $\delta^0 p^n = p^n$ and $\sigma^0 p^n = 0$, the above system includes also the nonincremental version of the previous section.

Thus, the split three-step procedure above is generalized as

$$\begin{aligned} &1. && C\tilde{\mathbf{u}}^{n+1} = \mathbf{r}_u^n - G\sigma^l p^n + \mathbf{b}_u^{n+1}, \\ &2. && DH_1G\delta^l p^{n+1} = D\tilde{\mathbf{u}}^{n+1} + b_p^{n+1}, \\ &3. && \mathbf{u}^{n+1} = \tilde{\mathbf{u}}^{n+1} - H_2G\delta^l p^{n+1}. \end{aligned} \tag{10}$$

Remark. It is the *unsplit* system that is reformulated to solve for the pressure increment, and then the approximate LU factorization is applied. If one reformulates the split system (by subtracting $[CH_2G\sigma^l p^n, D(H_2 - H_1)G\sigma^l p^n]^{\mathrm{T}}$), the concomitant splitting error will include an additional term which will be proportional to the full pressure.

4. APPROXIMATIONS TO C^{-1}

The choice of H_1 and H_2 is essential to the method. We want the approximations to C^{-1} to be sparse, to be easy to find, and perhaps to allow fast computation of the products DH_1G and H_2G. Requirements that will make H_1 and H_2 converge to C^{-1} when $\delta t \to 0$ may be established by examining C^{-1} itself.

Let $S = C - \delta t^{-1} M$, so that

$$C^{-1} = \delta t (I + \delta t M^{-1} S)^{-1} M^{-1}. \tag{11}$$

Using a von Neumann series, this is expanded as

$$C^{-1} = \delta t \left[\sum_{i=0}^{\infty} (-\delta t M^{-1} S)^i \right] M^{-1}. \tag{12}$$

The convergence criterion of the series is

$$\rho(M^{-1}S) < \delta t^{-1}, \tag{13}$$

where $\rho(\cdot)$ is the spectral radius. Since M and S depend on the space discretization, Eq. (13) relates δt to the largest element diameter h.

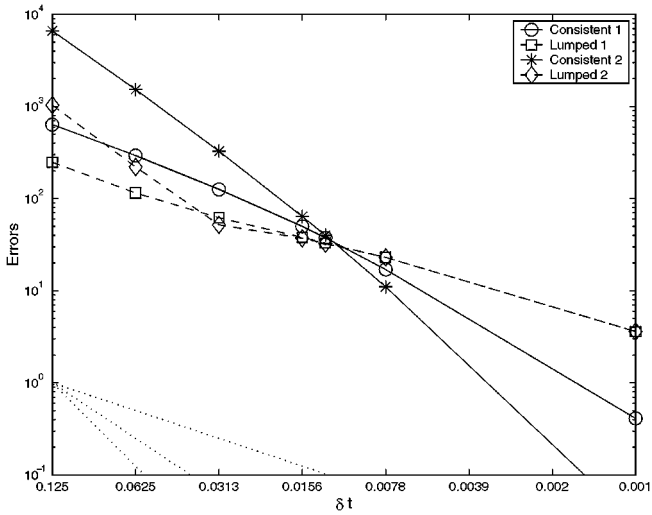

FIG. 1. Convergence in time of different approximation to C^{-1}. $\rho(M^{-1}S)^{-1} = 0.013$. The dashed lines in the corner show the inclinations of order 1, 2, and 3.

The matrix $\hat{X}(\delta t)$ is a kth-order approximation to $X(\delta t)$ in some norm whenever

$$\|X - \hat{X}\| = \mathcal{O}(\delta t^{k+1}).$$

The k first terms of Eq. (12) is a kth-order approximation to C^{-1}.

To use an approximation H_1 based on the von Neumann series, one needs to express M^{-1}. In the presence of iterative solvers, this is typically done via a pressure matrix method with H_1 in lieu of C^{-1}. Considering computational work, it is more attractive to substitute the lumped mass matrix M_{L} for the consistent mass matrix M in Eq. (12). Now, it is feasible to explicitly form the product DH_1G, and for small k (say, $k \leq 3$) the product is likely to have adequate sparseness. Explicit computation of the product permits the use of general-purpose iterative solvers.

The FEM does not produce a diagonal mass matrix, although as the grid is refined, $\|M - M_{\mathrm{L}}\| \to 0$. For any finite grid, $\delta t M_{\mathrm{L}}^{-1}$ is a zeroth-order approximation to C^{-1}, and the consequences of using lumped mass in H_1 and H_2 are not clear. Figure 1 demonstrates the behavior of four different approximations to C^{-1} as $\delta t \to 0$. The abscissa is $\|C^{-1} - H\|_2$ (H denotes the approximation), and the time step is the ordinate. The solid lines, denoted *Consistent 1* and *Consistent 2* in the figure legend, correspond to

$$H = \delta t M^{-1} \quad \text{and} \quad H = \delta t(I - \delta t M^{-1}S)M^{-1},$$

respectively, while the dashed lines, denoted *Lumped 1* and *Lumped 2*, are for

$$H = \delta t M_{\mathrm{L}}^{-1} \quad \text{and} \quad H = \delta t\left(I - \delta t M_{\mathrm{L}}^{-1}S\right)M_{\mathrm{L}}^{-1}.$$

Note that the lines cross for $\delta t \approx \rho(M^{-1}S)^{-1}$. The consistent approximations are clearly first and second order. Using lumped mass, however, the approximations eventually decay as $\mathcal{O}(\delta t)$ (i.e., zeroth order). It is interesting that for large δt, the lumped approximations show higher accuracy than do the consistent ones. Whether this is true for other flow cases

and different grids as well, we do not yet know, but there is a possibility that one might gain accuracy by using lumped mass.

The matrices in this experiment are as for the "Kim–Moin" case discussed in Section 8, but the high-order scheme is replaced by linearized backward Euler, and only 8^2 elements are used.

From practical simulations using FEM spatial discretization, we have found that if the criterion (13) is fulfilled, spatial errors will dominate the total error. In our small example (Fig. 1), it is only for this range of δt that use of two terms of Eq. (12) results in higher accuracy—whether lumping mass or not. Thus, using more than the first term seems unreasonable.

In the following, the difference between M^{-1} and M_{L}^{-1} is not considered. Lumped mass applications are discussed in Section 7 and the computations in Section 8.

5. SPLITTING ERRORS

If the errors introduced by the splitting are of the same order as the errors inherent in the solution of the coupled system, determined by the time-integration scheme, then $[\mathbf{u}^{n+1}, p^{n+1}]^{\mathrm{T}}$ computed by (10) converge in time as fast as if computed by (6).

The perturbation due to the approximate LU factorization (the splitting) is

$$E = \hat{A} - A = \begin{bmatrix} C & CH_2G \\ D & D(H_2 - H_1)G \end{bmatrix} - \begin{bmatrix} C & G \\ D & 0 \end{bmatrix} = \begin{bmatrix} 0 & E_{\mathrm{m}} \\ 0 & E_{\mathrm{c}} \end{bmatrix},$$

where

$$E_{\mathrm{m}} = (CH_2 - I)G \quad \text{and} \quad E_{\mathrm{c}} = D(H_2 - H_1)G.$$

To simplify the notation in this section, write the system (6) as $Ax = b$ and the split system (10) as $\hat{A}\hat{x} = \hat{b}$.

For the purpose of finding the time convergence order, we may regard $\widehat{\delta^l p}^{n+1}$ as known and write the system at time step $n + 1$ as

$$A\hat{x} - b = -E\hat{x} = -\begin{bmatrix} E_{\mathrm{m}}\widehat{\delta^l p}^{n+1} \\ E_{\mathrm{c}}\widehat{\delta^l p}^{n+1} \end{bmatrix}. \tag{14}$$

Errors at previous time steps are disregarded; hence $\hat{b} = b$. Note the importance of the pressure (increment) variable on the splitting error. It can be reduced by diminishing the pressure variable.

Using the sensitivity analysis of Ref. [4, Sect. 2.7], it follows that the relative splitting error is

$$\frac{\|\hat{x} - x\|}{\|x\|} \le \kappa(A)\frac{\|\epsilon f\|}{\|b\|}.$$

Here, $\kappa(A)$ is the condition number of A, and ϵf is the perturbation $E\hat{x}$ such that ϵ is a power of δt.

For a kth-order time-integration method, we want $\epsilon = \delta t^k$. (We do not need $\epsilon = \delta t^{k+1}$ since the block matrix C incorporates a factor δt^{-1}.) We look at two special cases related to

methods presented in [11, 14], respectively. In case 1 we set $H_1 = H_2 = \delta t M^{-1}$; in case 2 we set $H_1 = \delta t M^{-1}$ and $H_2 = C^{-1}$.

Case 1. We have $E_c = 0$, and $E_m = \delta t S M^{-1} G$. The splitting (10) gives rise to a perturbation

$$E\hat{x} = \begin{bmatrix} \delta t S M^{-1} G \widehat{\delta^l p^{n+1}} \\ 0 \end{bmatrix}.$$

It follows from Taylor's theorem that $\delta^l p^{n+1} \approx \delta t^l (\partial^l p / \partial t^l)|_{t_{n+1}}$, and we may write

$$\|E\hat{x}\| \le \delta t^{l+1} \alpha \|S M^{-1} G\|.$$

The number α depends on $\partial^l p / \partial t^l|_{t_{n+1}}$. In fact, since derivatives are bounded in any meaningful incompressible flow, α may be considered constant. The matrix S depends on $\partial \mathbf{u}^{n+1} / \partial \mathbf{x}$ when the chosen time-integration scheme uses linearized convection. Again, it is reasonable to assume that an upper bound exists. We arrive at the following proposition.

PROPOSITION 5.1. *If $H_1 = H_2 = \delta t M^{-1}$, there exists a number $\mathcal{M}_1$ depending only on the boundary conditions and the spatial discretization such that*

$$\frac{\|\hat{x} - x\|}{\|x\|} \le \delta t^{l+1} \mathcal{M}_1.$$

The relative splitting error of (10), introduced when stepping from n to $n+1$, is thus $\mathcal{O}(\delta t^{l+1})$.

Case 2. We have $E_m = 0$ and $E_c = D(C^{-1} - \delta t M^{-1}) G$, and so the perturbation in this case is

$$E\hat{x} = \begin{bmatrix} 0 \\ D(C^{-1} - \delta t M^{-1}) G \widehat{\delta^l p^{n+1}} \end{bmatrix}. \tag{15}$$

Considering the von Neumann series of C^{-1} (12), it is clear that

$$E_c = \delta t^2 D M^{-1} S M^{-1} G + \mathcal{O}(\delta t^3).$$

In actual computations δt is restricted to an interval. On this interval, the von Neumann series may not converge. Then the order of vanishing of E_c is less than 2 but greater than 1.

It is possible to bound $\|C^{-1}\|$. Indeed, $\|C^{-1}\| \le c\|S^{-1}\|$ for large δt, and $\|C^{-1}\| \le c\delta t \|M^{-1}\|$ for small δt, with c being a suitable constant. Restricting our attention to $\delta t < 1$, we have

$$\|C^{-1}\| \le c(\|M^{-1}\| + \|S^{-1}\|).$$

The other matrices involved in (15) are easily bounded, and by arguing as for case 1 above, we conclude:

PROPOSITION 5.2. *If $H_1 = \delta t M^{-1}$ and $H_2 = C^{-1}$, there exists a number $\mathcal{M}_2$ depending only on boundary conditions and the spatial discretization such that*

$$\frac{\|\hat{x} - x\|}{\|x\|} \leq \delta t^{l+\gamma} \mathcal{M}_2, \quad 1 < \gamma \leq 2.$$

The number γ depends on the distance between δt and $\rho(M^{-1}S)^{-1}$.

The splitting error in this case is $\mathcal{O}(\delta t^{l+2})$ (the $\mathcal{O}$-notation involves the limit $\delta t \to 0$). When the time step is not sufficiently small, one will observe convergence rates at reduced order.

This analysis indicates that if other factors are kept equal, case 2 would show the faster convergence. But the analysis does not state anything about the numbers $\mathcal{M}_1$ and $\mathcal{M}_2$, or how close γ is to 2, and so cannot be used to conclude that either case should be generally preferred to the other. In the eyes of the authors, the most interesting result is that by solving for an adequately high-order backward difference pressure, the full time convergence of the unsplit scheme can be recaptured.

6. AN ALTERNATIVE APPROACH

The same increase in accuracy can also be obtained *without* solving for the pressure increment: Simply add $[CH_2G\sigma^l p^n, D(H_2 - H_1)G\sigma^l p^n]^{\mathrm{T}}$ to both sides of Eq. (9). Such a scheme is actually an example of a general technique—first make an approximation, and then compensate for the error. The following discussion takes this view and sheds a different light on approximate LU factorizations.

As shown in Section 5, the algebraic splitting perturbs the equations into

$$\begin{bmatrix} C & G \\ D & 0 \end{bmatrix} \begin{bmatrix} \mathbf{u}^{n+1} \\ p^{n+1} \end{bmatrix} = \begin{bmatrix} \mathbf{r}_u^n \\ 0 \end{bmatrix} + \begin{bmatrix} \mathbf{b}_u^{n+1} \\ b_p^{n+1} \end{bmatrix} - \begin{bmatrix} E_{\mathrm{m}}\, p^{n+1} \\ E_{\mathrm{c}}\, p^{n+1} \end{bmatrix}.$$

With a good guess for p^{n+1}, we can add a term to the right-hand side to compensate for the splitting error. The guess might be found from extrapolation of former pressure values by a difference formula: $p^{n+1} \approx \sigma^l p^n = p^{n+1} - \delta^l p^{n+1}$. The error-corrected equations become

$$\begin{bmatrix} C & G \\ D & 0 \end{bmatrix} \begin{bmatrix} \mathbf{u}^{n+1} \\ p^{n+1} \end{bmatrix} = \begin{bmatrix} \mathbf{r}_u^n \\ 0 \end{bmatrix} + \begin{bmatrix} \mathbf{b}_u^{n+1} \\ b_p^{n+1} \end{bmatrix} - \begin{bmatrix} E_{\mathrm{m}}\, \delta^l p^{n+1} \\ E_{\mathrm{c}}\, \delta^l p^{n+1} \end{bmatrix}, \tag{16}$$

the same equations as (14), which we got by applying algebraic splitting to an incremental scheme.

The system (16) is equivalent to

$$\begin{aligned} &1. \qquad C\hat{\mathbf{u}}^{n+1} = \mathbf{r}_u^n + E_{\mathrm{m}}\sigma^l p^n + \mathbf{b}_u^{n+1}, \\ &2. \quad DH_1Gp^{n+1} = D\hat{\mathbf{u}}^{n+1} - E_{\mathrm{c}}\sigma^l p^n + b_p^{n+1}, \\ &3. \qquad \mathbf{u}^{n+1} = \hat{\mathbf{u}}^{n+1} - H_2 G p^{n+1}. \end{aligned} \tag{17}$$

Setting $\hat{\mathbf{u}}^{n+1} = \tilde{\mathbf{u}}^{n+1} - H_2 G\sigma^l p^n$, we get back procedure (10). Computation of E_{m} and E_{c} makes procedure (17) more expensive than procedure (10). The most preferable situation

is like case 1 above: $E_c = 0$, and since H_2 is diagonal, only two matrix-vector products are required for evaluating the correction term. Stability properties of (17) may be preferable since the pressure does not accumulate round-off errors.

7. A THIRD-ORDER SCHEME

To illustrate the effect of using incremental pressure, in the next section we solve the equations for $\delta^l p^{n+1}$, $l = 0, 1$, and 2 for both case 1 and case 2. To observe the splitting error, we then need a time-integration scheme which is at least as accurate and for this purpose we introduce a third-order semiimplicit scheme based on the backward differentiation formula (BDF) combined with a linearization of the convection term. Application of this type of schemes to the Navier–Stokes equations is examined by Peyret [12].

The differential equation $\partial u/\partial t = f(u)$ is interpolated according to

$$\delta t^{-1} \sum_{j=0}^{3} a_j u^{n+1-j} = f(u^{n+1}), \tag{18}$$

with

$$a_0 = \frac{11}{6}, \quad a_1 = -\frac{18}{6}, \quad a_2 = \frac{9}{6}, \quad \text{and} \quad a_3 = -\frac{2}{6}.$$

Before applying Eq. (18) to the momentum equation, the convection term is linearized by a third-order extrapolation:

$$(\mathbf{u}^{n+1} \cdot \nabla)\mathbf{u}^{n+1} \approx (\mathbf{u}^* \cdot \nabla)\mathbf{u}^{n+1} = ((3\mathbf{u}^n - 3\mathbf{u}^{n-1} + \mathbf{u}^{n-2}) \cdot \nabla)\mathbf{u}^{n+1}.$$

This yields the semidiscretized system

$$\frac{11\mathbf{u}^{n+1} - 18\mathbf{u}^n + 9\mathbf{u}^{n-1} - 2\mathbf{u}^{n-2}}{6\delta t} + \nabla p^{n+1} = -(\mathbf{u}^* \cdot \nabla)\mathbf{u}^{n+1} + \nu \Delta \mathbf{u}^{n+1},$$
$$\nabla \cdot \mathbf{u}^{n+1} = 0.$$

Use of the approximate algebraic splitting method with $H_1 = H_2 = (6/11)\delta t M_L^{-1}$ (case 1) and pressure incremental form gives the procedure

$$\begin{aligned} &1. \quad C\tilde{\mathbf{u}}^{n+1} = \mathbf{r}_u^n - G\sigma^l p^n + \mathbf{b}_u^{n+1}, \\ &2. \quad \frac{6}{11}\delta t D M_L^{-1} G \delta^l p^{n+1} = D\tilde{\mathbf{u}}^{n+1} + b_p^{n+1}, \\ &3. \quad \mathbf{u}^{n+1} = \tilde{\mathbf{u}}^{n+1} - \frac{6}{11}\delta t M_L^{-1} G \delta^l p^{n+1}, \end{aligned} \tag{19}$$

where

$$\mathbf{r}_u^n = \frac{1}{\delta t} M \left(\frac{18}{6}\mathbf{u}^n - \frac{9}{6}\mathbf{u}^{n-1} + \frac{2}{6}\mathbf{u}^{n-1} \right) \quad \text{and} \quad C = \frac{11}{6\delta t} M + K(\mathbf{u}^*) + \nu L.$$

This choice of matrix approximations follows [11] and produces a weakly solenoidal solution.

It should be mentioned that procedure (19) with $l = 0, 1, 2$ resembles the procedures suggested by Gresho and Chan [5] called *Projection 1*, *Projection 2*, and *Projection 3*. Besides using a first-order time integration, the main difference is a factor MM_L^{-1} in front of the pressure term in step 1 of (19). Gresho and Chan experienced stability problems for the incremental versions, also when omitting the factor MM_L^{-1}, and we suspect it is caused by using equal-order finite elements for pressure and velocity. Guermond and Quartapelle [6] examined the stability for a (continuous) projection method and found that, while the nonincremental scheme is stable for equal-order elements (and sufficiently large time steps), elements fulfilling the inf–sup condition is required to make the incremental scheme stable.

The other case we examine is the same time method with $H_2 = C^{-1}$ and $H_1 = (6/11)\delta t M_L^{-1}$ (case 2). This choice is equivalent to what is called the Yosida method in Quarteroni *et al.* [14] and changes the discrete continuity equation, while the momentum equation remains unaltered. Writing the equations with incremental pressure we get

$$\begin{aligned} &1. && C\tilde{\mathbf{u}}^{n+1} = \mathbf{r}_u^n - G\sigma^l p^n + \mathbf{b}_u^{n+1}, \\ &2. && \frac{6}{11}\delta t D M_{\mathrm{L}}^{-1} G\delta^l p^{n+1} = D\tilde{\mathbf{u}}^{n+1} + b_p^{n+1}, \\ &3. && C\mathbf{u}^{n+1} = C\tilde{\mathbf{u}}^{n+1} - G\delta^l p^{n+1}, \end{aligned} \tag{20}$$

where $\mathbf{r}_u^n$ and C are as for case 1.

8. NUMERICAL EXPERIMENTS

For the study of temporal convergence of the splitting error, we chose two test problems. The first is a version of the "Kim–Moin problem" [9] that was used in [14], and the second is lid-driven cavity flow.

The two dimensional Kim–Moin problem has the time-dependent solution

$$u_1 = -\cos(2\pi x)\sin(2\pi y)e^{-8\pi^2\nu t}, \tag{21}$$

$$u_2 = \sin(2\pi x)\cos(2\pi y)e^{-8\pi^2\nu t}, \tag{22}$$

$$p = -\frac{1}{4}(\cos(4\pi x) + \cos(4\pi y))e^{-16\pi^2\nu t} \tag{23}$$

over the domain $\Omega = (0.25, 1.25) \times (0.5, 1.5)$, $t \in (0, 1]$. Only the Dirichlet problem was considered, and boundary values $\mathbf{w}$ are immediate from (21) and (23). Both interior and boundary degrees of freedom were kept in the system, and the Dirichlet conditions are enforced on the system before solving. No boundary conditions were set on the pressure, except at one point (0.25; 0.5) where $p(t) = 0$. The time step δt was varied from 2^{-3} to 2^{-9} and we used a structured grid of 96^2 square elements: nine- and four-node Lagrangian elements for the velocity and pressure, respectively, which is known as a low-order Taylor–Hood element pair (cf. [15]).

An object-oriented computer program was implemented using the C++ library Diffpack [10]. The main classes were a simulator class, which handled the time stepping, a velocity class for steps one and three of the above procedures, and a pressure class for step two. Gridding, finite-element assembly, matrix algebra, preconditioning, and solution of linear equations were performed by routines from the library. All linear equations were solved with the Krylov method "BiCGStab" [16], which can be used for nonsymmetric matrices.

Incomplete LU factorization was used for the preconditioners. The convergence criterion was $\|r^n\|/\|b\| < 10^{-8}$, where the norm is the Cartesian norm, r^n is the residual after the nth iteration, and b is the righthand-side vector. The computationally most expensive operation was the solution of the Poisson-like pressure equation; 50–74 iterations were needed to satisfy the convergence criterion. Also, the computation of DH_1D^{T} ($D^{\mathrm{T}} = G$) took several minutes on our computer (Sun Sparc), but this product depends only on the grid and was performed once in a preprocessing stage. When the system is large (small elements), forming the product DH_1D^{T} is not feasible and one will have to use an iterative method combined with a suitable preconditioner.

Errors were measured by the norms

$$\|\mathbf{e}_u\| = \max_n \|\mathbf{u}^n - \mathbf{u}(n\delta t)\|_{L^2(\Omega)}, \tag{24}$$

$$\|e_p\| = \operatorname*{rms}_n \|p^n - p(n\delta t)\|_{L^2(\Omega)}, \tag{25}$$

where rms_n denotes the root mean square.

To examine the effect of the splitting error and the use of incremental pressure, we employed the BDF scheme of Section 7, since this time integration has sufficiently high order. We tried both case 1 and case 2 to see if there is any noticable difference between modifying the momentum equation and modifying the continuity equation.

The plots that follow show velocity and pressure errors, Eqs. (24) and (25), as functions of δt. A dotted, straight line is included in the left corners for reference on the convergence order.

Figures 2, 3, and 4 refer to case 1, with $l = 0, 1, 2$. Both pressure and velocity errors are close to the theoretical order of convergence. In Figs. 3 and 4, we see the effect of the spatial error; the pressure error reaches its minimum at about 8.8×10^{-5} and becomes $\mathcal{O}(1)$. Close to 2.3×10^{-6} the velocity error also reaches its minimum.

The only parameter that is varied in these plots is the order of the pressure increment l. Thus the three plots confirm that the splitting error is indeed proportional to $\delta^l p^{n+1}$.

In [11], Crank–Nicholson time integration with nonincremental algebraic splitting gives second order for the velocity and first order for the pressure. It is stated that due to limitations in the discrete representation of the continuous pressure equation, the discrete pressure

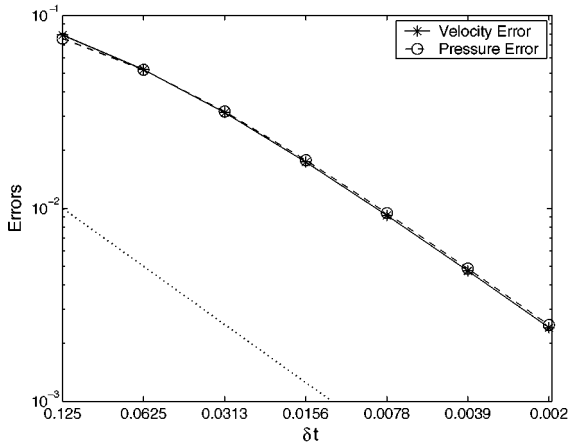

FIG. 2. Kim–Moin problem, case 1, $l = 0$.

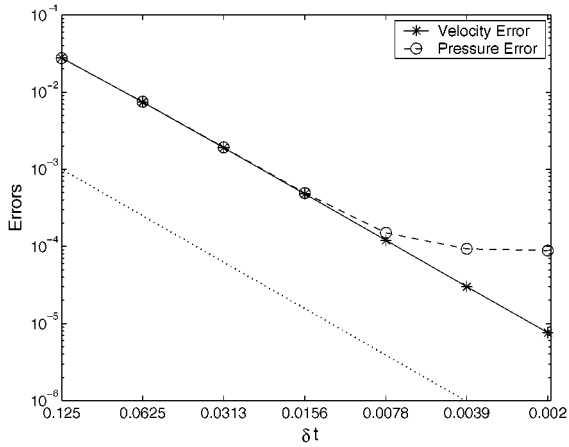

FIG. 3. Kim–Moin problem, case 1, $l = 1$.

will always be first-order accurate. The present study demonstrates that this is overcome by the use of incremental pressure. Third-order convergence of both velocity and pressure, as shown in Fig. 4, was previously obtained in [1]. In that study, second-order pressure increment and BDF time-integration are used with a projection method (modifies the continuous PDE). Spatial discretization is handled using a spectral method, and spatial errors are at the level of the machine precision.

Figures 5, 6, and 7 refer to case 2 for the Kim–Moin flow. With nonincremental pressure, the convergence orders of velocity and pressure are close to 1.8 and 1.3, respectively. With $l = 1$ (i.e., first-order pressure increment), we observe a convergence order of 2.8 for velocity (regarding only the first five values of δt) and a convergence order of 2.4 for pressure (first three values of δt). With $l = 2$ (see Fig. 7), the steepest parts of the graphs indicate the orders 3.8 and 3.4.

Compared to case 1, the velocity in case 2 converges faster and has higher accuracy for all values of l.

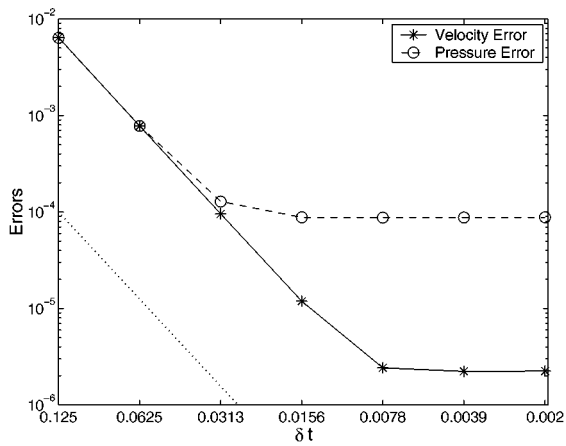

FIG. 4. Kim–Moin problem, case 1, $l = 2$.

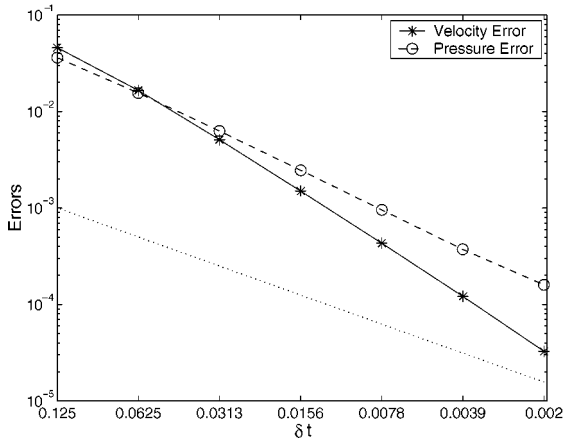

FIG. 5. Kim–Moin problem, case 2, $l = 0$.

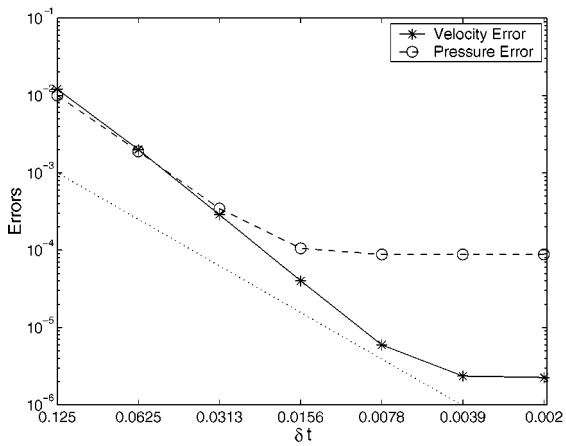

FIG. 6. Kim–Moin problem, case 2, $l = 1$.

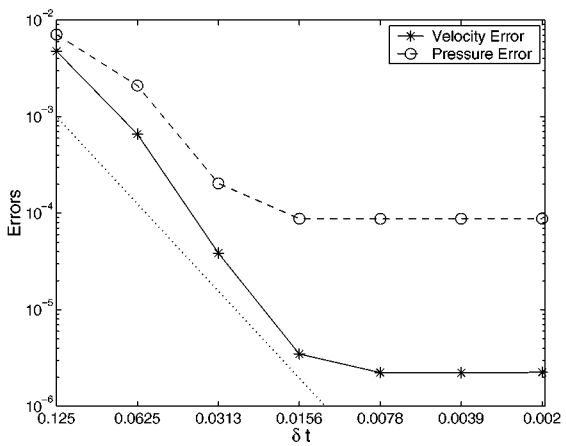

FIG. 7. Kim–Moin problem, case 2, $l = 2$.

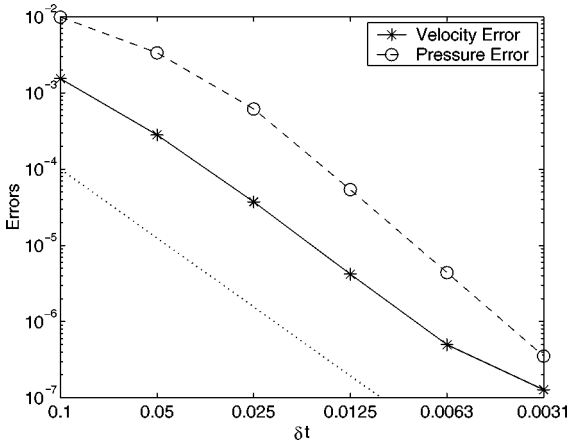

FIG. 8. Cavity flow, case 1, $l = 2$.

In the two-dimensional cavity problem, we used a unit box in which the lid started at rest and then was accelerated. This was done both to get a time-dependent problem and to avoid any possible problems with the inital values. The velocity function of the lid was

$$u_1 = \frac{(t/\tau)^4}{1 + (t/\tau)^4},$$

where $\tau = 0.6$ and $t \in (0, 1]$. Taylor–Hood elements (64^2 square) were used for discretizing this problem in space. A reference solution was obtained by running the same numerical method with a much smaller time step, $\delta t = 5 \times 10^{-4}$.

The splitting errors for case 1 and case 2 with $l = 2$ are plotted in Figs. 8 and 9. Again, the convergence rates for both pressure and velocity have the same order as the time-integration scheme, $\mathcal{O}(\delta t^3)$. For this problem, however, case 1 gives higher accuracy than case 2. The same was observed for $l = 1$ and $l = 0$.

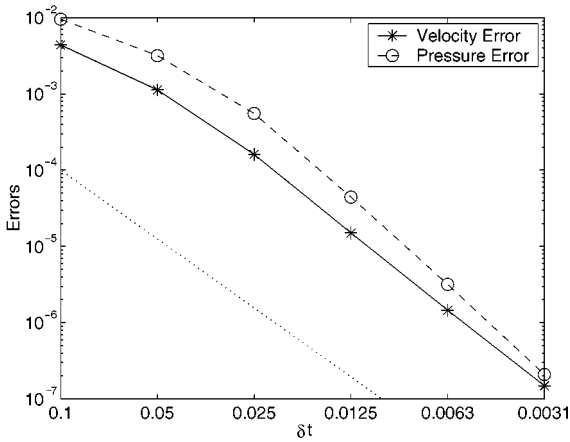

FIG. 9. Cavity flow, case 2, $l = 2$.

9. SUMMARY

Solving for incremental pressure, or correcting splitting errors, is a viable way to achieve higher order algebraic splitting methods. The Kim–Moin problem and the lid-driven cavity were used to test the effect of increasing the order of the pressure increment. It is clear from the analysis and the numerical experiments that the splitting error (in time) is proportional to the pressure variable. Use of pressure increments are thus an alternative—and cheaper in terms of computational work—to the use of high-order matrix factorizations (see [11]).

Two special cases of approximate factorizations were examined. They correspond to methods previously proposed by Perot [11] and Quarteroni *et al*. [14]. The method of the latter, which does not enforce mass conservation, has a somewhat faster time convergence but is not necessarily the most accurate on a specific range of time steps; this is indicated by both analysis and numerical experiments.

REFERENCES

1. O. Botella, On the solution of the Navier–Stokes equations using Chebyshev projection schemes with third-order accuracy in time, *Comput. Fluids* **26**, 107 (1997).
2. A. J. Chorin, Numerical solution of the Navier–Stokes equations, in *Computational Fluid Mechanics: Selected Papers* (Academic, San Diego, 1989), p. 17.
3. J. K. Dukowicz and A. S. Dvinsky, Approximate factorization as a high order splitting for the implicit incompressible flow equations, *J. Comput. Phys.* **102**, 336 (1992).
4. G. H. Golub and C. F. van Loan, *Matrix Computations* 3rd ed. (Johns Hopkins Univ. Press, Baltimore, 1996).
5. P. M. Gresho and S. T. Chan, On the theory of semi-implicit projection methods for viscous incompressible flow and its emplementation via a finite element method that also introduces a nearly consistent mass matrix. Part 2: Implementation, *Int. J. Numer. Methods Fluids* **11**, 621 (1990).
6. J. L. Guermond and L. Quartapelle, On stability and convergence of projection methods based on pressure Poisson equation, *Int. J. Numer. Methods Fluids* **26**, 1039 (1998).
7. C. W. Hirt and J. L. Cook, Calculating three-dimensional flows around structures and over rough terrain, *J. Comput. Phys.* **10**, 324 (1972).
8. G. E. Karniadakis, M. Israeli, and S. A. Orzag, High-order splitting methods for the incompressible Navier–Stokes equations, *J. Comput. Phys.* **97**, 414 (1991).
9. J. Kim and P. Moin, Application of a fractional-step method to incompressible Navier–Stokes equations *J. Comput. Phys.* **59**, 308 (1985).
10. H. P. Langtangen, *Computational Partial Differential Equations* (Springer-Verlag, New York/Berlin, 1999).
11. J. B. Perot, An analysis of the fractional step method, *J. Comput. Phys.* **108**, 51 (1993), doi:10.1006/jcph.1993.1162.
12. R. Peyret, Introduction to high-order approximation methods for computational fluid dynamics, in *Advanced Turbulent Flow Computations*, edited by R. Peyret and E. Krause, CISM Courses and Lectures (Springer-Verlag, Wien/New York, 2000), Vol. 395, p. 1.
13. L. Quartapelle, *Numerical Solution of the Incompressible Navier–Stokes Equations* (Birkhäuser Verlag, Basel, 1993).
14. A. Quarteroni, F. Saleri, and A. Veneziani, Factorization methods for the numerical approximation of Navier–Stokes equations, *Comput. Methods Appl. Mech. Eng.* **188**, 505 (2000).
15. A. Quarteroni and A. Valli, *Numerical Approximation of Partial Differential Equations* (Springer-Verlag, New York/Berlin, 1994).
16. Y. Saad, *Iterative Methods for Sparse Linear Systems* (PWS Publ. Co. Boston, 1996).
17. R. Temam, Sur l'approximation de la solution des équation de Navier–Stokes par la méthode de pas fractionnaires (II), *Arch. Ration. Mech. Anal.* **33**, 377 (1969).