JOURNAL OF COMPUTATIONAL PHYSICS 34, 54–73 (1980)

On the Structure and Use of Linearized Block Implicit Schemes

W. R. BRILEY AND H. MCDONALD

Scientific Research Associates, Inc., P.O. Box 498, Glastonbury, Connecticut 06033

Received July 5, 1978

The recent use of methods which may be termed "linearized block ADI methods" or more generally "consistently split linearized block implicit" methods has been a significant development in the efficient and noniterative solution of certain systems of coupled nonlinear multidimensional partial differential equations. Some observations on their structure, derivation, and use are given. Consistently split linearized block implicit (LBI) methods are unified here and are related to the earlier scalar ADI schemes, as well as to existing iterative and noniterative methods for solving both systems of nonlinear algebraic equations, and systems of nonlinear ordinary differential equations (including those having multipoint boundary conditions). It is shown that the method used by Lindemuth and Killeen and that of Briley and McDonald (utilizing a two-dimensional Crank–Nicolson formulation) are both consistently split block implicit schemes which differ in principle only with regard to implementation of the linearization technique. It is also observed that the first approximate factorization scheme of Beam and Warming utilizes a splitting due to D'Yakonov whose intermediate steps are inconsistent in the sense that they do not approximate the governing equations to within a truncation error which vanishes to some order for small Δt. Methods based on splittings which have inconsistent intermediate steps are placed in a separate category and are shown to present serious difficulties, which apparently have escaped notice, in treating derivative boundary conditions accurately. Although similar difficulties can arise in the transient with consistently split schemes, the consistent splitting normally provides one order of accuracy improvement. It is further demonstrated that the two-level version of the second and more recent "delta" form approximate factorization scheme of Warming and Beam and the earlier method of Briley and McDonald have identical linearized block implicit structures. Finally, further substantial gains in efficiency resulting from reducible block submatrices and the use of multiple time steps are described.

INTRODUCTION

In independent investigations, Lindemuth and Killeen [1] and Briley and McDonald [2] devised closely related schemes capable of obtaining numerical solutions to systems of nonlinear multidimensional partial differential equations (PDE's). These schemes were applied to a two-dimensional magnetohydrodynamic problem [1] and to the three-dimensional compressible Navier–Stokes equations [2, 4]. McDonald and Briley also employed their method to compute three-dimensional viscous supersonic flow by forward marching integration [3]. The methods of [1–4] both combine a formal linearization technique (Taylor series expansion in time) and utilize ADI schemes originally formulated for scalar equations, in their natural extension as

0021-9991/80/010054–20$02.00/0

consistently split block implicit schemes for linear systems of equations. The result is a highly efficient and stable noniterative method for solving nonlinear multidimensional systems of PDE's, wherein the formal error due to linearization is no larger than that introduced by temporal discretization. The resulting algorithm requires solution of block tridiagonal systems (for example) as opposed to simple tridiagonal systems obtained for scalar equations. Beam and Warming [5–7] have also developed two related methods based on D'Yakonov's approximate factorization approach. The first method [5] was derived for nonlinear systems of first-order equations in conservation-law form. In this special case, the method combines either standard or high-order "compact" Padé spatial difference formulas with a split "factored" algorithm. Warming and Beam [6–7] later reformulated this method as a "delta" form factorization and added second-order (viscous) terms. They also derived a three-level version of this latter method which is highly efficient in its storage utilization, requiring only two levels of storage even though it is a three-level scheme. Beam and Warming [7] have applied the "delta" form factored scheme to the two-dimensional compressible Navier–Stokes equations. In an early development, Gourlay and Morris [8] proposed block ADI methods for nonlinear hyperbolic systems in either conservation or nonconservation form, utilizing a two-step explicit-predictor, block-ADI-corrector formulation to accomplish linearization. Considerations of efficiency and coding complexity appear to favor one-step linearized block implicit methods over the predictor-corrector approach. Here, the methods of [1–7] are considered specifically as linearized block implicit methods, and salient features of the different methods are discussed without a detailed formal derivation, as this would be lengthy, would tend to obscure the points being made, and in any event derivations are given in the cited references. Omitted here but considered in detail by other authors elsewhere are the topics of stability and error properties [5, 6], spatial differencing [5, 6, 9], special treatments for mixed derivatives [1, 6, 7], choice of dependent variables, and other details which may vary from application to application.

Systems of governing equations encountered in applications may contain (for example) terms such as $\mathscr{A}(\phi)\,\partial F(\phi)/\partial x$, where $\mathscr{A}$ is a square matrix whose elements are functions of ϕ, a column vector of dependent variables, and F is a column vector function of ϕ. The development of Lindemuth and Killeen [1] considers primarily the quasilinear form ($F = \phi$); that of Briley and McDonald [2–4] treats conservative ($\mathscr{A} = I$, the identity matrix), quasilinear, and mixed forms. Lindemuth [9] gives a detailed treatment of spatial differencing techniques for both conservative and nonconservative forms. The development of Beam and Warming [5] considered first-order conservation forms and later [6, 7] conservation forms of mixed order were added. Here, we employ notation similar to that of Briley and McDonald [4], which formalizes the method first reported in [2], and use a differential-difference operator notation. The operator notation is quite general and permits the simultaneous and unambiguous treatment of equations having either conservation (divergence) form, quasilinear form, or various mixtures. Any one of these forms can have distinct advantages depending on the particular application. For example, for the special case of first-order hyperbolic systems in conservation form, Beam and Warming [5] were able to utilize Padé fourth-order central and second-order one-sided spatial differences while retaining a block tridiagonal matrix structure. The recent operator-compact implicit differencing of Ciment, Leventhal, and Weinberg [10] retains tridiagonal matrices in the presence of mixed-order derivatives and may be useful in more general circumstances.

Linearization

The technique of linearization by Taylor expansion used in [1–7] is well known as an integral part of the Newton–Raphson method (cf. Ralston [11]) for iterative solution of systems of nonlinear algebraic equations. Keller [12] describes the use of Newton's method for the iterative solution of nonlinear systems of first- and second-order ordinary differential equations (ODE's) with two-point boundary conditions. Bellman and Kalaba [13] advocate the "quasilinearization" (Newton–Raphson) method for the iterative solution of implicit nonlinear difference approximations for PDE's. A different emphasis is present here and in [1–7], however, and that is to pose an initial value problem and linearize by Taylor expansion in time (or a timelike variable) rather than in some iteration space.

The difference in viewpoint can be illustrated as follows: If as before, $\boldsymbol{F}(\boldsymbol{\phi})$ is a (nonlinear) vector function of the vector $\boldsymbol{\phi}$, and if t^n denotes a discretized time variable such that $\Delta t = t^{n+1} - t^n$, then the linearization of $\boldsymbol{F}^{n+1}$ in time is given by

$$\boldsymbol{F}^{n+1} = \boldsymbol{F}^n + (\partial \boldsymbol{F}/\partial \boldsymbol{\phi})^n(\boldsymbol{\phi}^{n+1} - \boldsymbol{\phi}^n) + O(\Delta t^2). \tag{1}$$

Note that one form (cf. Keller [12]) of the Newton–Raphson method (with iteration index n) for solving the system $\boldsymbol{F}(\boldsymbol{\phi}) = 0$ is obtained by setting $\boldsymbol{F}^{n+1} = 0$ in Eq. (1). On the other hand, if Eq. (1) is combined with the implicit time difference approximation

$$(\boldsymbol{\phi}^{n+1} - \boldsymbol{\phi}^n) = \Delta t[\beta \boldsymbol{F}^{n+1} + (1-\beta)\,\boldsymbol{F}^n] \tag{2}$$

(centered about $t^n + \beta\,\Delta t$; $0 \leqslant \beta \leqslant 1$) for the system of ODE's

$$\partial \boldsymbol{\phi}/\partial t = \boldsymbol{F}(\boldsymbol{\phi}), \tag{3}$$

the result is

$$[I/\Delta t - \beta(\partial \boldsymbol{F}/\partial \boldsymbol{\phi})^n](\boldsymbol{\phi}^{n+1} - \boldsymbol{\phi}^n) = \boldsymbol{F}^n \tag{4}$$

and a stable noniterative implicit method for the system of ODE's (3) (posed as an initial value problem) is obtained. The use of Eq. (4) requires solution of a linear system having a square matrix, at each time step. The noniterative scheme (4) falls within the general approach described by Lomax [14] for solution of "stiff" systems of ODE's, where stiffness is associated with widely differing eigenvalues of the Jacobian matrix $\partial \boldsymbol{F}/\partial \boldsymbol{\phi}$. Steady solutions of Eq. (4) also satisfy $\boldsymbol{F}(\boldsymbol{\phi}) = 0$, but the Newton–

Raphson method is obtained from Eq. (4) only in the limit as $\Delta t \to \infty$, $\beta = 1$. Alternatively, Newton–Raphson (or quasilinearization) iteration can be used to solve Eq. (2) directly, which leads to the recursive formula

$$(\phi^{k+1} - \phi^n) = \Delta t\{\beta[F^k + (\partial F/\partial\phi)^k(\phi^{k+1} - \phi^k)] + (1 - \beta)\, F^n\}, \quad k = 0, 1, 2, \ldots . \tag{5}$$

Note that the noniterative scheme (4) can be derived from Eq. (5) by taking $\phi^{k=0} = \phi^n$ and performing one iteration.

The attraction of the noniterative time-linearization formulation as opposed to that of Newton–Raphson or quasilinearization is that it leads naturally to the following observations: if transient accuracy is an objective in the solution of the system of ODE's (3), then the errors arising from linearization as in Eq. (1) are expressed as temporal errors which may be compared with those of the temporal discretization (2). For example, it is easily shown (cf. McDonald and Briley [3]) that for $\beta = 0.5$ the scalar form of Eq. (2) approximates Eq. (3) with a local time truncation error E_t of order $E_t = -(\Delta t)^2(\partial^3\phi/\partial t^3)/12$, whereas an additional nonlinear truncation error E_{nl} is present in the linearized form Eq. (4) and is of local order $E_{nl} = (\Delta t)^2(\partial^2 F/\partial\phi^2)(\partial\phi/\partial t)^2/4$. Note that use of the linearization (1) does not lower the formal order of accuracy of the time discretization (2). Since each iteration of Eq. (5) generally requires at least as much effort as one time step using the noniterative scheme (4), the relative efficiency of the iterative and noniterative formulations can be clarified by comparing the reduction in composite error $E_t + E_{nl}$ obtained under the following alternatives: (a) perform two, three, or more iterations of Eq. (5), or instead (b) reduce the step size Δt by a factor of two, three, or more and repeat the noniterative scheme (4) a corresponding number of times in the given time interval. This comparison is shown in Table I. Note that for the same or less effort, reducing the step size Δt in Eq. (4) reduces *both* the linearization error and the temporal truncation error, and, although successive iterations of Eq. (5) are extremely effective in reducing the linearization error (assuming convergence), iteration leaves the temporal truncation error unaffected. Consequently, it appears that one iteration of Eq. (5) will usually be optimum and the simpler noniterative formulation of Eq. (4) preferred. Evidently, iteration can be more efficient than reducing the time step only in cases for which the linearization error is much larger than the temporal truncation error. From Table I

TABLE I

Computational Effort to Reduce Error in Solution of Scalar Form of Eq. (2)

Noniterative scheme (4), $\beta = \frac{1}{2}$			Iterative Scheme (5), $\beta = \frac{1}{2}$	
Time steps	Step size	Composite error	Iterations	Composite error
1	Δt	$E_t + E_{nl}$	1	$E_t + E_{nl}$
2	$\frac{1}{2}\Delta t$	$\frac{1}{4}(E_t + E_{nl})$	2	$E_t + O(E_{nl})^2$
3	$\frac{1}{3}\Delta t$	$\frac{1}{9}(E_t + E_{nl})$	3	$E_t + O(E_{nl})^4$

for example, assuming $(E_{nl})^2 \rightarrow 0$ after two iterations, and further assuming E_t and E_{nl} are additive rather than offsetting errors, then performing two time steps is still superior to iterating unless E_{nl} is more than three times as large as E_t. Nevertheless, it should be acknowledged that in some applications iteration may be desirable [Lindemuth, private communication]. Finally, it may be that the only objective in solving Eq. (3) is the steady solution $\boldsymbol{F}(\phi) = 0$. In such circumstances, each time step "iteration" computed using (4) is efficient, and although the quadratic convergence of the Newton–Raphson method (for "good" initial approximations) is recovered only as $\Delta t \rightarrow \infty$, $\beta = 1$, the time step is available as a free parameter which can be used (and even optimized) to both guarantee and accelerate convergence.

It should be emphasized that in the case of ODE's a wide variety of methods is available, including Runge–Kutta methods and various multistep iterated predictor-corrector methods (cf. [11, 15]). Generally, implicit methods are preferred for "stiff" systems of ODE's [14, 15]. The situation with regard to available schemes is not so favorable for the more demanding systems of multidimensional PDE's of primary interest here, however, and a noniterative time-dependent development analogous to that leading to Eq. (4) has considerable merit. Systems of parabolic–hyperbolic PDE's can be considered simply by adding a *homogeneous* vector spatial differential-difference operator $\mathscr{D}(\phi)$, which may be multidimensional, to $\boldsymbol{F}$ in Eq. (2) and repeating steps analogous to Eqs. (1)–(4). For example, in one spatial dimension x, if $\mathscr{D}$ has a quasilinear form such that

$$\mathscr{D}_x(\phi) \equiv \boldsymbol{G}\left(\phi, \frac{\partial \phi}{\partial x}, \frac{\partial^2 \phi}{\partial x}\right), \tag{6}$$

then $\mathscr{D}_x^{n+1} \equiv \mathscr{D}_x(\phi^{n+1})$ can be linearized in time as follows

$$\mathscr{D}_x^{n+1} = \mathscr{D}_x^{\,n} + \left\{\left(\frac{\partial \boldsymbol{G}}{\partial \phi}\right)^n + \left[\frac{\partial \boldsymbol{G}}{\partial(\partial \phi/\partial x)}\right]^n \frac{\partial}{\partial x} + \left[\frac{\partial \boldsymbol{G}}{\partial(\partial^2 \phi/\partial x^2)}\right]^n \frac{\partial^2}{\partial x^2}\right\} (\phi^{n+1} - \phi^n) + O(\Delta t^2). \tag{7}$$

If we denote the term in braces in Eq. (7) by the symbolic shorthand representation $(\partial \mathscr{D}_x/\partial \phi)^n$, then adding $\mathscr{D}_x$ to $\boldsymbol{F}$ in the time discretization (2) results in

$$(\phi^{n+1} - \phi^n) = \Delta t[\beta(\mathscr{D}_x + \boldsymbol{F})^{n+1} + (1 - \beta)(\mathscr{D}_x + \boldsymbol{F})^n] \tag{8a}$$

and linearizing in Eq. (1, 7) leads to

$$[\boldsymbol{I} - \beta\, \Delta t(\partial \mathscr{D}_x/\partial \phi)^n - \beta\, \Delta t(\partial \boldsymbol{F}/\partial \phi)^n](\phi^{n+1} - \phi^n) = \Delta t(\mathscr{D}_x + \boldsymbol{F})^n. \tag{8b}$$

If in addition, $\partial/\partial x$ and $\partial^2/\partial x^2$ are replaced by standard three-point difference operators, then Eq. (8b) becomes a noniterative method for systems of PDE's in one space dimension. The matrix resulting from Eq. (8b) has a block banded form (bandwidth equal to the size of the spatial difference molecule), and can be solved efficiently by block elimination [15]. The scheme (8b) for PDE's has generally the same desirable properties mentioned previously for scheme (4) for ODE's.

It is further emphasized that the implicit scheme (8b) represents a stable and efficient solution procedure not only in the presence of "stiffness" associated with F, as for ODE's, but also in the presence of perhaps analogous but more complex stiffness problems associated with the spatial operator $\mathscr{D}_x$. In the former case, F now amounts to a "nonlinear source term." In the latter case of $\mathscr{D}_x$, it is noted that conditionally stable methods usually are subject to stability restrictions depending not only upon Δt, but also upon the spatial mesh increment Δx. In the case of compressible aerodynamics, for example, these correspond to the well-known Courant–Friedrichs–Lewy (CFL) and viscous stability restrictions, which have the respective one-dimensional forms $\Delta t \leqslant \Delta x/(|u| + c)$ and $\Delta t \leqslant (\Delta x)^2/2\nu$, where u is velocity, c is the speed of sound, and ν is kinematic viscosity. It is noted that the CFL condition reflects different eigenvalues having the physical interpretation of the convection speed u and the relative speed of propagation of sound waves, $u \pm c$. The scalar analog of the CFL condition is $\Delta t \leqslant \Delta x/|u|$. The stability restriction on diffusion is inherently scalar, and since it involves $(\Delta x)^2$, it is rather severe. Implicit methods have long been accepted as among the most efficient means for solving diffusion equations. Finally, it is noted that steady solutions of Eq. (8b) satisfy the system of ODE's $\mathscr{D}_x(\phi) + F = 0$, which in this case may be subject to multipoint boundary conditions. Furthermore, in computing such steady solutions, stability and optimization of Δt are important, but transient accuracy is irrelevant.

The iterative use of linearization by Taylor expansion for solving coupled PDE's in one space dimension is well known. Keller's [17] box scheme employs Newton iteration for implicit solution of coupled systems of first-order PDE's in one space dimension and is widely used. Various successful implicit methods based on quasilinearization for solving the two-dimensional boundary layer equations (cf. Blottner [18] and references; Rubin and Khosla [19]) can be thought of as iterative analogs of Eq. (8b). Apparently, one of the first uses of a noniterative formulation as exemplified by Eq. (8b) was that of Richtmyer and Morton [20], who considered a scalar nonlinear diffusion equation in one space dimension. More recently, Kreskovsky and Shamroth [21] employed a slight generalization of Eq. (8b) to solve a reduced form of the two-dimensional Navier–Stokes equations by spatial forward marching integration. The resulting numerical method is noteworthy in that third-order differential equations are treated, and the internal flow application results in augmented block (4 × 4) pentadiagonal matrices. The multidimensional application of the noniterative scheme typified by Eq. (8b) is one of the key elements of the linearized block implicit (LBI) schemes of [1–7].

Consistently Split Linearized Block Implicit Methods

Multidimensional analogs of scheme (8b) can be derived by allowing $\mathscr{D}$ to be a multidimensional operator having, for example, the very general form

$$\mathscr{D}(\phi) = G\left(\phi, \frac{\partial}{\partial x}, \frac{\partial^2}{\partial x^2}, \frac{\partial}{\partial y}, \frac{\partial^2}{\partial y^2}, \frac{\partial}{\partial z}, \frac{\partial^2}{\partial z^2}, x, y, z, t\right). \tag{9}$$

In this case, however, the resulting system of linear implicit difference equations is not easily solved. On the other hand, if (block) ADI or other splitting techniques are introduced, narrow block-banded matrices can be recovered, and the result is a highly efficient and general class of implicit methods for systems of nonlinear multidimensional PDE's. To the authors' knowledge, the utility of the one-step noniterative implicit formulation for coupled nonlinear PDE's, as in Eq. (8b), and more importantly its multidimensional generalization using block ADI techniques, was not recognized until applied independently by Lindemuth and Killeen [1] and Briley and McDonald [2]. Finally, it is noted that in the multidimensional case where D has a form such as Eq. (9), steady solutions satisfy $\mathscr{D}(\phi) + F = 0$, which includes systems of nonlinear PDE's of mixed parabolic, hyperbolic, and elliptic classification. In this latter case, the steady solution is very likely to be the only item of interest in applying the multidimensional analog of Eq. (8b); hence, considerations of temporal accuracy are unimportant and the high single step efficiency of the noniterative formulation assumes a greatly enhanced role.

In deriving the multidimensional methods, use of the time linearization technique exemplified by Eqs. (1) and (7) at an appropriate point in the algorithm produces linear difference operators from nonlinear ones [4], as is necessary for efficient noniterative solution of implicit schemes. Viewed in this manner, implementation of the linearization can be discussed separately from the splitting techniques being used. As will be noted, Lindemuth and Killeen [1] implemented the linearization in a manner slightly different from that of Briley and McDonald [2]; however, we shall first *unify* the methods of [1–7] on the basis of their treatment of linear systems of equations, as this parallels existing ADI literature.

Consider an implicit difference scheme of the following form, which may be spatially nonlinear:

$$(\phi^{n+1} - \phi^n) = \Delta t[\mathscr{D}_x + \mathscr{D}_y + \mathscr{D}_z][\beta\phi^{n+1} + (1 - \beta)\phi^n]. \tag{10}$$

Here, $\mathscr{D}_x$, $\mathscr{D}_y$, $\mathscr{D}_z$ can represent either vector differential or vector difference operators associated with (for example) x, y, z coordinate directions. Although here as in most applications the component $\mathscr{D}$ operators are presumed associated with coordinate directions (hence the "alternating direction" terminology), it should be emphasized that the original ADI concept was generalized by Douglas and Gunn [22] to include any number of component operators which need not be associated with coordinate directions or even directions within the computational lattice. The key requirement for application of ADI or splitting techniques in general is that component operators $\mathscr{D}_i$ ($\mathscr{D} \equiv \sum_i \mathscr{D}_i$) be identified whose associated matrices are "easily solved" [22]. As before, β permits a variable centering in time, and although multitime levels are easily accommodated in the development, a two-time-level scheme is supposed. Further, although a nonlinear time derivative such as $\partial \boldsymbol{H}(\phi)/\partial t$ is allowed for in the development given in [2–4], Eq. (10) is limited for simplicity to the linear case $\boldsymbol{H} \equiv \phi$. Nevertheless, the generality afforded by retention of a time derivative term for which $\partial \boldsymbol{H}/\partial \phi \neq \boldsymbol{I}$ is worth noting. The assumption $\boldsymbol{H} \equiv \phi$ may require a particular form for the governing equations and/or a particular choice of dependent variables. Finally, for the moment attention is restricted to the case in which the $\mathscr{D}$-operators are linear transformations and hence homogeneous.

The derivation of block ADI schemes can proceed in several ways. The very general procedure of Douglas and Gunn [22] for deriving ADI schemes as perturbations of fundamental implicit schemes was developed in a scalar context. Nevertheless, their technique for generating ADI schemes employs linear operators and is thus applicable to systems of equations. Consequently, the Douglas–Gunn procedure was applied by the authors [2–4] to (linearized) systems such as Eq. (10), which may contain mixed parabolic and first-order hyperbolic forms. Here as in [2–4], the Douglas–Gunn procedure is viewed as a general splitting technique whose intermediate steps have the very desirable property of being consistent. The Douglas–Gunn splitting of Eq. (10) as employed in [2, 4] is given by

$$(\phi^{*} - \phi^{n})/\Delta t = \beta\mathscr{D}_x\phi^{*} + [(1 - \beta)\mathscr{D}_x + \mathscr{D}_y + \mathscr{D}_z]\,\phi^{n}, \tag{11a}$$

$$(\phi^{**} - \phi^{n})/\Delta t = \beta\mathscr{D}_x\phi^{*} + \beta\mathscr{D}_y\phi^{**} + [(1 - \beta)(\mathscr{D}_x + \mathscr{D}_y) + \mathscr{D}_z]\,\phi^{n}, \tag{11b}$$

$$(\phi^{***} - \phi^{n})/\Delta t = \beta\mathscr{D}_x\phi^{*} + \beta\mathscr{D}_y\phi^{**} + \beta\mathscr{D}_z\phi^{***} + [(1 - \beta)(\mathscr{D}_x + \mathscr{D}_y + \mathscr{D}_z)]\,\phi^{n}, \tag{11c}$$

$$\phi^{n+1} = \phi^{***} + O(\Delta t^{3}). \tag{11d}$$

Note that each successive step treats one more $\mathscr{D}$-operator implicitly. Although Eqs. (11) are not in the form most desirable for coding purposes, the form of Eqs. (11) is a critical step in the Douglas–Gunn splitting process since it is apparent that each of the intermediate steps (11a)–(11c) is a consistent approximation of Eq. (10) and that each of the intermediate solutions ϕ^{*}, ϕ^{**}, ϕ^{***} approximates ϕ^{n+1}. Thus if the original scheme (10) is consistent in the sense that it approximates the governing equations to within a truncation error which vanishes to some order for small Δt, then each step in the split form (11) is also consistent. For coding purposes, Douglas and Gunn [22] recommend the following simplified form of Eqs. (11b) and (11c) obtained by subtracting (11a) from (11b) and (11b) from (11c):

$$\phi^{**} - \phi^{*} = \Delta t\beta\mathscr{D}_y(\phi^{**} - \phi^{n}), \tag{12a}$$

$$\phi^{***} - \phi^{**} = \Delta t\beta\mathscr{D}_z(\phi^{***} - \phi^{n}). \tag{12b}$$

For the purpose of analysis, Douglas and Gunn rewrite Eqs. (11a), (12a), and (12b) as

$$(I - \beta\,\Delta t\mathscr{D}_x)(\phi^{*} - \phi^{n}) = \Delta t(\mathscr{D}_x + \mathscr{D}_y + \mathscr{D}_z)\,\phi^{n}, \tag{13a}$$

$$(I - \beta\,\Delta t\mathscr{D}_y)(\phi^{**} - \phi^{n}) = \phi^{*} - \phi^{n}, \tag{13b}$$

$$(I - \beta\,\Delta t\mathscr{D}_z)(\phi^{***} - \phi^{n}) = \phi^{**} - \phi^{n}, \tag{13c}$$

$$\phi^{n+1} = \phi^{***} + O(\Delta t^{3}). \tag{13d}$$

The present authors prefer to code in terms of Eqs. (13), solving for the temporal increment $\psi \equiv \phi - \phi^n$, primarily because of apparent improvement in round-off characteristics. Specifically, when all quantities except ϕ are exact, the use of Eqs. (13) permits the computation of steady solutions to an arbitrary number of significant figures, whereas (11) and (12) are limited to significant figures carried by the computer minus those significant figures lost in the solution of block tridiagonal systems. The round-off errors are reduced by solving for $O(\Delta t)$ quantities such as the difference $(\phi^* - \phi^n)$ and correcting ϕ^n, rather than solving directly for $O(1)$ quantities such as ϕ^*. Nevertheless, Eqs. (11)–(13) are just different forms of the same scheme.

Eliminating the intermediate steps in Eqs. (13a)–(13d) results in

$$\begin{aligned}(I - \beta\,\Delta t\mathscr{D}_x)(I - \beta\,\Delta t\mathscr{D}_y)(I - \beta\,\Delta t\mathscr{D}_z)(\phi^{n+1} - \phi^n)\\ = \Delta t(\mathscr{D}_x + \mathscr{D}_y + \mathscr{D}_z)\,\phi^n + O(\Delta t^3)\end{aligned} \tag{14}$$

which is a combined form of the splitting (11) and (12) or (13) and also an approximate factorization of Eq. (10), since upon multiplying the factors in Eq. (14) and noting that $\phi^{n+1} - \phi^n = O(\Delta t)$ the result differs from Eq. (10) only by terms of $O(\Delta t)^3$ or higher. A change in the sequence of steps from x, y, z to some other ordering such as y, z, x is equivalent to commuting the factors in Eq. (14). Although these factors are not in general commutative, the differences in the resulting schemes remain $O(\Delta t)^3$.

Lindemuth and Killeen [1] considered a problem in two space dimensions and for their system of PDE's followed the procedure originally used by Peaceman and Rachford [23] to obtain the first (scalar) ADI scheme. Lindemuth and Killeen [1] were thus led to the following scheme:

$$\left(I - \frac{\Delta t}{2}\mathscr{D}_x\right)\hat{\phi} = \left(I + \frac{\Delta t}{2}\mathscr{D}_y\right)\phi^n, \tag{15a}$$

$$\left(I - \frac{\Delta t}{2}\mathscr{D}_y\right)\phi^{**} = \left(I + \frac{\Delta t}{2}\mathscr{D}_x\right)\hat{\phi}, \tag{15b}$$

$$\phi^{n+1} = \phi^{**} + O(\Delta t^3). \tag{15c}$$

Note, however, that if the change of variables $\hat{\phi} = (\phi^* + \phi^n)/2$ is made in Eqs. (15) and if Eq. (15a) is subtracted from (15b), then Eqs. (15a) and (15b) can be written as

$$\phi^* - \phi^n = \Delta t\left[\mathscr{D}_x\left(\frac{\phi^* + \phi^n}{2}\right) + \mathscr{D}_y\phi^n\right], \tag{16a}$$

$$\phi^{**} - \phi^* = \frac{\Delta t}{2}\mathscr{D}_y(\phi^{**} - \phi^n) \tag{16b}$$

which is a special case of Eqs. (11a) and (12a), corresponding to $\beta = \frac{1}{2}$, $\mathscr{D}_z = 0$. Thus, as in the case of a scalar equation, the Peaceman–Rachford block ADI scheme for systems of equations is recovered in disguised form as the Douglas–Gunn splitting of the two-dimensional Crank–Nicolson scheme. Scheme (11) provides the unambiguous generalization of the consistently split Crank–Nicolson scheme to three space dimensions, a process which in the earlier development of scalar ADI methods proved elusive. The general formulation of Douglas and Gunn [22] also includes as special cases earlier scalar ADI schemes developed by Douglas and Rachford (1956), Douglas (1961), and Douglas and Gunn (1963), but does not include the scheme of D'Yakonov or the so-called LOD or fractional step schemes (cf. [24]).

Linearization of the $\mathscr{D}$-Operators

Although quasilinear $\mathscr{D}$-operators of a form such as Eq. (6) are not of sufficient generality to cover many applications, the $\mathscr{D}$-operators, if nonlinear, can be linearized by Taylor expansion in time as necessary [4]. In the construction of generalized algorithms, it is convenient to introduce a notation which represents a linearized approximation to $\mathscr{D}(\phi^{n+1})$ obtained by expansion in time about a (known) solution ϕ^* at some intermediate time level t^* where $t^n \leqslant t^* \leqslant t^{n+1}$. The linearization is performed simply by computing the vector quantity $[\partial\mathscr{D}(\phi)/\partial t]^*(t^{n+1} - t^*)$ and replacing all occurrences of $(\partial\phi/\partial t)^*$, arising from chain-rule differentiation, by $(\phi^{n+1} - \phi^*)/(t^{n+1} - t^*)$. As in Eqs. (7) and (8), the result can be denoted symbolically as $(\partial\mathscr{D}/\partial\phi)^*(\phi^{n+1} - \phi^*)$, and the linearization formula is then given by

$$\mathscr{D}(\phi^{n+1}) = \mathscr{D}(\phi^*) + (\partial\mathscr{D}/\partial\phi)^*(\phi^{n+1} - \phi^*) + O(\Delta t)^2 \tag{17}$$

which is linear in ϕ^{n+1} as desired. Linearization as in Eq. (17) is effectively performed [illegible] manipulation, and yet maintains considerable generality.

As mentioned earlier, there is one notable difference between the methods of [1, 2] regarding implementation of the linearization. Briley and McDonald [2–4] linearize the fundamental implicit scheme in the manner of (1), (7) and (17) and then apply the Douglas–Gunn splitting technique to generate a consistently split block implicit scheme for the purpose of "solving" the linearized multidimensional difference equations. In this instance, the linearized scheme is the same as Eqs. (13) except that all $\mathscr{D}$-operators appearing on the left-hand side of Eqs. (13) are replaced by $(\partial\mathscr{D}/\partial\phi)^n$ as defined previously. The linearized scheme is thus given by

$$[I - \beta\,\Delta t(\partial\mathscr{D}_x/\partial\phi)^n](\phi^* - \phi^n) = \Delta t(\mathscr{D}_x + \mathscr{D}_y + \mathscr{D}_z)^n\,\phi^n, \tag{18a}$$

$$[I - \beta\,\Delta t(\partial\mathscr{D}_y/\partial\phi)^n](\phi^{**} - \phi^n) = \phi^* - \phi^n, \tag{18b}$$

$$[I - \beta\,\Delta t(\partial\mathscr{D}_z/\partial\phi)^n](\phi^{***} - \phi^n) = \phi^{**} - \phi^n. \tag{18c}$$

In the special case where $\mathscr{D}$ is linear, the relationship

$$(\partial\mathscr{D}/\partial\phi)^n(\phi^{n+1} - \phi^n) = \mathscr{D}^n(\phi^{n+1} - \phi^n) \tag{19}$$

holds and Eqs. (18) are seen to reduce to Eqs. (13).

On the other hand, Lindemuth and Killeen [1] constructed a nonlinear Peaceman–Rachford scheme and then linearized the ADI scheme by Taylor series expansion about $\boldsymbol{\phi}^n$ during the step analogous to (15a), followed by expansion about $\hat{\boldsymbol{\phi}}$ during the step analogous to (15b). In the present notation, the linearized extension of Eqs. (15a) and (15b) used by Lindemuth and Killeen can be written as

$$\left[\boldsymbol{I} - \frac{\Delta t}{2}\left(\frac{\partial \mathscr{D}_x}{\partial \boldsymbol{\phi}}\right)^n\right](\hat{\boldsymbol{\phi}} - \boldsymbol{\phi}^n) = \frac{\Delta t}{2}(\mathscr{D}_x + \mathscr{D}_y)^n \boldsymbol{\phi}^n, \tag{20a}$$

$$\left[\boldsymbol{I} - \frac{\Delta t}{2}\left(\frac{\partial \hat{\mathscr{D}}_y}{\partial \boldsymbol{\phi}}\right)\right](\boldsymbol{\phi}^{**} - \hat{\boldsymbol{\phi}}) = \frac{\Delta t}{2}(\hat{\mathscr{D}}_x + \hat{\mathscr{D}}_y)\hat{\boldsymbol{\phi}}. \tag{20b}$$

As in the linear case, the relationship of Eqs. (18) and (20) is clarified under the change of variables $\hat{\boldsymbol{\phi}} = (\boldsymbol{\phi}^* + \boldsymbol{\phi}^n)/2$ and with $\beta = 0.5$, $\mathscr{D}_z = 0$. Under these circumstances, Eqs. (18a) and (20a) for the first ADI sweep are equivalent, but Eqs. (18b) and (20b) for the second ADI sweep differ by the quantity

$$\frac{\Delta t}{4}\left[\left(\frac{\partial \mathscr{D}_x{}^n}{\partial \boldsymbol{\phi}}\right)(\boldsymbol{\phi}^* - \boldsymbol{\phi}^n) - \hat{\mathscr{D}}_x(\boldsymbol{\phi}^* + \boldsymbol{\phi}^n) + 2\mathscr{D}_x{}^n\boldsymbol{\phi}^n\right.$$
$$\left. + \left(\frac{\partial \hat{\mathscr{D}}_y}{\partial \boldsymbol{\phi}}\right)(\boldsymbol{\phi}^* - \boldsymbol{\phi}^n) - \hat{\mathscr{D}}_y(\boldsymbol{\phi}^* + \boldsymbol{\phi}^n) + 2\mathscr{D}_y{}^n\boldsymbol{\phi}^n\right] \tag{21}$$

which reduces to zero if $\mathscr{D}$ is linear and independent of t.

Both techniques (18) and (20) are accurate to $O(\Delta t)^2$, lead to identical steady solutions, and have been successful in numerous applications. Use of the updated linearization technique as in Eq. (20b) may well reduce linearization errors. On the other hand, the use of updated expansions complicates the treatment of nonlinear boundary conditions, making it necessary to compute intermediate boundary values on lateral boundaries, which otherwise may not be needed. In addition, Eq. (18b) is less complicated and requires fewer arithmetic operations than Eq. (20b). Although the Peaceman–Rachford formulation is restricted to two space dimensions, the use of updated linearizations can be generalized to three dimensions and other ADI schemes by using the Douglas–Gunn formulation (11) and expanding about the most recent intermediate solution available. Without enumerating the various options, it is noted that the resulting schemes with updated intermediate step linearizations have the same formal accuracy as Eqs. (18), but are not as simple. Although there is no particular evidence to support the conjecture, the use of updated linearizations in schemes derived from Eqs. (11) may also adversely affect stability, since Douglas and Gunn [22] noted that if they had used the most recent approximation in constructing their (linear) ADI schemes, then unconditional stability would be lost in three dimensions. The use of updated linearizations in conjunction with split schemes which have inconsistent intermediate steps is certainly not recommended, for reasons which will subsequently become apparent. Finally, it is noted that the two-level version of the "delta" form factorization of Beam and Warming [6, 7] has the form of Eq. (14) in two dimensions, and the recommended computational form is the Douglas–Gunn splitting written in the form of Eq. (13). Since Warming and Beam [6, 7] and Briley and McDonald [2–4] both implement the time linearization as in Eqs. (18), these methods are identical in their structure as linearized block ADI schemes.

As a final note, for the case of nonlinear $\mathscr{D}$-operators, ADI schemes such as Eqs. (11) or (15) can be solved directly using (one-dimensional) Newton–Raphson iteration. The resulting schemes might be termed "nonlinear block ADI methods," and although they require correspondingly more computational effort per time step, they do eliminate linearization errors in transient solutions. The method of Baum and Ndefo [25] consists of a nonlinear two-dimensional Peaceman–Rachford scheme solved by quasilinear iteration using a banded (as opposed to block) linear solver and falls into this general category.

Schemes Having Inconsistent Intermediate Steps

It should be emphasized that the factored form (14) was derived [4] as a combined form of the block ADI scheme (11) and (12) or (13). It is possible to "reverse" the derivation and first obtain the factored form which is then split to produce an ADI scheme. The general approach of deriving schemes as split forms of difference schemes written in a factored form is termed approximate factorization and was originated and developed extensively by Yanenko and D'Yakonov [26]. Varga [27] expresses a variety of schemes for linear scalar equations in terms of Padé rational approximations for exponentials of matrices and derives the Peaceman–Rachford matrix in factored form as an approximate form of the Crank–Nicolson formula. Mitchell [24] identifies numerous scalar ADI and LOD schemes as split forms of implicit difference formulas in factored form. Beam and Warming [5–7] also follow the D'Yakonov type of derivation to arrive at their approximate factorization methods, and for example, note that Eq. (14) is an approximate factorization of the implicit scheme (10) and that Eq. (14) can be split in the manner of D'Yakonov to yield the suggested computational form Eq. (13). Finally, Gourlay [28] has recently surveyed and categorized numerous splitting methods for linear scalar time-dependent PDE's.

In deriving algorithms by the D'Yakonov approach of creating split versions of "factored" implicit schemes, it should be noted that certain key features of the various algorithms are not discernable in the factored form but become evident only after splitting into a usable computational form, a process which in general is not unique. This is particularly evident if the general factored form Eq. (14), is rewritten as

$$\begin{aligned}(I - \beta\,\Delta t\mathscr{D}_x)(I - \beta\,\Delta t\mathscr{D}_y)(I - \beta\,\Delta t\mathscr{D}_z)\,\phi^{n+1}\\ = (I - \beta\,\Delta t\mathscr{D}_x)(I - \beta\,\Delta t\mathscr{D}_y)(I - \beta\,\Delta t\mathscr{D}_z)\,\phi^{n}\\ + \Delta t(\mathscr{D}_x + \mathscr{D}_y + \mathscr{D}_z)\,\phi^{n} + O(\Delta t^3).\end{aligned} \tag{22}$$

As an illustration, if the factored form given by Eq. (22) is restricted to two dimensions ($\mathscr{D}_z = 0$) and $\beta = \frac{1}{2}$, then Eq. (22) can be rewritten as

$$\left(I - \frac{\Delta t}{2}\mathscr{D}_x\right)\left(I - \frac{\Delta t}{2}\mathscr{D}_y\right)\phi^{n+1} = \left(I + \frac{\Delta t}{2}\mathscr{D}_x\right)\left(I + \frac{\Delta t}{2}\mathscr{D}_y\right)\phi^n + O(\Delta t^3). \qquad (23)$$

Equation (23) is the factored form originally used by Beam and Warming [5] to simplify the treatment of homogeneous functions in the compressible Euler equations. It is apparent from the previous discussion that Eq. (23) is also a combined form of the Peaceman–Rachford splitting (15). In their first method, Beam and Warming [5] employed the following alternative splitting of Eq. (23):

$$\phi^* = \left(I + \frac{\Delta t}{2}\mathscr{D}_y\right)\phi^n, \qquad (24a)$$

$$\left(I - \frac{\Delta t}{2}\mathscr{D}_x\right)\phi^{**} = \left(I + \frac{\Delta t}{2}\mathscr{D}_x\right)\phi^*, \qquad (24b)$$

$$\left(I - \frac{\Delta t}{2}\mathscr{D}_y\right)\phi^{***} = \phi^{**}, \qquad (24c)$$

$$\phi^{n+1} = \phi^{***} + O(\Delta t^3) \qquad (24d)$$

which upon combining Eqs. (24a) and (24b) is a splitting due to D'Yakonov (cf. Mitchell [24]). Since each of Eqs. (24a)–(24c) contain either $\mathscr{D}_x$ or $\mathscr{D}_y$ but not both, scheme (24) also bears a resemblance to locally one-dimensional (LOD) schemes. A more important property of this scheme is that the intermediate steps (24a) and (24b) do not satisfy the consistency condition mentioned previously. For example, whereas the combined scheme or factored form (23) approximates

$$\frac{\partial\phi}{\partial t} = (\mathscr{D}_x + \mathscr{D}_y)\,\phi + O(\Delta t^2), \qquad (25)$$

the intermediate solution ϕ^{**} obtained from Eq. (24b) approximates a solution of

$$\frac{\partial\phi}{\partial t} = (\mathscr{D}_x + \tfrac{1}{2}\mathscr{D}_y)\,\phi + O(\Delta t). \qquad (26)$$

Thus, intermediate steps possess properties somewhat analogous to the well-known inconsistency of the Dufort–Frankel scheme (e.g., Richtmyer and Morton [20]). Specifically, the Dufort–Frankel scheme for approximating the scalar equation $\partial u/\partial t = \partial^2 u/\partial x^2$ contains a truncation error term $(\Delta t/\Delta x)^2\, \partial^2 u/\partial t^2$ and as a consequence represents its differential counterpart not as $\Delta t \to 0$ but only as $\Delta t/\Delta x \to 0$. In the present case of Eq. (24b), it can be seen from Eq. (26) that ϕ^{**} does not represent a solution of Eq. (25) regardless of how $\Delta t \to 0$. This lack of consistency will now be demonstrated to have rather serious consequences for implementation of boundary conditions.

INTERMEDIATE BOUNDARY CONDITIONS

Since the overall scheme (24) nevertheless approximates Eq. (25), the inconsistent intermediate step (24b) would not be of serious concern except that *implicit* boundary conditions ϕ^{**} are required for solution of Eq. (24b). For Dirichlet problems, implicit boundary conditions for intermediate steps can be derived for all split schemes by recombining intermediate steps in reverse order (cf. Mitchell [24]). Following this procedure, boundary conditions for ϕ^{**} for use during the x-implicit step, as required by Eq. (24b), are given by the third-order formula

$$\phi^{**} = \left(I - \frac{\Delta t}{2}\mathscr{D}_y\right)\phi_{n+1} + O(\Delta t^3). \tag{27}$$

Note that use of the "physical" boundary condition ϕ^{n+1} as ϕ^{**} introduces an $O(\Delta t)$ error not present in the combined form (23). If transient accuracy is of interest, this implies an $O(1)$ error in representing $\partial\phi/\partial t$. For Dirichlet problems, the use of the intermediate boundary condition (27) is merely an inconvenience. If derivative boundary conditions are prescribed, however, more serious problems can arise. For example, suppose $\partial\phi^{n+1}/\partial x = f(y, t)$ is prescribed as an x-implicit boundary condition. Differentiation of Eq. (27) results in

$$\frac{\partial\phi^{**}}{\partial x} = \frac{\partial\phi^{n+1}}{\partial x} - \frac{\Delta t}{2}\frac{\partial}{\partial x}\mathscr{D}_y\phi^{n+1} + O(\Delta t^3). \tag{28}$$

If $\mathscr{D}_y\phi$ has a form such as $c_1\partial\phi/\partial y + c_2\partial^2\phi/\partial y^2$, where c_1, c_2 are scalar and either constants or independent of x, then $\partial(\mathscr{D}_y\phi)/\partial x$ can be evaluated *analytically*, and Eq. (28) becomes

$$\frac{\partial\phi^{**}}{\partial x} = f^{n+1} - \frac{\Delta t}{2}\left(c_1\frac{\partial f}{\partial y} + c_2\frac{\partial^2 f}{\partial y^2}\right)^{n+1} + O(\Delta t^3). \tag{29}$$

However, in the more common case wherein $\mathscr{D}_y$ has variable coefficients of some sort [redacted] when needed by the algorithm. In such cases, a less satisfactory alternative is available, and that is to evaluate $\mathscr{D}_y\phi^{n+1}$ *numerically* but to replace ϕ^{n+1} by "lagged" values ϕ^n. Since by Taylor's theorem $\phi^{n+1} = \phi^n + O(\Delta t)$, Eq. (28) becomes the second-order formula

$$\frac{\partial\phi^{**}}{\partial x} = f^{n+1} - \frac{\Delta t}{2}\frac{\partial}{\partial x}\mathscr{D}_y\phi^n + O(\Delta t^2) \tag{30}$$

and time derivatives can at least be represented to first order. Evidently, the formal accuracy of the overall scheme can be maintained only by resorting to multilevel extrapolation or a predictor–corrector treatment.

The foregoing situation is precisely one order of accuracy improved with the Peaceman–Rachford splitting (15), since (consistent) intermediate boundary conditions for (15a) are given by

$$\phi^* = \frac{\phi^{n+1} + \phi^n}{2} + \frac{\Delta t}{4}\mathscr{D}_y(\phi^{n+1} - \phi^n) + O(\Delta t^3). \tag{31}$$

Since $(\phi^{n+1} - \phi^n) = O(\Delta t)$, accuracy is the same for both function and derivative boundary conditions. The use of either $\phi^{n+1/2}$ or $(\phi^{n+1} + \phi^n)/2$ as ϕ^* introduces only an $O(\Delta t^2)$ error. Time derivatives are represented to $O(\Delta t)$. If $\mathscr{D}_y\phi$ can be evaluated analytically as in Eq. (29), then use of Eq. (31) retains the transient accuracy of the overall scheme. Otherwise, there is no reason to use anything other than "physical" boundary conditions.

It should be emphasized that the Douglas–Gunn procedure for deriving split schemes leads only to schemes such as (11–13) and (15) whose intermediate steps represent consistent solutions (provided the unsplit scheme is consistent) and which also satisfy

$$\phi^*, \phi^{**} = \phi^{n+1} + O(\Delta t^2). \tag{32}$$

Furthermore, steady (stationary) solutions satisfy

$$\phi^* = \phi^{**} = \phi^{***} = \phi^{n+1} = \phi^n. \tag{33}$$

The D'Yakonov splitting (24) does not satisfy either Eq. (32) or (33). As a consequence of Eqs. (32) and (33), boundary conditions for ϕ^{n+1} can be applied during intermediate steps of the Douglas–Gunn splitting without serious loss of transient accuracy and with no loss of accuracy in steady solutions. In contrast, failure to correct intermediate

TABLE II

Influence of Intermediate Boundary Condition Treatment on Error

	Error in $\partial\phi/\partial t$ on boundary, transient applications		Error in satisfying steady boundary conditions	
Type of boundary correction	Peaceman–Rachford (15)	D'Yakonov (24)	Peaceman–Rachford (15)	D'Yakonov (24)
None–"physical" conditions used	$O(\Delta t)$	$O(1)$	None	$O(\Delta t)$
Approximate as in Eq. (30)	Not applicable	$O(\Delta t)$	Not applicable	None
Exact, as in Eqs. (27), (28), and (31) (if possible)	$O(\Delta t)^2$	$O(\Delta t)^2$	None	None

boundary conditions using the D'Yakonov splitting (24) leads to a steady solution which depends on Δt and which does not satisfy the boundary conditions by an amount proportional to Δt. These results are summarized in Table II. Although derivative and some other boundary conditions may preclude the use of exact formulas for boundary corrections such as Eq. (28), it may be possible to order the ADI steps such that the troublesome boundary conditions occur only on the last step, where boundary correction is unnecessary. Finally, consistent splitting is attractive in any application requiring accuracy of the intermediate steps (for example, if linearizations are updated). For the foregoing reasons, the Douglas–Gunn (consistent) splitting (11)–(13) seems highly preferable to the D'Yakonov (inconsistent) splitting (24) as well as other inconsistent splittings such as that of LOD schemes.

Efficiency Gains from Reducible Block Submatrices

In almost all practical applications, linearization by Taylor expansion in time necessarily leads to implicitly coupled difference equations. In the multidimensional case, use of a splitting scheme reduces the implicitly coupled difference equations to a sequence of narrow block-banded systems (usually block-tridiagonal), as is of critical importance for the overall efficiency of the method. Generally, if there are L equations in L dependent variables, block $(L \times L)$ submatrices are obtained. In many applications, however, the $(L \times L)$ submatrices are *reducible* (for a definition, see Varga [27]), and in such cases, further gains in efficiency are possible. In any block-banded system, if all $(L \times L)$ submatrices *including boundary conditions* are reducible in the same manner, then that block $(L \times L)$ banded system can be solved as a sequence of irreducible block $(l_i \times l_i)$ banded systems, where $\sum_i l_i = L$ $(1 \leqslant l_i \leqslant L)$. This can result in a substantial reduction in arithmetic operations, since solution of a block tridiagonal system nominally requires $(3N - 2)(L^3 + L^2)$ operations [16], where N is the number of diagonal blocks (i.e., L coupled equations along a row of N grid points).

As an example, if after suitable permutation the $(L \times L)$ submatrices contain a (row/column) containing only zero off-diagonal entries, the $(L \times L)$ submatrices are reducible, and the particular variable(s) associated with those diagonal entries can be computed (in this case) by a scalar tridiagonal elimination (before/after) solution of the remaining variables. Briley and McDonald [2–4] exploited this property for the three-dimensional compressible Navier–Stokes equations and reduced the block (5×5)-tridiagonal systems to one block (3×3) and two scalar tridiagonal systems [i.e., $L = 5$, $l = (3, 1, 1)$]. The partitioned submatrices have the following form:

$$\left[\begin{array}{c|c|ccc} X & 0 & X & X & 0 \\ \cline{1-2} 0 & X & X & X & 0 \\ \cline{3-5} 0 & 0 & X & X & 0 \\ 0 & 0 & X & X & X \\ 0 & 0 & X & X & X \end{array}\right]$$

where X denotes a nonzero entry, and the solution vector has an ordering such as $(v, w, \rho, u, T)^T$. The associated programming logic is not difficult. McDonald and Briley [3] also obtained $L = 4$; $l = (3, 1)$ for steady three-dimensional adiabatic supersonic flow.

When L becomes large, the effort required to solve block-banded systems is considerable, and the partitioning of reducible submatrices can easily become critical. As a further indication of what can be accomplished through reducible submatrices, Gibeling, McDonald, and Briley [29] have considered the three-dimensional compressible Navier–Stokes equations for reacting flow with "stiff" chemistry. If, for example, the chemistry considered is the formation of nitric oxide by the Zeldovitch mechanism [30], then five chemical species are introduced as a result of the two chemical reactions which occur. Since one of the species can be determined from an algebraic conservation equation, only four species are effectively coupled with the five fluid dynamic variables; thus $L = 9$. Nevertheless, on the first intermediate step of the splitting, the submatrices are reducible such that $l = (7, 1, 1)$. Since the chemistry coupling occurs only through a pointwise source term vector (analogous to F in Eq. (8b)) which does not appear in the second and third steps of the Douglas–Gunn splitting, further reduction is possible. In particular, if on the second and third steps, the species change contribution to the pressure gradient is neglected (an excellent assumption in many chemistry problems) the submatrices are then reducible such that $l = (3, 1, 1, 4)$, with another major gain in efficiency.

As a further observation, once the block submatrices have been partitioned into an irreducible form, nonzero elements which are small $[O(\epsilon)]$ compared with other remaining elements (but which prevent further formal reduction of the submatrices) may be identified. If the consistently split block implicit scheme given by Eqs. (13) is being used, then it is obvious that neglecting any such $O(\epsilon)$ element is equivalent to explicit evaluation of the corresponding term or terms in the governing equations, that is, evaluating these terms at t^n rather than at $t^n + \beta \Delta t$. Neglect of such $O(\epsilon)$ submatrix elements may be termed "order epsilon decoupling" and may be desirable if it permits further reduction of the submatrices. Of course, the explicit evaluation of such terms raises the prospect of reduced stability bounds, and hence this concept must be used judiciously. Nevertheless, it seems well worthwhile to weigh the trade-offs involved. With this in mind, we return to the example of the Zeldovitch formation of nitric oxide and note that the chemical reactions are essentially isoenergetic and that, apart from the principle constituent species (which are virtually constant), all other varying species are present only in trace quantities. As a consequence, submatrices can be reduced such that $l = (3, 1, 1, 4)$ for all three split steps. Furthermore, if the transient behavior is of no interest, then the $l_{i=4}$ blocks may be solved only once after the fluid dynamic steady state has been achieved. For the foregoing reasons, linearized block implicit methods are particularly well suited for chemically reacting flows having "stiff" chemical reactions, provided they are consistently split. In this same environment, for a model time-dependent stiff reaction—diffusion equation in one space dimension, Dwyer and Otey [31] found the linearized block implicit scheme very effective, although for their system reducible subblocks were not investigated.

More recently, Steger [32] has further explored benefits associated with reducible block submatrices and has obtained $L = 4$; $l = (2, 1, 1)$ for the two-dimensional compressible Euler equations.

The Use of Multiple Time Steps

The use of time-dependent schemes to obtain steady solutions for complex systems of elliptic or mixed type is well known and, at least in fluid dynamics applications, is probably more common than their use for computing accurate transients. In such circumstances, an emphasis on order of accuracy of the transient is unwarranted, and the availability of the time step as a free parameter which can be used to improve the rate of convergence to a steady solution is more relevant. In their consideration of a scalar diffusion equation, Peaceman and Rachford [23] showed that the cyclic use of a sequence of acceleration parameters (or equivalently time steps) of differing magnitude provides rapid convergence using their ADI method. The effectiveness of parameter cycling in accelerating convergence, particularly for refined meshes, is further demonstrated for model problems in a survey article by Birkhoff, Varga, and Young [33]. Although rigorous extension of the associated analysis to complex systems of nonlinear equations is far from straightforward, the underlying concepts can [illegible]

spatial wavelength, whereas a large time step (small parameter) would tend to reduce errors of large wavelength more effectively. Thus in iterating toward a steady solution, rather than use as large a time step as possible, or even attempt to determine an optimum time step, one should perhaps use a sequence of time steps distributed throughout the range of time scales present in the problem being solved. For example, in the scalar diffusion equation $\partial\phi/\partial t = \partial^2\phi/\partial x^2$, a suitable minimum time step would be the explicit stability limit $\Delta t_{\min} \doteq \Delta x^2/2$ and a suitable maximum would be the time required for diffusion to span the entire computational domain, $\Delta t_{\max} \doteq (N\Delta x)^2/4$, where N is the number of mesh increments. In this particular case, these time scales are known to correspond to reasonable bounds on the eigenvalue range from which the theoretical acceleration parameter sequences are selected (cf. [33]). The utility of considering relevant time scales is simply that in complex applications the user often has a much better understanding of the time scales present in his particular problem than he does of the eigenvalues of matrices associated with the numerical scheme. In fluid dynamics applications, the present authors have used this technique for selecting sequences of time steps on a number of problems with apparent success in reducing the number of time steps to reach steady state. In support of these general arguments, it has been found that during various stages of the iteration process, the largest changes in the dependent variables often occur when taking the *smallest* time step.

Summary and Concluding Remarks

"Linearized block implicit" methods have been unified and briefly related to scalar ADI schemes as well as to iterative and noniterative methods for systems of nonlinear algebraic and differential equations. Methods based on the D'Yakonov or other splittings whose intermediate steps do not satisfy the consistency condition were found to present difficulties in treating boundary conditions, particularly derivative conditions. Substantial gains in efficiency, obtainable when block submatrices are reducible, were described. Regarding the derivation of split linearized block implicit schemes, it is noted that the development of Beam and Warming places an emphasis on the factored or combined form rather than on the split scheme actually used in computations. Since the splitting process is not unique, key features of the various algorithms are not discernable in the factored form and, as demonstrated here, become evident only after splitting. Further, the restriction that a scheme be factorable might eliminate from consideration potentially useful schemes. Here, an emphasis is placed on the consistency of intermediate steps obtained through use of the Douglas–Gunn procedure for deriving split schemes, in view of its implications for treating boundary conditions. Thus, in the view of the present authors, the "factored forms," Eqs. (14), (22), and (23) are all equivalent combined forms of the split schemes Eqs. (11)–(13), (15), and (16) which in turn are minor variations of the Douglas–Gunn [22] (consistent) splitting technique; whereas, the D'Yakonov (inconsistent) splitting (24) of the factored form (23) produces a method with entirely different properties. Finally, an alternative D'Yakonov type splitting of the "delta" factored form, Eq. (14), as proposed by Warming and Beam, is shown here to rederive the Douglas–Gunn splitting, Eq. (13).

References

1. I. Lindemuth and J. Killeen, *J. Computational Phys.* **13** (1973), 181.
2. W. R. Briley and H. McDonald, Solution of the three-dimensional compressible Navier–Stokes equations by an implicit technique, *in* "Proceedings Fourth International Conference on Numerical Methods in Fluid Dynamics, Boulder, Color. June 1974," Springer-Verlag, New York/Berlin, 1975; also, United Aircraft Research Laboratories Report M911363-6, 1973.
3. H. McDonald and W. R. Briley, *J. Computational Phys.* **19** (1975), 150; also, United Aircraft Research Laboratories Report N111078-1, 1974.
4. W. R. Briley and H. McDonald, *J. Computational Phys.* **24** (1977), 372; also, United Technologies Research Center Report R75-911363-15, 1976.
5. R. M. Beam and R. F. Warming, *J. Computational Phys.* **22** (1976), 87.
6. R. F. Warming and R. M. Beam, "On the Construction and Application of Implicit Factored Schemes for Conservation Laws," *Symposium on Computational Fluid Dynamics*, New York, April 1977; SIAN-AMS Proceedings, Vol. 11, 1977.
7. R. M. Beam and R. F. Warming, *AIAA J.* **16** (1978), 393.
8. A. R. Gourlay and J. L. Morris, *Math. Comp.* **22** (1968), 28.
9. I. R. Lindemuth, *J. Computational Phys.* **18** (1975), 119; Erratum, **19** (1976), 338.
10. M. Ciment, H. Leventhal, and B. C. Weinberg, *J. Computational Phys.* **28** (1978), 135.
11. A. Ralston, "A First Course in Numerical Analysis," McGraw–Hill, New York, 1965.

12. H. B. KELLER, "Numerical Methods for Two-Point Boundary Value Problems," Ginn/Blaisdell, Waltham, Mass., 1968.
13. R. E. BELLMAN AND R. E. KALABA, "Quasilinearization and Nonlinear Boundary Value Problems," American Elsevier, New York, 1965.
14. H. LOMAX, Stable implicit and explicit numerical methods for integrating quasi-linear differential equations with parasitic–stiff and parasitic–saddle eigenvalues, NASA TN D-4703, 1968.
15. L. LAPIDUS AND J. H. SEINFELD, "Numerical Solution of Ordinary Differential Equations," Academic Press, New York, 1971.
16. E. ISAACSON AND H. B. KELLER, "Analysis of Numerical Methods," Wiley, New York, 1966.
17. H. B. KELLER, A New Difference Scheme for Parabolic Problems, *in* "Numerical Solution of Partial Differential Equations," (J. Bramble Ed.), Vol. 2, Academic Press, New York, 1970.
18. F. G. BLOTTNER, *Comput. Methods Appl. Mech. Engrg.* **6** (1975), 1.
19. S. G. RUBIN AND P. K. KHOSLA, *Computers and Fluids* **5** (1977), 241.
20. R. D. RICHTMYER AND K. W. MORTON, "Difference Methods for Initial Value Problems," 2nd ed., Interscience, New York, 1967.
21. J. P. KRESKOVSKY AND S. J. SHAMROTH, *Comput. Methods Appl. Mech. Engrg.* **13** (1978), 307.
22. J. DOUGLAS AND J. E. GUNN, *Numer. Math.* **6** (1964), 428.
23. D. W. PEACEMAN AND H. H. RACHFORD, *J. Soc. Indust. Appl. Math.* **3** (1955), 28.
24. A. R. MITCHELL, "Computational Methods in Partial Differential Equations," Wiley, New York, 1969.
25. E. BAUM AND E. NDEFO, Proceedings of the AIAA Computational Fluid Dynamics Conference, New York, 1973, p. 133.
26. N. N. YANENKO, "The Method of Fractional Steps," Springer-Verlag, New York/Berlin, 1971.
27. R. S. VARGA, "Matrix Iterative Analysis," Prentice–Hall, Englewood Cliffs, N.J., 1962.
28. A. R. GOURLAY, Splitting Methods for Time-Dependent Partial Differential Equations, *in* "The State of the Art in Numerical Analysis" (D. Jacobs, Ed.), p. 757, Academic Press, London, 1977.
29. H. J. GIBELING, H. MCDONALD, AND W. R. BRILEY, Development of a three-dimensional combustor flow analysis, Vol. I, Air Force Aero Propulsion Laboratory Report AFAPL-TR-75-59, 1975.
30. YA. B. ZELDOVITCH, P. YA SADOUNIKOV, AND D. A. FRANK-KANENETSKILL, "Oxidation of Nitrogen in Combustion," Academy of Sciences of USSR, Institute of Chemical Physics, 1947.
31. H. DWYER AND G. OTEY, A numerical study of the interaction of fast chemistry and diffusion, AIAA Paper 78-946, 1978.
32. J. L. STEGER, *Comput. Methods Appl. Mech. Engrg.* **13** (1978), 175.
33. G. BIRKHOFF, R. S. VARGA, AND D. YOUNG, Alternating Direction Implicit Methods, *in* "Advances in Computers," Vol. 3, Academic Press, New York, 1962.