

16-A079 305

NAVAL RESEARCH LAB WASHINGTON DC
RECENT DEVELOPMENTS IN COMPUTATIONAL TECHNIQUES FOR APPLIED HYD--ETC(U)
DEC 79 D L BOOK , J P BORIS , M J FRITTS
NRL-MR-4095

F/G 20/4

UNCLASSIFIED

NL

1 of 1
AD
A079310C

END
DATE
FILMED
2-80
DDC

ADA 079305

**Recent Developments in Computational Techniques
for Applied Hydrodynamics**

D. L. BOOT, J. P. BOAT, AND M. J. FARRIS

Laboratory for Computational Physics

R. V. MADALA, B. E. McDONALD, N. K. WINSON, AND S. T. ZALERAK

Plasma Physics Division

December 7, 1979

DDC
RECEIVED
JAN 11 1980
A



NAVAL RESEARCH LABORATORY
Washington, D.C.

Approved for public release; distribution unlimited

80 1 10 035

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM	
1. REPORT NUMBER NRL Memorandum Report 4095	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER 14 NR. - 11 1 - 1751	
4. TITLE (and Subtitle) RECENT DEVELOPMENTS IN COMPUTATIONAL TECHNIQUES FOR APPLIED HYDRODYNAMICS,		5. TYPE OF REPORT & PERIOD COVERED Final Report	6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) D. L. Book, J. P. Boris, M. J. Fritts, R. V. Madala B. E. McDonald, N. K. Winsor, and S. T. Zalesak		8. CONTRACT OR GRANT NUMBER(s)	
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Research Laboratory Washington, D.C. 20375		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS ONR Project RR011-09-41 and RR0140302 NRL Problem 62H02-51 and 62H02-39	
11. CONTROLLING OFFICE NAME AND ADDRESS Office of Naval Research 800 N. Quincy Street Arlington, Va. 22203		12. REPORT DATE Dec 1979	
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office) 10 RR011-09-41 11 1751		13. NUMBER OF PAGES 74	
		15. SECURITY CLASS. (of this report) Unclassified	
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE	
16. DISTRIBUTION STATEMENT (of this Report) Approved for public release; distribution unlimited.			
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)			
18. SUPPLEMENTARY NOTES This project was sponsored by Office of Naval Research, Project No. RR011-09-41 and RR0140302. A			
19. KEY WORDS (Continue on reverse side if necessary and identify by block number) Numerical Method Fluids Incompressible Flow Finite Difference Methods Poisson Equation Convective Equations			
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) Recently developed techniques for numerical solution of fluid equations are reviewed. Both convective equations and Poisson's equations are discussed. The emphasis is on methods developed and in use at NRL, although brief descriptions of competitive and related methods are included as background material. Examples and applications are given, with particular attention to the advantageous of vector computers.			

DD FORM 1473
1 JAN 73

EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-014-6601

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

25

CONTENTS

1. INTRODUCTION	1
2. NUMERICAL TECHNIQUES FOR SOLUTION OF CONVECTIVE EQUATIONS	2
3. FLUX-CORRECTED TRANSPORT	10
4. A MULTILEVEL SEMI-IMPLICIT NUMERICAL MODEL FOR BAROCLINIC OCEANS	25
5. LAGRANGIAN FLUID DYNAMICS USING TRIANGULAR GRIDS	30
6. SOLUTION OF ELLIPTIC EQUATIONS	38
7. SPECTRAL METHODS	62
REFERENCES.....	67

Accession	
NTIS	
EDITION	
1	
2	
3	
4	
5	
6	
7	
8	
9	
10	
11	
12	
13	
14	
15	
16	
17	
18	
19	
20	
21	
22	
23	
24	
25	
26	
27	
28	
29	
30	
31	
32	
33	
34	
35	
36	
37	
38	
39	
40	
41	
42	
43	
44	
45	
46	
47	
48	
49	
50	
51	
52	
53	
54	
55	
56	
57	
58	
59	
60	
61	
62	
63	
64	
65	
66	
67	
68	
69	
70	
71	
72	
73	
74	
75	
76	
77	
78	
79	
80	
81	
82	
83	
84	
85	
86	
87	
88	
89	
90	
91	
92	
93	
94	
95	
96	
97	
98	
99	
100	

A

RECENT DEVELOPMENTS IN COMPUTATIONAL TECHNIQUES FOR APPLIED HYDRODYNAMICS

I. INTRODUCTION

A tremendous range of physical phenomena are described by fluid equations. All four of the classical "elements" — earth, water, air and fire — behave, at least on some scales, as fluids. For this to be the case, the features they share in common must be very simple, in spite of the variegation and superficial complexity we are accustomed to dealing with. This is indeed the case, and the present review deals chiefly with those central aspects.

The mathematical description of fluid motion makes use of partial differential equations which propagate the fluid variables (density, velocity, pressure, etc.) in time. Most systems of such equations closely resemble one another in containing convective derivatives, expressing causality, conservation laws and other properties, and in emphasizing the connectedness of the physical system. This latter property is often manifested in the appearance of a Poisson equation, which relates the pressure or velocity potential at a given point to the properties of the system as a whole.

Because we have emphasized the common features and generality of fluid behavior, most of the following discussion is theoretical, and the applications given are fairly simple. To apply the techniques to specific problems of interest, one or more of the following must be added to the basic (Navier-Stokes) fluid equations: (i) more dimensions; (ii) more complicated geometry or boundary conditions; (iii) additional source, sink or transport terms; (iv) more dependent variables; (v) an equation of state. For example, ideal MHD requires the introduction of the magnetic field strength B and the Lorentz force; nonideal MHD requires thermal conduction, resistivity and viscosity terms. Ocean internal waves require a buoyancy term and an eddy viscosity derived from a turbulence closure model. Chemically reactive flow requires the introduction of chemical variables, together with reaction rates, thermal conduction, and a turbulence model.

Although most of the physics in any given problem is comprised in these added terms, conditions and modifications, computational practice shows that most of the difficult numerical problems are associated with the universal fluid features. If adequate care is bestowed on developing accurate general numerical techniques for treating these, the resulting code invariably turns out to be robust, adaptable and efficient. Hence the techniques described here represent a substantial portion of the effort needed to construct even the most elaborate numerical hydrodynamic model.

As used in this review "computational techniques" relate to the solution of equations derived from theoretical models of hydrodynamic phenomena and are to be distinguished from the use of computers in data reduction and analysis. Almost all of the techniques are geared

toward solving nonlinear problems, which are inherently less accessible to classical analytic techniques.

Every theoretical model yields its predictions as solutions of the model equations. These may be as general as the 3D Navier-Stokes equations, or as specific as an equation for the evolution of a particular type of wave (e.g., solitons) or a steady-state turbulent spectrum. Broadly speaking, there is a correlation between generality and the following difficulties:

1. Heavy machine requirements (storage and CP time);
2. Highly sophisticated numerical methods needed;
3. Limitations in precision and long-term validity of predictions,

while specificity is associated with

1. Theoretical problems in deriving and justifying the model;
2. Limited plausibility and agreement of predictions with experiment.

In addition to determining the concrete predictions inherent in a particular set of assumptions, computer modeling can be used in an exploratory role, through "computer experiments." Where the precise set of assumptions or the values of the physical parameters are not clear, it is possible to generate a family of solution to study trends and systematic properties. This form of computation is often referred to as "numerical simulation" and is contrasted with "numerical solution" of equations.

Below we survey the computational techniques appropriate to fluid motion, with emphasis on those of greatest utility and already implemented, along with remarks about hardware and utility software requirements. The techniques are arranged roughly in order of decreasing generality. In several cases they overlap, and a particular technique may have application to physical problems other than the ones noted.

2. NUMERICAL TECHNIQUES FOR SOLUTION OF CONVECTIVE EQUATIONS

2.1 Importance of Convective Equations

One can either choose to treat the primitive (Navier-Stokes) equations

$$\frac{\partial \rho}{\partial t} + \mathbf{v} \cdot \nabla \rho = 0; \quad (2.1)$$

$$\nabla \cdot \mathbf{v} = 0; \quad (2.2)$$

$$\rho \left(\frac{\partial \mathbf{v}}{\partial t} + \mathbf{v} \cdot \nabla \mathbf{v} \right) + \nabla p = \rho \mathbf{g} + \nu \nabla^2 \mathbf{v} \quad (2.3)$$

in two or three dimensions, or one can replace (2.2) and (2.3) with the equivalent vortex dynamics equations

$$\mathbf{v} = \nabla \times \mathbf{A}; \quad (2.4)$$

$$\nabla^2 \mathbf{A} = -\boldsymbol{\zeta}; \quad (2.5)$$

$$\frac{\partial \zeta}{\partial t} + \nabla \cdot (\zeta \mathbf{v}) = \zeta \cdot \nabla \mathbf{v} + \nabla \rho \times \mathbf{g}/\rho + \nu \nabla^2 \zeta; \quad (2.6)$$

where $\mathbf{A}$ is the vector stream function. In two dimensions (2.4)–(2.6) become scalar equations:

$$\mathbf{v} = \hat{\mathbf{z}} \times \nabla \psi; \quad (2.7)$$

$$\nabla^2 \psi = \zeta; \quad (2.8)$$

$$\frac{\partial \zeta}{\partial t} + \nabla \cdot \zeta \mathbf{v} = \hat{\mathbf{z}} \cdot \nabla \rho \times \mathbf{g}/\rho + \nu \nabla^2 \zeta. \quad (2.9)$$

Either formulation involves convective equations [i.e., (2.1), (2.3), (2.6), (2.9)], which therefore assume a central role in numerical fluid dynamics. In addition, the vorticity formulation involves equations of Poisson's type, (2.5) or (2.8). In this section we treat only the former; the latter are discussed in Section 6. Except for eddy viscosity terms, etc., which arise from turbulence closure models, we ignore transport terms (heat conduction, molecular viscosity, diffusion) which are small except in very narrow regions, and in any case are computed as corrections and do not affect the convective properties.

We turn now to a survey of methods for treating convective equations.

2.2 A Review of Convective Equation Algorithms

This section reviews the different numerical techniques which have been developed to solve fluid dynamics problems based on the convective equation (written in conservation form in terms of a general variable ρ)

$$\frac{\partial \rho}{\partial t} = - \nabla \cdot \rho \mathbf{v}, \quad (2.1')$$

and the resulting hydrocodes at NRL based on these techniques (Boris, 1976a). Six different available methods are compared and categorized in a framework provided by a set of six requirements which ought to be satisfied by an ideal continuity equation algorithm. The strengths and weaknesses of the different approaches are analyzed. Finite-difference techniques have particularly attractive properties in this framework. Hence it will be worthwhile to correct, at least partially, the difficulties from which Eulerian and Lagrangian finite-difference techniques suffer, discussed in Section 3.1. Section 3 as a whole is devoted to Flux-Corrected Transport (FCT) and related Eulerian solutions of the nonlinear continuity equation which preserve the positivity property of the real equations and yet have relatively low numerical diffusion. In Section 5 we will consider the Lagrangian techniques incorporated in the SPLISH code, which uses a reconnectable grid of triangles to allow accurate numerical solution of complicated flows with breaking waves and other severe nonlinearities.

The continuity equation, as it appears in (2.1'), is written in conservation form. There are other ways to write the continuity equation which get reflected in some of the numerical solution techniques:

$$\text{Convection form: } \frac{\partial \rho}{\partial t} + \underbrace{\mathbf{v} \cdot \nabla \rho}_{\text{convection}} = \frac{d\rho}{dt} = - \underbrace{\rho \nabla \cdot \mathbf{v}}_{\text{compression}}; \quad (2.10)$$

$$\text{Integral form: } \frac{\partial}{\partial t} \int_{\text{region}} \rho d^3 \mathbf{r} = - \int_{\text{boundary}} \rho \mathbf{v} \cdot d\mathbf{A}. \quad (2.11)$$

The convective term $\mathbf{v} \cdot \nabla \rho$ displayed explicitly in (2.10) gives it its intrinsically hyperbolic form and causes severe problems numerically. The compression term $-\rho \nabla \cdot \mathbf{v}$ is absent in incompressible hydrodynamics applications. Without this term (2.9) is often called the convection or the advection equation.

The convective equation appears in almost all descriptions of dynamic physical systems simply because it expresses two of the general principles in physics, conservation and causality. The integral form given by (2.11) states that the amount of material ($\int \rho d^3r$) in a given region can only change due to flows of material ($\rho \mathbf{v} \cdot d\mathbf{A}$) through the boundary (conservation). Any material currently at a given point could only have gotten there by leaving someplace else (causality). Continuity equations underlie compressible and incompressible fluid dynamics, hydrodynamics, plasma physics, and quantum mechanics.

The convective equation displays the positivity property. A quantity being transported will never turn negative anywhere if it was everywhere positive to start with. Matter cannot be removed from a region which is devoid of matter to begin with. The fact that matter obeys particle and fluid-like equations on microscopic and macroscopic scales, respectively, has its counterpart for numerical solution techniques. The first three of the techniques described below model the system as an aggregate of discrete entities, while the last three aim more directly at solving the partial differential equation itself.

The convective equation is hyperbolic because it has first derivatives in both space and time, thus permitting propagating (wave) solutions. It is quite different from second-order wave equations, which are also hyperbolic. Therefore, the solution techniques can differ significantly. Generally the characteristics of the equation are in one direction only. Wave equations usually occur in pairs with two oppositely directed characteristics, and the two quantities being propagated (say $\mathbf{E}$ and $\mathbf{B}$ in electromagnetics) appear in the time derivative of one equation and in the space derivative of the other, respectively. Thus second-order wave equations have no positivity to speak of, and the most useful conservation property is quadratic rather than linear. Because of these distinctions, much of what is said here about convective equations does not apply to wave equations.

This brief survey considers only conservative solution techniques, although nonconservative techniques are adequate for some problems. In most situations the conservation properties of the physics have to be mirrored in the numerical method or else nonphysical instabilities or unacceptable secular errors will result. Although positivity is probably as important as conservation, not all of the methods generally applied have recognized this fact. The most common and annoying problems which arise in the solution of coupled systems of convective equations arise because the quantity convected becomes falsely negative somewhere. Frequently this is related to the occurrence of numerical oscillations in the vicinity of steep gradients and discontinuities.

The analysis here is limited to discussing the basic methods even though there are a host of hybrid techniques combining two or more of the basic approaches. The number of such combinations is staggering, and even the definitions of the basic methods are in some dispute. Therefore no useful purpose is served by trying to classify and analyze all of the possible combinations here. Rather, the general properties of each are described, and the pros and cons considered.

As a framework for evaluating the algorithms, six general requirements are presented which an ideal numerical algorithm for solving the convective equation should satisfy to be generally useful. These requirements are given in what we estimate to be their order of importance. An ideal algorithm should:

1. Be linearly stable for all cases of interest;
2. Mirror conservation properties of the physics;
3. Ensure the positivity property when appropriate;
4. Be reasonably accurate;
5. Be computationally efficient;
6. Be independent of specific properties of one particular application.

This last requirement restates that we are seeking generally useful methods and that all hybrid schemes cannot be discussed. As with the methods themselves, some of the requirements are related. Stability, conservation, and positivity generally relate to accuracy, for example, and yet no one of these can guarantee any of the others. On the other hand, the requirements can also be partially contradictory. Accuracy and efficiency, for example, often tug in opposite directions.

The following list shows the six methods to be discussed, the order being chosen to range from most discrete (or particulate) in nature to most continuous (or global).

- Quasiparticle Methods

1. Collisionless particles
2. Collisional particles for fluids

- Characteristic Methods

- Lagrangian Finite Difference Methods

- Eulerian Finite Difference Methods

1. Explicit vs implicit
2. Order vs accuracy

- Finite Element Methods

- Spectral Methods

Quasiparticle Methods

In their simplest form quasiparticle methods attempt to solve (2.9) by simulating the microscopic physics of the particles which make up the fluid. If a large number N of quasiparti-

cles are initialized in the calculation at positions $\{X_i(0)\}$ at time $t = 0$, their later positions can be found at time t by integrating the simple orbit equation

$$\frac{dX_i}{dt} = V_i(t), \quad (2.12)$$

assuming that the velocity of each particle is known as a function of time. Equation (2.12) is an ordinary differential equation and is usually solved easily with a high degree of accuracy. When each quasiparticle has a velocity which is determined solely as a function of the particle's position at any time, i.e.,

$$V_i(t) = V[X_i(t), t], \quad (2.13)$$

the quasiparticles cannot pass each other, except as a result of finite time-step errors [or perhaps singularities in $v(x, t)$]. The model is then appropriate for collisional fluids. Nonetheless the incompressibility condition, $\nabla \cdot v = 0$, is very difficult to enforce self-consistently and rigorously in these models. When several quasiparticles can occupy essentially the same location in space and yet be moving with significantly different velocities, the model is more appropriately used to describe "collisionless" systems such as plasmas on the microscopic scale or stars in self-gravitating systems on a somewhat grander scale.

The collisional quasiparticle methods which have been tried to date include PIC (Harlow, 1964), GAP (Marder, 1975), VORTEX (Christiansen, 1973), and more recent attempts of the same genre by workers at Los Alamos (Taggart et al., 1975). These methods introduce what amounts to a finite-difference grid for some of the derived physical quantities, such as pressure, while retaining the particle mass, momentum, and energy as variables. The collisionless quasiparticle methods are reviewed by Birdsall and Fuss (1969), by Langdon and Lasinski (1976), and in the *Proceedings of the Fourth Conference on Numerical Simulation of Plasma*, edited by Boris and Shanny (1970). The original applications seem to stem from work by Hockney (1965, 1966) and Buneman (1967).

The term quasiparticle has been used because it is almost never possible to carry in the computer memory as many particles as are involved in the actual physical problem of interest. It is not unusual to simulate systems of 10^{11} particles (galaxies) to 3×10^{22} particles (1 cm^3 of water) by using $10^3 - 10^6$ quasiparticles. In fact, when the number of quasiparticles exceeds 10^3 or so, it is usually impossible to treat the forces by calculating a direct interaction law. A potential equation is introduced which relates the source terms (pressure, charge, or mass) and the corresponding long-range fields. It is solved on a finite-difference grid in either configuration or transform space (see Section 6).

The quasiparticle methods are generally stable, and conserve where they should (almost by definition). These methods can also guarantee positivity since it is just as hard to remove a quasiparticle from an already empty region of space as it is a real particle. The methods can even be made reasonably general and flexible. Where they fail is in efficiency and accuracy. Because the statistics and hence the smoothness of the computed solutions depend on the number of quasiparticles which the user can afford, many applications of interest are beyond the particle approach. Long-time, turbulent incompressible flows are unbearably expensive.

Characteristic methods

Characteristic methods, which take advantage of the physical content of the specific system of equations being solved, are not entirely counter to our sixth requirement, because different characteristic methods can often be formulated for different sets of governing equations. The equations of ideal MHD (see e.g., Kulikovsky and Lyubimov, 1965) and ideal compressible flow (Moretti and Abbott, 1966), for example, can be formulated and solved by characteristics. These solutions are often the best that can be obtained, because the discontinuities which arise naturally in these physical systems are followed individually like the particles in the previous methods, and the continuous fluid behavior between the characteristics of the discontinuities is easy to represent. Nevertheless, even though conservation and positivity are relatively easy to ensure, characteristic methods are not generally applicable. The presence of even one diffusion term in the governing equations usually abrogates the use of characteristic methods, because the characteristics of the model cease to exist.

A further drawback to characteristic methods is their complexity and relative inefficiency for multidimensional flows. The characteristic interfaces are complicated lines in two dimensions and surfaces in three dimensions. These can become progressively more and more difficult to follow as the flows proceed in time. The relative inefficiency, which is not troublesome in simple flows where a few characteristics may be sufficient to describe the whole system, becomes far more serious when the new parallel and pipeline computers are contemplated. Since the characteristics, like quasiparticles, must each be considered individually, and since aspects of random access to computer memory as well as complicated logic are involved, the future of these methods on the newer computers is correspondingly dim.

Finally, characteristic methods are generally applied to compressible systems rather than to incompressible systems where the sound speed is effectively infinite. In the types of fluid flow problems being discussed and tackled here, these characteristic techniques reduce to the quasiparticle methods described above or to very similar algorithms.

Finite differences

We come now to a discussion of finite differences. Unlike the quasiparticle and characteristic methods, finite difference solutions of the convective equation are based on an approximation in which the continuous flow variables such as $\rho(\mathbf{x}, t)$ and $\mathbf{V}(\mathbf{x}, t)$ are discretized. A finite set of representative values $\{\rho_i\}$, $\{\mathbf{V}_i\}$ is defined at N distinct points in space called grid (or mesh) points. The desired continuous functions are assumed to be known only at the mesh points, and a whole culture of techniques has been developed to predict future values of $\{\rho_i\}$, etc., at some time t , given $\{\rho_i\}$ at $t = 0$. The term "finite differences" refers to the fact that the spatial derivatives (and usually the time derivative as well) are approximated by using only the grid point values, e.g.,

$$\left. \frac{\partial \rho}{\partial x} \right|_{x_i} \approx \frac{\rho_{i+1} - \rho_{i-1}}{x_{i+1} - x_{i-1}}. \quad (2.14)$$

The attraction of these finite-difference methods is their richness and simplicity. The classic text is by Richtmyer and Morton (1967), but the literature is virtually limitless.

Although a detailed classification of possible finite difference schemes is probably purposeless (there are too many hybrids) and may be impossible, a few of the choices should be discussed. The first of these is the distinction between Lagrangian and Eulerian methods. A Lagrangian finite-difference scheme (see, e.g., Brennan and Whitney, 1970; Crowley, 1971; Chan, 1974) is one in which the location of the grid points is allowed to move along with the fluid. This approach is useful because the convective term is transformed away in the moving system. Hence the positivity property is easily ensured, along with conservation. The trouble with this approach is that complicated flows in two or three dimensions will rather quickly distort the moving mesh. Much of the accuracy is lost, often before useful information can be extracted from the calculation.

Eulerian methods, on the other hand, keep the grid fixed in position, so that distortion of the mesh cannot occur. Since fluid is now flowing across a stationary and usually rather coarse mesh, truncation errors occur. These are secular in time and can play havoc with accuracy and positivity in a long run (Richtmyer and Morton, 1967; Wurtele, 1961; Roache, 1972; Boris and Book, 1973). Nevertheless, we will see that the problems with both the Eulerian and Lagrangian approaches can be largely overcome.

The second major finite difference choice is that of the time integration technique, in particular the controversial question of choosing an explicit or an implicit scheme. Implicit schemes for the convective derivatives involve, in the worst cases, gigantic matrix equations and in the best cases the inversion of a tridiagonal matrix. Thus their operation count and speed are appreciably worse than obtained using corresponding explicit schemes. The advantage of implicit schemes, which in some problems outweighs the dual disadvantages of program complexity and operation count per timestep, is the ability to take much longer finite-difference timesteps without exciting numerical modes of instability. The choice lies in the physics (mathematics) of the *specific* problem being solved. When the physical phenomenon of interest varies appreciably on a timescale τ , no calculation with $\Delta t \gg \tau$ can reasonably be claimed to reproduce the phenomenon accurately. In such cases computationally efficient explicit methods are called for, with $\Delta T < \tau$ for stability. In other words, *the stability condition for an explicit method is usually the accuracy condition for the corresponding implicit method.*

When the phenomenon of real interest is not the fastest (shortest timescale) effect in the problem, additional considerations are operative. Consider, for example, the expanding flow of compressed air out a whistle. The airflow speed is slow, so the fine-scale high-speed vibrations of the whistle sound are expected to have little effect on the slow (time averaged) macroscopic flow pattern, even though the energy emitted as sound may be comparable with the flow energy imparted to the air. Clearly an implicit scheme is desired, since the speed of sound is about two orders of magnitude larger than the flow speed, and the condition $\Delta t \leq \tau_{\text{flow}}$ is far better than the more expensive $\Delta t \leq \tau_{\text{sound}}$ found for explicit schemes. Unfortunately the problem is not yet resolved, because implicit schemes which are forward-differenced beyond second-order centered differencing tend to damp the undesired high-frequency sound waves strongly. Thus sonic energy from the whistle can get deposited as heat near the whistle by a conservative, forward-differenced implicit scheme. A centered scheme (or one quite near centered) would allow much of the sonic energy to propagate away as sound without heating the local air. The strongly dispersed short-wavelength sonic components would generally stay in the vicinity of the whistle far too long, however, and might eventually deposit their energy near the whistle anyway.

The best solution for incompressible hydrodynamics problems formulated in terms of the primitive equations involves developing a Poisson equation for the fluid pressure which in effect implicitly couples all of the grid points to obtain a formulation without sonic or compression waves. The pressure at any location is generally that value required to ensure that $\frac{d}{dt}(\nabla \cdot \mathbf{v}) \equiv 0$. Of course cavitation involves large changes in density because the pressure cannot go negative. Physically the $\nabla \cdot \mathbf{v} = 0$ condition relaxes before the $p \geq 0$ condition does. Within the Eulerian framework there are a number of ways to develop the implicit formulation. Leaving the treatment of the convective terms free for nonlinear monotonic integration (Harten, 1974; Van Leer, 1977, 1977a; Hain, 1977) is much more accurate than the linear nonpositive algorithms usually used. This method seems more suitable for the purpose discussed in this document.

Finite-element and spectral methods

Both finite-element (Strang and Fix, 1973; Gottlieb, 1977) and spectral (Gottlieb and Orszag, 1977) methods utilize continuum concepts fully in solving (2.9). The functions of interest, $\rho(\mathbf{x}, t)$ and $\mathbf{v}(\mathbf{x}, t)$, are expanded in a set of useful basis functions whose expansion coefficients are now determined from a complicated set of coupled, usually nonlinear, but ordinary differential equations. For finite elements (Strang and Fix, 1973), the basis functions are localized to a mesh cell or two, and the techniques of matching expansion function boundary conditions at cell boundaries remind one very much of using splines. The spectral methods generally employ a series of global basis functions (Gottlieb and Orszag, 1977; Orszag and Israel, 1972). Both methods employ projection to recover ordinary differential equations which are implicit in one way or another. The finite element methods acquire their implicitness by coupling of spatially adjacent basis functions via projection when evaluating the time derivative on the left-hand side of (2.9). The spectral methods become effectively implicit when convolutions are needed to evaluate, for example, the nonlinear $\mathbf{v} \partial \mathbf{v} / \partial \mathbf{x}$ term, where the velocity field is represented as a Fourier expansion.

The disadvantages of these two approaches are clear:

- They are slow per timestep compared with finite-difference methods for *general* problems.
- They are ill-suited (spectral methods more so than finite element) to problems displaying highly localized effects, such as density discontinuities at a thermocline.
- They are generally complicated and expensive to program, particularly so for additional nonlinear terms.

The advantage is far greater accuracy than is possible with simple difference or particle methods for smooth well-behaved flows and oscillations. The spectral methods have been described as "infinite-order" accurate and have been characterized as "optimal" for certain finite-difference evaluations. Whether or not the potential increased accuracy outweighs the drawbacks for real problems using either finite elements or spectral methods depends very strongly on the specific problem being solved. The degree to which these two classes of techniques satisfy or do not satisfy our requirements five and six may be subject to legitimate debate, but there is no doubt

that the positivity property of the continuity equation and accuracy cannot be guaranteed when expanding in a time-independent set of basis functions. The Gibbs phenomenon (Orszag and Israel, 1972) represents a minimal, or lower-bound error, which is only reduced by increased resolution. In a very real sense an uncertainty principle is at work. The numerical error can only be reduced by increasing the number of discrete degrees of freedom in the representation.

Before giving our evaluations, we would like to quote three conclusions, in a similar context, from another source (Kreiss, 1972):

- The spectral method has no stability problems but is much more complicated and slower than generalized difference methods.
- It is doubtful whether the finite-element method, based on piece-wise polynomials, can compete with the above methods.
- If Eulerian difference methods are used, they should be at least fourth-order accurate.

While we agree basically with these remarks, they do not really do justice to the quasiparticle schemes, nor do they adequately reflect a very important piece of personal experience. Whenever a theory, an algorithm, or a new mathematical technique is to be tested, attention rather quickly turns to the crucial yet simple conservation equations of ideal compressible flow.

Finite-difference methods have solved the transient Rankine-Hugoniot shock problem adequately, as have the various flavors of quasiparticle methods if enough particles are used. There seems to be no calculation, even in one spatial dimension, using either a finite-element or a spectral method which has correctly solved the problem of an ideal gas compressible shock. Because the breaking of internal waves and tidal bores bears a strong resemblance to one-dimensional shock phenomena, the analysis also applies by analogy to hydrodynamics problems.

Until such "shock" calculations become commonplace and computationally attractive, finite-difference methods would seem to have the inside track for general applications. Therefore, in the next section we take a closer look at the remaining deficiencies in Lagrangian and Eulerian finite-difference techniques. We return to the subject of spectral methods in Section 7.

3. FLUX-CORRECTED TRANSPORT

3.1 Improvements in Finite Difference Algorithms

There is undoubtedly some merit in trying to boost the performance of quasiparticle methods on the one hand and the basis-function expansion methods on the other toward the level obtained by finite differences. But it is a low-risk, high-return investment to patch up the obvious failings of the front-runners, finite differences. In Lagrangian methods, the major outstanding problems arise from secularly unattractive distortions of the grid which wreck calculations of interesting flows quite quickly. In Eulerian methods, the major outstanding weakness in a huge class of problems of real interest is the need for a large artificial damping (numerical diffusion) to fill in what would otherwise be pits of "negative density" in the calculated profiles. Since the "Eulerian" positivity problem is encountered even in Lagrangian calculations for many situations, it demands the largest share of attention.

The outstanding problem with Eulerian techniques for solving continuity equations is the competition between accuracy and positivity (Kreiss, 1972; Van Leer, 1974). Consider the rather general three-point approximation to (2.1):

$$\begin{aligned} \bar{\rho}_i = & \rho_i^n - \frac{1}{2} \left(\rho_{i+1}^n + \rho_i^n \right) \epsilon_{i+1/2} + \frac{1}{2} \left(\rho_i^n + \rho_{i-1}^n \right) \epsilon_{i-1/2} \\ & + \nu_{i+1/2} \left(\rho_{i+1}^n - \rho_i^n \right) - \nu_{i-1/2} \left(\rho_i^n - \rho_{i-1}^n \right), \end{aligned} \quad (3.1)$$

where $\epsilon_{i+1/2} \equiv v_{i+1/2} \delta t / \delta x_{i+1/2}$. Equation (2.16) is in finite-difference conservation form, with whole indices representing cell centers and half indices indicating cell interfaces. The additional numerical diffusion terms with diffusion coefficients $\nu_{i+1/2}$ have to be added to ensure positivity. The stability of (3.1) is ensured, at least roughly, when

$$\frac{1}{2} > \nu_{i+1/2} > \frac{1}{2} \epsilon_{i+1/2}^2. \quad (3.2a)$$

The upper limit arises from the explicit diffusion time-step condition, while the lower limit is the Lax-Wendroff damping coefficient (Boris and Book, 1976a). Unfortunately positivity is only ensured linearly when

$$\nu_{i+1/2} > \frac{1}{2} \left| \epsilon_{i+1/2} \right|, \quad (3.2b)$$

the first-order upstream-centered scheme result.

The escape route is signaled in the preceding sentence by the word "linearly." By relaxing the linearity implied by (3.1) and letting the diffusion coefficients be nonlinear functionals of the flow velocities $\{\epsilon_{i+1/2}\}$, we can hope to reduce the integrated dissipation below the rather ghastly limit (3.2b) and yet retain sufficient dissipation near steep gradients to ensure positivity. Boris and Book (1973) introduced this basic nonlinear approach with the techniques of Flux-Corrected Transport. A literature is beginning to form (Boris and Book, 1976; Van Leer, 1974, 1976, 1976a; Harten, 1974; Book, Boris and Hain, 1975; Hain, 1978) about these "monotonic difference schemes" since the dilemma of accuracy versus positivity in Eulerian difference schemes can be resolved in no other way. Several different approaches are possible, and the area is still largely unexplored.

Figure 3-1 shows a comparison of four common difference schemes solving the standard square wave advection problem. The effects of excess numerical damping in the donor-cell treatment (upstream centered first order), and of excess dispersion in the leapfrog and Lax-Wendroff treatments are clearly visible. Dispersion manifests itself as a trail or projection of oscillations in the computed solution near discontinuities and sharp gradients of the "correct" solution. The two second-order algorithms yield results which are almost indistinguishable. In Fig. 3-1a we have a result typical of those obtained from a first-order algorithm, even in more general contexts, when sharp gradients arise. Conversely, the results of Figs. 3-1b and 3-1c typify the dispersive errors to be expected in second-order treatments. The results obtained by using SHASTA, the fourth algorithm tested, show a striking improvement. The "discontinuities" are resolved to within a mesh space or two. At the same time, gentle gradients ("continuous" portions of the profile) are propagated with second-order accuracy. Thus the FCT algorithm is able to handle shocks and steep concentration gradients with no apparent penalty in regions where they are absent. Here and in the next section we discuss how this is achieved.

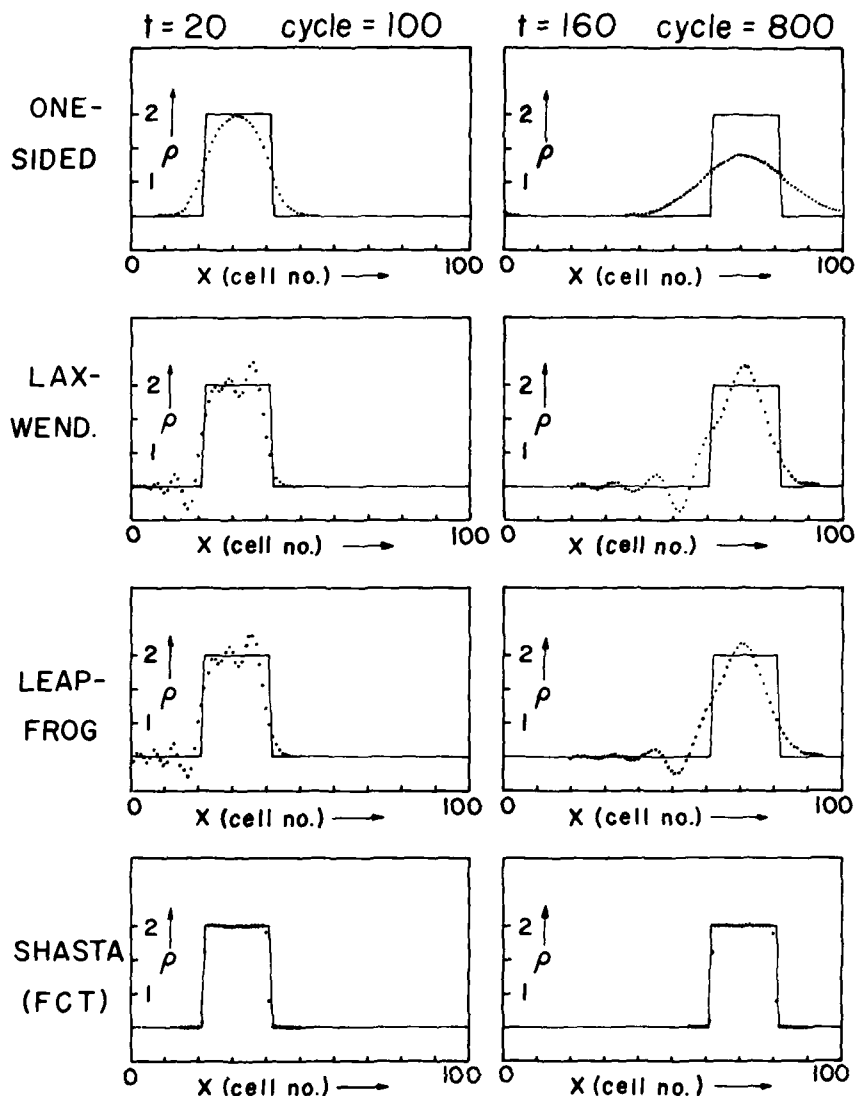


Fig. 3-1 — Comparison of four difference schemes solving the square wave problem. The merits of Flux-Corrected Transport relative to the other methods are clear.

The calculation in Fig. 3-1d was performed by the first FCT algorithm SHASTA and had an error about four or five times smaller than the simple linear methods also shown. The damping was second order, as were the relative phase errors. The basic technique was quickly generalized to cylindrical and spherical systems, to Lagrangian as well as fixed Eulerian grids, and was applied to a number of one-, two-, and three-dimensional problems. More recent work has been devoted toward extending the basic nonlinear flux-correction techniques to convection algorithms other than SHASTA and toward discovering an "optimum" FCT algorithm (Book, Boris and Hain, 1975; Boris and Book, 1976, 1976a).

This latter effort came to several conclusions:

1. There is an error inherent in finite-difference techniques which arises solely from the finite resolution of the grid and which does not vanish even when both numerical dispersion and numerical dissipation vanish. This error is two to three times smaller than achieved with the original FCT algorithms.
2. In local algorithms where dispersion and diffusion errors are present also, the residual dispersion errors seem to be more severe. Therefore, the optimal FCT algorithm was found to be one which has the diffusion coefficients $\nu_{j+1/2} = \frac{1}{6} + \frac{\epsilon_{j+1/2}^2}{3}$, driving dispersion errors from second to fourth order (cf. Kreiss, 1972).
3. The best of the generally useful FCT variants was almost twice as accurate as the original FCT algorithm and within 50% of the nonzero minimal error. This is roughly six to eight times more accurate than the old methods.
4. Implicit convection algorithms, in addition to being somewhat slower than the local explicit methods to compute, also have bad nonlinear properties. Information transfer occurs faster than fluid characteristics, so shocks are relatively poorly treated.
5. Aside from optimizing phase errors, large diffusion/antidiffusion coefficients $\nu_{j+1/2} \geq 1/8$ are necessary in order to provide enough local dissipation at shocks to satisfy the Rankine-Hugoniot conditions. This requirement is related to the one which motivated the introduction of artificial viscosity in earlier algorithms (cf. Richtmyer and Morton, 1967).

Since the latest FCT algorithms have eliminated 90-95% of the *removable* error and the removable error that remains is barely half of the irreducible error, it is natural to develop algorithms aimed at optimization in speed, flexibility, and generality.

In FCT algorithms, the basic convective transport algorithm is augmented with a strong enough diffusion to ensure positivity at the expense of excess smoothing. Since the amount of diffusion which has been added is known, FCT then performs a conservative antidiffusion step to remove the diffusion in excess of the stability limit. To preserve monotonicity, however, the antidiffusive fluxes are effectively multiplied by a coefficient which ranges from zero to unity. The criterion for choosing the reduction factors of the antidiffusive fluxes is that the antidiffused solution exhibit no new maxima or minima where the diffused solution had none. Thus positivity (or more generally, monotonicity) is built in.

Although the limit (3.2b) represents the minimum amount of diffusion needed for stability, FCT algorithms generally use a larger zero-order diffusion because it has been found that the correct choice of the $\{\nu_{j+1/2}\}$ within the monotonic and stable range will reduce convective phase errors from second to fourth order as suggested by Kreiss (1972). Since the antidiffusion can also be chosen correspondingly larger, no real price is exacted for this improvement in phase properties. The most recent efforts have taken advantage of this fact to generate minimum-operation-count FCT algorithms for Cartesian, cylindrical, and spherical coordinate systems with stationary (Eulerian) and movable (Lagrangian) grid systems (Boris, 1976). Because these algorithms, particularly the nonlinear flux-correction formula, were very carefully designed, they are fully "vectorizable" for pipeline and parallel processing. The execution time

per continuity equation per grid point is roughly $1.3 \mu\text{sec}$ on the Texas Instruments ASC at NRL. A complete 2D calculation on a system of 200 grid points $\times$ 200 grid points requires roughly 2 to 2.5 seconds per timestep depending on extra physics and boundary conditions incorporated in the problem.

3.2 Multidimensional FCT

The most advanced method to date for solution of convective equations in multidimensions is that of Zalesak (1978). To understand it, let us review the essentials of the FCT method. In 1D it consists of six sequential operations at each interior point x_i :

1) Compute $F_{i+1/2}^L$, the transportive flux given by some low-order scheme guaranteed to give monotonic (ripple-free) results for the problem at hand.

2) Compute $F_{i+1/2}^H$, the transportive flux given by some high-order scheme.

3) Define the "antidiffusive flux":

$$A_{i+1/2} \equiv F_{i+1/2}^H - F_{i+1/2}^L.$$

4) Compute the updated low-order ("transported and diffused") solution:

$$w_i^{td} = w_i^n - \Delta x_i^{-1} \left[F_{i+1/2}^L - F_{i-1/2}^L \right].$$

5) *Limit* the $A_{i+1/2}$ in a manner such that w^{n+1} as computed in step 6 below is free of overshoots and undershoots:

$$A_{i+1/2}^C = C_{i+1/2} A_{i+1/2}, \quad 0 \leq C_{i+1/2} \leq 1.$$

6) Apply the limited antidiffusive fluxes:

$$w_i^{n+1} = w_i^{td} - \Delta x_i^{-1} \left[A_{i+1/2}^C - A_{i-1/2}^C \right].$$

The critical step in the above is, of course, step 5, which will be discussed shortly. In the absence of the flux limiting step ($A_{i+1/2}^C = A_{i+1/2}$), w_i^{n+1} would simply be the time-advanced, high-order solution. We note that this definition of FCT is considerably more general than that given originally by Boris and Book (1973).

Before proceeding to a discussion of flux limiting, let us see how the procedure given above might be implemented in multidimensions. An obvious choice would be to use a Strang-type time-splitting procedure [see Gottlieb (1972)] when it can be shown that the equations allow such a technique to be used without serious error. Such a procedure may even be preferable from programming and time-step considerations. However, there are many problems for which time-splitting produces unacceptable numerical results, among which are those involving incompressible or nearly incompressible flow fields. This technique, which is straightforward, will not be discussed here. Instead we consider as an example the fully two-dimensional equation

$$w_i + f_i + g_i = 0, \quad (3.3)$$

where w , f , and g are functions of x , y , and t . In finite difference flux form we have

$$w_{i,j}^{n+1} = w_{i,j}^n - \Delta V_{i,j}^{-1} \left[F_{i+1/2,j} - F_{i-1/2,j} + G_{i,j+1/2} - G_{i,j-1/2} \right], \quad (3.4)$$

where now w , f , and g are defined on spatial grid points x_i , y_j at time levels t^n , and $\Delta V_{i,j}$ is a two-dimensional area element centered on grid point (i, j) . Now there are two sets of transportive fluxes F and G , and the FCT algorithm proceeds as before:

- 1) Compute $F_{i+1/2,j}^L$ and $G_{i,j+1/2}^L$ by a low-order monotonic scheme.
- 2) Compute $F_{i+1/2,j}^H$ and $G_{i,j+1/2}^H$ by a high-order scheme.
- 3) Define the antidiffusive fluxes:

$$A_{i+1/2,j} \equiv F_{i+1/2,j}^H - F_{i+1/2,j}^L;$$

$$A_{i,j+1/2} \equiv G_{i,j+1/2}^H - G_{i,j+1/2}^L.$$

- 4) Compute the low-order time-advanced solution:

$$w_{i,j}^{td} = w_{i,j}^n - \Delta V_{i,j}^{-1} \left[F_{i+1/2,j}^L - F_{i-1/2,j}^L + G_{i,j+1/2}^L - G_{i,j-1/2}^L \right].$$

- 5) Limit the antidiffusive fluxes:

$$A_{i+1/2,j}^C = A_{i+1/2,j} C_{i+1/2,j}, \quad 0 \leq C_{i+1/2,j} \leq 1;$$

$$A_{i,j+1/2}^C = A_{i,j+1/2} C_{i,j+1/2}, \quad 0 \leq C_{i,j+1/2} \leq 1.$$

- 6) Apply the limited antidiffusive fluxes:

$$w_{i,j}^{n+1} = w_{i,j}^{td} - \Delta V_{i,j}^{-1} \left[A_{i+1/2,j}^C - A_{i-1/2,j}^C + A_{i,j+1/2}^C - A_{i,j-1/2}^C \right].$$

As can be easily seen, implementation of FCT in multidimensions is straightforward with the exception of step 5.

We describe now in one spatial dimension a flux-limiting algorithm (Zalesak, 1978) which generalizes easily to multidimensions and which, even in one dimension, exhibits a superiority with regard to peaked profiles over the limiter described by Boris and Book (1973).

We seek to limit the antidiffusive flux $A_{i+1/2}$ such that

$$A_{i+1/2}^C = C_{i+1/2} A_{i+1/2}, \quad 0 \leq C_{i+1/2} \leq 1, \quad (3.5)$$

and such that $A_{i+1/2}^C$ acting in concert with $A_{i-1/2}^C$ will not allow

$$w_i^{n+1} = w_i^{td} - \Delta x_i^{-1} \left[A_{i+1/2}^C - A_{i-1/2}^C \right]$$

to exceed some maximum value $w_i^{\max}$ nor fall below some minimum value $w_i^{\min}$. We leave the determination of $w_i^{\max}$ and $w_i^{\min}$ until later.

We define three quantities:

$$\begin{aligned} P_i^+ &= \text{the sum of all antidiffusive fluxes into grid point } i \\ &= \max(0, A_{i-1/2}) - \min(0, A_{i+1/2}). \end{aligned} \quad (3.6)$$

$$Q_i^+ = (w_i^{\max} - w_i^{td}) \Delta x_i. \quad (3.7)$$

$$R_i^+ = \begin{cases} \min(1, Q_i^+/P_i^+), & P_i^+ > 0 \\ 0, & P_i^+ = 0 \end{cases}. \quad (3.8)$$

Provided that $w_i^{\max} \geq w_i^{td}$ (it *must* be), all three of the above quantities are positive and R_i^+ represents the least upper bound on the fraction which must multiply all antidiffusive fluxes into grid point i to guarantee no overshoot at grid point i .

Similarly we define three corresponding quantities:

$$\begin{aligned} P_i^- &= \text{the sum of all antidiffusive fluxes away from grid point } i \\ &= \max(0, A_{i+1/2}) - \min(0, A_{i-1/2}). \end{aligned} \quad (3.9)$$

$$Q_i^- = (w_i^{td} - w_i^{\min}) \Delta x_i. \quad (3.10)$$

$$R_i^- = \begin{cases} \min(1, Q_i^-/P_i^-), & P_i^- > 0 \\ 0, & P_i^- = 0 \end{cases}. \quad (3.11)$$

Again assuming that $w_i^{\min} \leq w_i^{td}$, we find that R_i^- represents that least upper bound on the fraction which must multiply all antidiffusive fluxes away from grid point i to guarantee no undershoot at grid point i .

Finally we observe that all antidiffusive fluxes are directed away from one grid point and into an adjacent one. Limiting will therefore take place with respect to undershoots for the former and with respect to overshoots for the latter. A guarantee that neither event comes to pass demands our taking a minimum:

$$C_{i+1/2} = \begin{cases} \min(R_{i+1}^+, R_i^-), & A_{i+1/2} \geq 0 \\ \min(R_i^+, R_{i+1}^-), & A_{i+1/2} < 0 \end{cases}. \quad (3.12)$$

Furthermore, we shall call upon our previously described experience with the original flux limiter and set

$$A_{i+1/2} = 0$$

if

$$A_{i+1/2} (w_{i+1}^{td} - w_i^{td}) < 0,$$

and either

$$A_{i+1/2} (w_{i+2}^{td} - w_{i+1}^{td}) < 0$$

or

$$A_{i+1/2} (w_i^{id} - w_{i-1}^{id}) < 0 \quad (3.13)$$

otherwise. In practice the effect of (3.13) is minimal and is primarily cosmetic in nature. This is because cases of antidiffusive fluxes directed down gradients in w^{id} are rare, and even when they occur usually involve flux magnitudes that are small compared to adjacent fluxes. If (3.13) is used, it should be applied *before* (3.5)-(3.12).

We come now to a determination of the quantities $w_i^{\max}$ and $w_i^{\min}$ in (3.7) and (3.10). A safe choice is

$$w_i^{\max} = \max(w_{i-1}^{id}, w_i^{id}, w_{i+1}^{id}); \quad (3.14)$$

$$w_i^{\min} = \min(w_{i-1}^{id}, w_i^{id}, w_{i+1}^{id}). \quad (3.15)$$

This choice will produce results identical with those of the Boris-Book (1973) formulation of step 5 in one dimension, including the occurrence of the "clipping" phenomenon to be mentioned shortly.

A better choice is:

$$w_i^a = \max(w_i^n, w_i^{id});$$

$$w_i^{\max} = \max(w_{i-1}^a, w_i^a, w_{i+1}^a). \quad (3.16)$$

$$w_i^b = \min(w_i^n, w_i^{id});$$

$$w_i^{\min} = \min(w_{i-1}^b, w_i^b, w_{i+1}^b). \quad (3.17)$$

This choice allows us to look back to the previous time step for upper and lower bounds on w_i^{n+1} .

It is clear that these two methods of determining $w_i^{\max}$ and $w_i^{\min}$ represent only a small sample of possible methods. The alternative flux limiter described in (3.5)–(3.13) admits of any physically motivated upper and lower bound on w_i^{n+1} supplied by the user, introducing a flexibility unavailable with the original flux limiter.

The alternative flux limiting algorithm just presented generalizes trivially to any number of dimensions. For the sake of completeness we present here the algorithm for two spatial dimensions.

We seek to limit the antidiffusive fluxes $A_{i+1/2,j}$ and $A_{i,j+1/2}$ such that

$$A_{i+1/2,j}^C = C_{i+1/2,j} A_{i+1/2,j}, \quad 0 \leq C_{i+1/2,j} \leq 1; \quad (3.18)$$

$$A_{i,j+1/2}^C = C_{i,j+1/2} A_{i,j+1/2}, \quad 0 \leq C_{i,j+1/2} \leq 1,$$

and such that $A_{i+1/2,j}^C$, $A_{i,j+1/2}^C$, and $A_{i,j-1/2}^C$ acting in concert shall not cause

$$w_{i,j}^{n+1} = w_{i,j}^{id} - \Delta V_{i,j}^{-1} \left[A_{i+1/2,j}^C - A_{i-1/2,j}^C + A_{i,j+1/2}^C - A_{i,j-1/2}^C \right]$$

to exceed some maximum value $w_{i,j}^{\max}$ or to fall below some minimum value $w_{i,j}^{\min}$.

Again we compute six quantities completely analogous to those computed in (3.6) through (3.11):

$$P_{i,j}^+ = \text{the sum of all antidiffusive fluxes into grid point } (i, j) \quad (3.19)$$

$$= \max(0, A_{i-1/2,j}) - \min(0, A_{i+1/2,j}) + \max(0, A_{i,j-1/2}) - \min(0, A_{i,j+1/2}).$$

$$Q_{i,j}^+ = (w_{i,j}^{\max} - w_{i,j}^{td}) \Delta V_{i,j} \quad (3.20)$$

$$R_{i,j}^+ = \begin{cases} \min(1, Q_{i,j}^+/P_{i,j}^+), & P_{i,j}^+ > 0 \\ 0, & P_{i,j}^+ = 0 \end{cases} \quad (3.21)$$

$$P_{i,j}^- = \text{the sum of all antidiffusive fluxes away from grid point } (i, j)$$

$$= \max(0, A_{i+1/2,j}) - \min(0, A_{i-1/2,j}) + \max(0, A_{i,j+1/2}) - \min(0, A_{i,j-1/2}). \quad (3.22)$$

$$Q_{i,j}^- = (w_{i,j}^{td} - w_{i,j}^{\min}) \Delta V_{i,j}. \quad (3.23)$$

$$R_{i,j}^- = \begin{cases} \min(1, Q_{i,j}^-/P_{i,j}^-), & P_{i,j}^- > 0 \\ 0, & P_{i,j}^- = 0 \end{cases} \quad (3.24)$$

Equation (3.12) becomes

$$C_{i+1/2,j} = \begin{cases} \min(R_{i+1,j}^-, R_{i,j}^+), & A_{i+1/2,j} < 0 \\ \min(R_{i,j}^-, R_{i+1,j}^+), & A_{i+1/2,j} \geq 0 \end{cases} \quad (3.25a)$$

$$C_{i,j+1/2} = \begin{cases} \min(R_{i,j+1}^-, R_{i,j}^+), & A_{i,j+1/2} < 0 \\ \min(R_{i,j}^-, R_{i,j+1}^+), & A_{i,j+1/2} \geq 0 \end{cases} \quad (3.25b)$$

while (3.13) becomes

$$A_{i+1/2,j} = 0$$

if

$$A_{i+1/2,j} (w_{i+1,j}^{td} - w_{i,j}^{td}) < 0,$$

and either

$$A_{i+1/2,j} (w_{i+2,j}^{td} - w_{i+1,j}^{td}) < 0$$

or

$$A_{i+1/2,j} (w_{i,j}^{td} - w_{i-1,j}^{td}) < 0. \quad (3.26a)$$

Likewise,

$$A_{i,j+1/2} = 0$$

if

$$A_{i,j+1/2} (w_{i,j+1}^{td} - w_{i,j}^{td}) < 0,$$

and either

$$A_{i,j+(1/2)} (w_{i,j+2}^{td} - w_{i,j+1}^{td}) < 0$$

or

$$A_{i,j+(1/2)} (w_{i,j}^{td} - w_{i,j-1}^{td}) < 0. \quad (3.26b)$$

and (3.16) and (3.17) become

$$w_{i,j}^a = \max(w_{i,j}^n, w_{i,j}^{td}),$$

$$w_{i,j}^{\max} = \max(w_{i-1,j}^a, w_{i,j}^a, w_{i+1,j}^a, w_{i,j-1}^a, w_{i,j+1}^a). \quad (3.27)$$

$$w_{i,j}^b = \min(w_{i,j}^n, w_{i,j}^{td}),$$

$$w_{i,j}^{\min} = \min(w_{i-1,j}^b, w_{i,j}^b, w_{i+1,j}^b, w_{i,j-1}^b, w_{i,j+1}^b). \quad (3.28)$$

Again, the effect of (3.26) is minimal, but if it is used it should be applied before (3.18)–(3.25). Note that our search for $w_{i,j}^{\max}$ and $w_{i,j}^{\min}$ now extends over both coordinate directions. Where finite gradients exist in both directions, this procedure will allow us to stop the clipping phenomenon in regions where a peak exists with respect to one coordinate direction but not in the other.

As a test problem, consider solid body rotation (see also Forester, 1977). That is, we have (3.3) with $f = wv_x$, $g = wv_y$, $v_x = -\Omega(y - y_0)$, and $v_y = \Omega(x - x_0)$. Here Ω is the (constant) angular velocity in rad/sec and (x_0, y_0) is the position of the axis of rotation. The configuration is shown in Fig. 3-2. The computational grid is 100×100 cells, $\Delta x = \Delta y$, with counterclockwise rotation taking place about grid point (50, 50). Centered at grid point (50, 75) is a cylinder of radius 15 grid points, through which a slot has been cut of width 5 grid points. The time step and rotational speed are chosen such that 628 time steps will effect one complete revolution of the cylinder about the central point. A perspective view of the initial conditions is shown in Fig. 3-3.

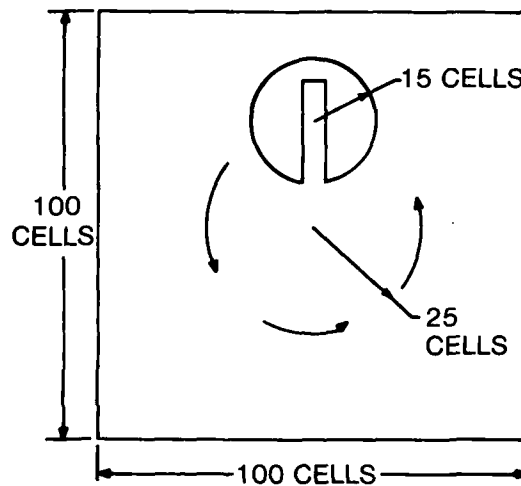


Fig. 3-2 — Schematic representation of two-dimensional solid body rotation problem. Initially w inside the cut-out cylinder is 3.0, while outside $w = 1.0$. The rotational speed is such that one full revolution is effected in 628 cycles. The width of the gap separating the two halves of the cylinder, as well as the maximum extent of the "bridge" connecting the two halves, is 5 cells.

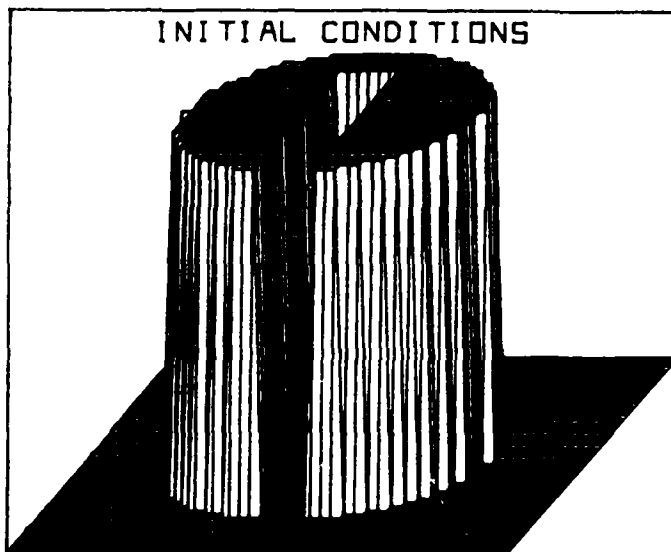


Fig. 3-3 — Perspective view of initial conditions for the two dimensional solid body rotation problem. Note that only a 50×50 portion of the mesh centered on the cylinder is displayed. Grid points inside the cylinder have $w_{i,j} = 3.0$. All others have $w_{i,j} = 1.0$.

Our high-order scheme for the following tests is a fully two-dimensional, fourth-order in space, second-order in time leapfrog-trapezoidal scheme, the leapfrog step of which is a two-dimensional fourth-order Kreiss-Oliger (1972) scheme. The low-order scheme is simply two-dimensional donor cell plus a two-dimensional zeroth order diffusion term with diffusion coefficient $1/8$.

In Fig. 3-4 we show a perspective view of the two calculations after $1/4$ revolution (157 iterations). Figure 3-5 presents a comparison of the results of the two calculations for one full revolution (628 cycles). Two features are obvious. The first is a much greater filling-in of the slot with the time-split (3.15) than with the fully two-dimensional flux limiter. The second is the loss of the bridge connecting the two halves of the cylinder in the case of the time-split application of (3.15). Less obvious is the lack of clipping of the peaked profiles defining the front surface of the cylinder for the case of the fully multidimensional limiter. Clearly this is due to the fact that the multidimensional flux limiter can look in both directions to determine whether or not a genuine maximum exists. Note that there are two factors working in favor of the fully multidimensional flux limiter: (1) this ability to look in both directions to find minima and maxima; and (2) the ability to scan both $w_{i,j}^n$ and $w_{i,j}^d$ to find maxima and minima. Both of these factors are responsible for the improved profiles.

A physical problem treated by this technique which is mathematically close to that of the vorticity dynamics formulations is that of barium cloud striations. A two dimensional ($x-y$) plasma cloud initialized in a region of constant magnetic field $\mathbf{B}_0$ directed along the z axis, with an externally imposed electric field $\mathbf{E}_0$ directed along the x axis, will tend to drift in the $\mathbf{E}_0 \times \mathbf{B}_0$ direction (along the negative y axis). If the ion-neutral collision frequency is finite,

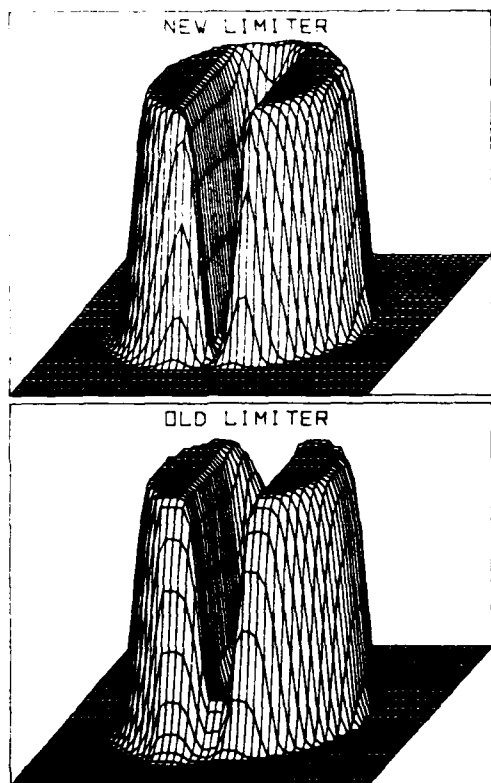
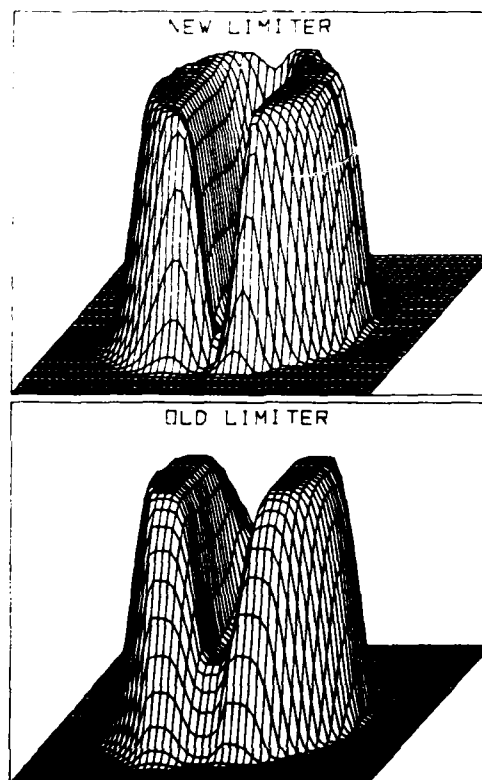


Fig. 3-4 — Comparison of perspective views of the w profile after 157 iterations (1/4 revolution) with both the old and new flux limiters. The perspective view has been rotated with the cylinder, so that direct comparison with Fig. 3-2 can be made. Again we plot only the 50×50 grid centered on the analytic center of the cylinder. Features to compare are the filling-in of the gap, erosion of the "bridge," and the relative sharpness of the profiles defining the front surface of the cylinder.

Fig 3-5 — Same as Fig. 3-4, but after 628 iterations (one full revolution). Again note decreased diffusion with new flux limiter.



Pedersen conductivity effects will produce polarization fields which tend to shield the inner (more dense) regions of the cloud from E_o , causing this inner portion of the cloud to drift more slowly than the outer portions of the cloud. This results in a steepening of gradients on the back side of the cloud. Arguments similar to those above, applied to infinitesimal perturbations imposed upon this back side gradient, show that the back side of the cloud is physically unstable to perturbations along $\hat{x}$. For a detailed description of this problem, see Scannapieco et al. (1976).

Figures 3-6 to 3-10 show isodensity contours of N_e/N_o for the above configuration at various times in the integration. It is seen that, as expected, the back of the cloud (the upper half in the plots) is unstable, growing linearly in the very early stages of development. Non-linear effects soon enter the physics, however, as each striation successively bifurcates, producing smaller and smaller scale structures, in agreement with the results of the ionospheric barium cloud releases which we are attempting to model. Two points which bear on the numerics should be noted: (1) the intense gradients dictated by the physics are *not* diffused away, nor do there appear in the problem any of the "ripples" associated with numerical dispersion which normally appear when steep gradients try to form; (2) precisely because we did not have to resort to time-splitting, none of the usual time-splitting phenomena, such as temporal density oscillations and spurious density values, are evident.

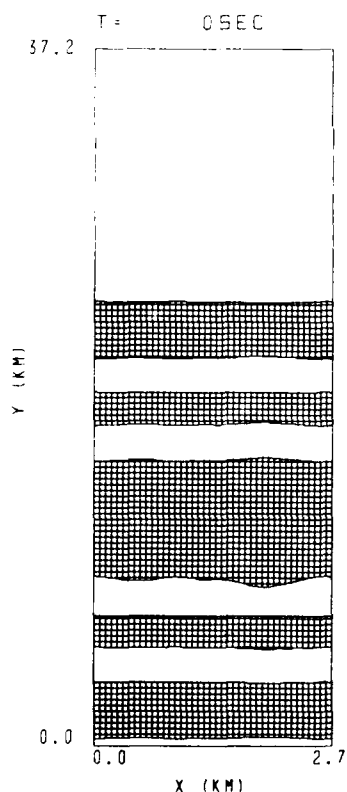


Fig. 3-6 — Isodensity contours of plasma density at $t = 0$ sec. The initial distribution for N_e/N_o is a Gaussian in y , centered at $y = 12.1$ km, plus a small random perturbation in x . Contours are drawn for $N_e/N_o = 1.5, 3.5, 5.5, 7.5$, and 9.5 . The area between every other contour line is cross-hatched. Only 120 of the 160 cells actually used in the y direction are displayed. Boundary conditions are periodic in both directions. In our plot B_o is toward the reader, and E_o is directed toward the right, and we have placed ourselves in a frame moving with the $(c/|B_o|^2) E_o \times B_o$. The upper portion of the Gaussian is physically unstable to perturbations, while the lower half is (linearly) stable.

NRL MEMORANDUM REPORT 4095

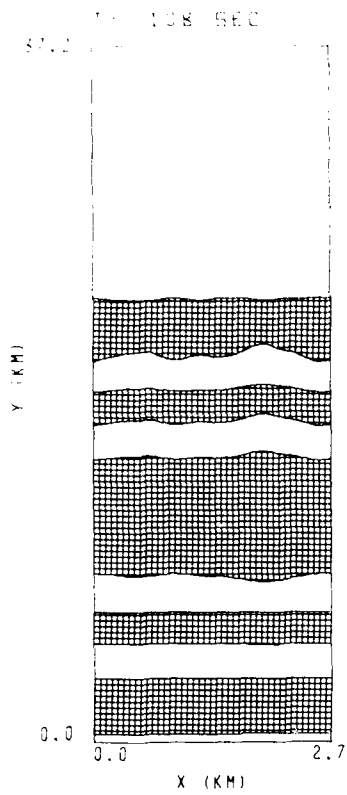
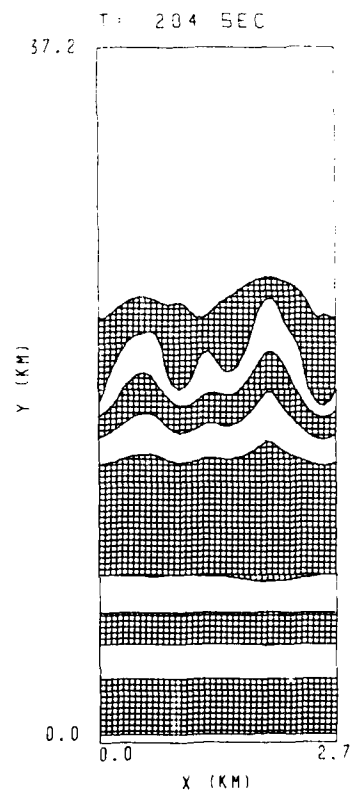


Fig. 3-7 — Same as Fig. 3-6, but for $t = 108$ sec. Note slow linear growth on unstable side.

Fig. 3-8 — Same as Fig. 3-6, but for $t = 204$ sec. Growth is now much more rapid, and we are entering a highly nonlinear regime.



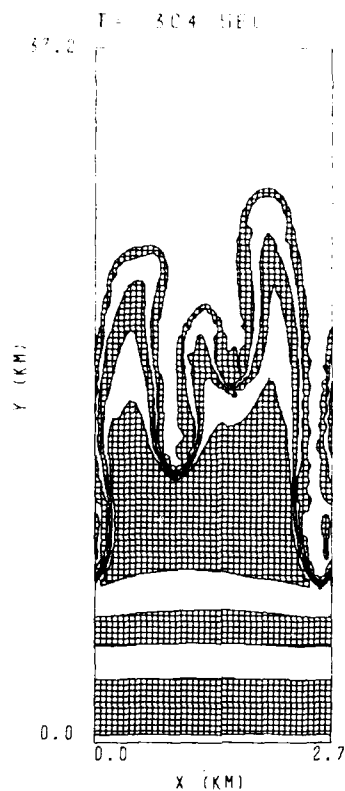
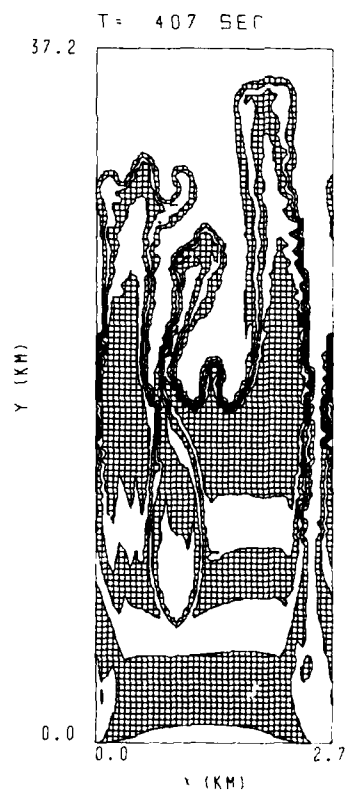


Fig. 3-9 — Same as Fig. 3-6, but for $t = 304$ sec. Development is fully nonlinear, as the intense gradients and associated high Fourier wave numbers become apparent.

Fig. 3-10 — Same as Fig. 3-6, but for $t = 407$ sec. Several plasma bifurcations are apparent, in agreement with the experimental results from ionospheric barium cloud releases, and we have maximum to minimum density variations resolved over only 2 cells.



On NRL's Texas Instruments ASC computer, the test problem calculations required 93 seconds and 125 seconds of CPU time for the time-split and fully multidimensional cases respectively, a cost penalty of slightly more than 30% for the multidimensional limiter. Of course this extra cost is highly problem dependent. For instance, the striations code spends 80% of its time solving the Poisson equation, making the net cost penalty of fully multidimensional flux limiting only a few percent.

4. A MULTILEVEL SEMI-IMPLICIT NUMERICAL MODEL FOR BAROCLINIC OCEANS

Madala and Piacsek (1977) have developed a three-dimensional Eulerian model of a stratified incompressible ocean. Their interest lay in keeping free-surface effects (air-sea interactions) in their baroclinic hurricane model. Accordingly, they retained the Coriolis terms in their hydrostatic model. Here we review the description given by Madala and Piacsek (1977), with emphasis on those features which are novel or of unusual computational interest.

The motivation to study surface effects without the severe timestep limitation imposed by fast surface waves resulted in the development of a code in which the baroclinic and barotropic (vertically averaged) modes are separated, with the surface waves coupled to the latter. The baroclinic modes are then treated explicitly and the barotropic waves implicitly. In contrast with previous semi-implicit models (e.g., O'Brien and Hurlburt, 1972), where a Helmholtz equation in an appropriate variable had to be solved in each layer, the technique of Madala and Piacsek (1977) leads to only a single Helmholtz equation for the surface height. As many layers can be introduced (usually 10-30) as are necessary to resolve the thermocline and neighboring levels.

4.1 Physical Model

The relevant equations of momentum and diffusion can be written

$$\frac{Du}{Dt} = fv - \frac{1}{\rho_0} \frac{\partial P}{\partial x} + \frac{\partial}{\partial x} K_H \frac{\partial u}{\partial x} + \frac{\partial}{\partial y} K_H \frac{\partial u}{\partial y} + \frac{\partial}{\partial z} K_V \frac{\partial u}{\partial z}, \quad (4.1)$$

$$\frac{Dv}{Dt} = -fu - \frac{1}{\rho_0} \frac{\partial P}{\partial y} + \frac{\partial}{\partial x} K_H \frac{\partial v}{\partial x} + \frac{\partial}{\partial y} K_H \frac{\partial v}{\partial y} + \frac{\partial}{\partial z} K_V \frac{\partial v}{\partial z}, \quad (4.2)$$

$$\frac{\partial P}{\partial z} = -pg, \quad (4.3)$$

$$\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z} = 0, \quad (4.4)$$

$$\frac{DT}{Dt} = \frac{\partial}{\partial x} A_H \frac{\partial T}{\partial x} + \frac{\partial}{\partial y} A_H \frac{\partial T}{\partial y} + \frac{\partial}{\partial z} A_V \frac{\partial T}{\partial z}, \quad (4.5)$$

$$\frac{Ds}{Dt} = \frac{\partial}{\partial x} B_H \frac{\partial s}{\partial x} + \frac{\partial}{\partial y} B_H \frac{\partial s}{\partial y} + \frac{\partial}{\partial z} B_V \frac{\partial s}{\partial z}, \quad (4.6)$$

$$\frac{\partial h}{\partial t} = -H \left\{ \frac{\partial [u]}{\partial x} + \frac{\partial [v]}{\partial y} \right\} - \left\{ \frac{\partial}{\partial x} (u_1 h) + \frac{\partial}{\partial y} (v_1 h) \right\}. \quad (4.7)$$

Here K_H and K_V are horizontal and vertical eddy diffusivities for momentum, A_H , A_V , and B_H , B_V are the corresponding eddy diffusivities for temperature and salinity, and square brackets denote vertical averages, i.e., for an arbitrary fluid variable q , $[q] \equiv (1/H) \int_0^H dz q(z)$. In

these equations h is the height of the free surface above (or below) the mean depth H of the undisturbed ocean. The rest of the symbols have their usual meaning.

The equation of state is taken in the form

$$\rho = \frac{P_1 + P_0}{1.000027(\lambda + \alpha_0(P' + P_0))} \quad (4.8a)$$

where

$$\alpha_0 = 0.698, \quad (4.8b)$$

$$\lambda = 1779.5 + 11.25 T_1 - 0.0745(T_1)^2 - (3.8 + 0.01 T_1) S, \quad (4.8c)$$

$$P_1 = 5890 + 38 T_1 - 0.375 T_1^2 + 3 S, \quad (4.8d)$$

with P_0 being the total pressure and T_1 the temperature deviation in units of atmospheres and degrees Celsius, respectively.

Equations (4.1) and (4.2) can be rewritten as

$$\frac{\partial u}{\partial t} - f v = -g \frac{\partial h}{\partial x} - \frac{1}{\rho_0} \frac{\partial P_a}{\partial x} + A, \quad (4.9)$$

$$\frac{\partial v}{\partial t} + f u = -g \frac{\partial h}{\partial y} - \frac{1}{\rho_0} \frac{\partial P_a}{\partial y} + B, \quad (4.10)$$

where

$$A = - \left[u \frac{\partial}{\partial x} + v \frac{\partial}{\partial y} + w \frac{\partial}{\partial z} \right] u - \frac{1}{\rho_0} \frac{\partial P'}{\partial x} + \left[\frac{\partial}{\partial x} K_H \frac{\partial}{\partial x} + \frac{\partial}{\partial y} K_H \frac{\partial}{\partial y} + \frac{\partial}{\partial z} K_V \frac{\partial}{\partial z} \right] u, \quad (4.11a)$$

$$B = - \left[u \frac{\partial}{\partial x} + v \frac{\partial}{\partial y} + w \frac{\partial}{\partial z} \right] v - \frac{1}{\rho_0} \frac{\partial P'}{\partial y} + \left[\frac{\partial}{\partial x} K_H \frac{\partial}{\partial x} + \frac{\partial}{\partial y} K_H \frac{\partial}{\partial y} + \frac{\partial}{\partial z} K_V \frac{\partial}{\partial z} \right] v, \quad (4.11b)$$

with

$$P' = P - \rho_0 g h - P_a - \rho_0 g z. \quad (4.11c)$$

Here P' is the pressure due to the baroclinicity of the ocean (i.e., its density perturbation), P_a is atmospheric pressure, P being the total pressure in the water.

Averaging (4.9) and (4.10) with respect to height, and assuming that h , the surface deviation, is at least two orders of magnitude smaller than H , the total mean depth of the ocean, we have

$$\frac{\partial[u]}{\partial t} - f[v] = -g \frac{\partial h}{\partial x} - \frac{1}{\rho_0} \frac{\partial P_a}{\partial x} + [A], \quad (4.12)$$

$$\frac{\partial[v]}{\partial t} + f[u] = -g \frac{\partial h}{\partial y} - \frac{1}{\rho_0} \frac{\partial P_a}{\partial y} + [B]. \quad (4.13)$$

Subtracting (4.12) and (4.13) from (4.9) and (4.10) we have

$$\frac{\partial u'}{\partial t} - f v' = A - [A], \quad (4.14)$$

$$\frac{\partial v'}{\partial t} + f u' = B - [B]. \quad (4.15)$$

We note that (4.14) and (4.15) are independent of the terms that govern external gravity waves, namely, $-g \nabla h$ and $-(1/\rho_0) \nabla P_a$. Thus, (4.14) and (4.15), together with (4.3)-(4.6) govern the slow-moving baroclinic modes, mostly Rossby waves and internal waves. Equations (4.12), (4.13) and (4.7) govern the external gravity modes and can be solved implicitly.

4.2 Numerical Model

In this section we will present the finite difference techniques employed to solve the two systems of equations given above. We shall represent values of a dependent variable $\phi(x, y, z, t)$ as discrete values of the independent variables $x = i \Delta x$, $y = j \Delta y$, $z = k \Delta z$, and $t = n \Delta t$ as ϕ_{ijk}^n . The finite difference operators that replace first and second derivatives are the following:

$$\delta_x \phi = (\phi_{i+1/2} - \phi_{i-1/2}) / \Delta x, \quad (4.16a)$$

$$\delta_{2x} \phi = (\phi_{i+1} - \phi_{i-1}) / 2 \Delta x, \quad (4.16b)$$

$$\delta_x^2 \phi = \delta_x (\delta_x \phi) = (\phi_{i+1} + \phi_{i-1} - 2\phi_i) / \Delta x^2. \quad (4.16c)$$

$$\bar{\phi} = (\phi_{i+1/2} + \phi_{i-1/2}) / 2. \quad (4.16d)$$

The expression (4.16d) is used to define values of ϕ at spatial locations halfway between the original grids; these are needed in the differencing of the transport terms for the various quantities (see Figs. 4-1 and 4-2).

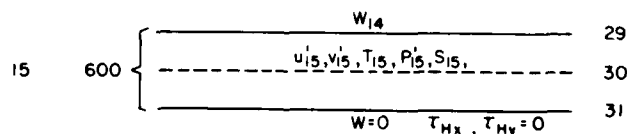


Fig. 4-1 — Vertical layering of the model

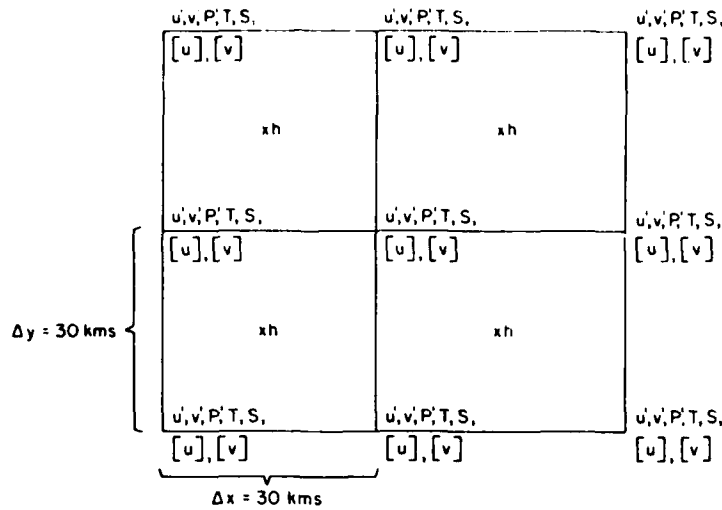


Fig. 4-2 — Grid network on a horizontal plane.

We shall first treat the system governing the barotropic motion, (4.12), (4.13), and (4.7)

$$[u]^{n+1} = [u]^{n-1} + 2\Delta t \{ (F^{n+1} + F^{n-1})/2 + [A]^n \}, \quad (4.17)$$

$$[v]^{n+1} = [v]^{n-1} + 2\Delta t \{ (G^{n+1} + G^{n-1})/2 + [B]^n \}, \quad (4.18)$$

$$h^{n+1} = h^{n-1} + 2\Delta t \{ (J^{n+1} + J^{n-1})/2 + 2\Delta t K^n \}, \quad (4.19)$$

where

$$F = f[v] - g\delta_x \bar{h}^v - (1/\rho_0)\delta_x \bar{P}_a^x, \quad (4.20a)$$

$$G = -f[u] - g\delta_y \bar{h}^x - (1/\rho_0)\delta_y \bar{P}_a^y, \quad (4.20b)$$

$$J = -H\{\delta_x [\bar{u}]^y + \delta_y [\bar{v}]^x\}, \quad (4.20c)$$

$$K = -\{\delta_x (\bar{u}^y \bar{h}^x) + \delta_y (\bar{v}^x \bar{h}^y)\}. \quad (4.20d)$$

All terms in a given equation are expanded about the spatial location on which the respective variable occurring on the left-hand side is defined. Note that u, v and u', v' are all defined at the same lattice points, so that no averaging is necessary for the Coriolis terms.

We now assume that the pressure P_a of the atmosphere at sea level is known. Because of the simple way that $[u]$ and $[v]$ enter F and G , we can then find direct expressions for $[u]^{n+1}$ and $[v]^{n+1}$.

$$[u]^{n+1} = \frac{1}{(1 + f^2 \Delta t^2)} \{ (-g\Delta t)(\delta_x \bar{h}^v + f\Delta t \delta_y \bar{h}^v)^{n+1} + \gamma_1^n + f\Delta t \gamma_2^n \}, \quad (4.21a)$$

$$[v]^{n+1} = \frac{1}{(1 + f^2 \Delta t^2)} \{ (-g\Delta t)(\delta_y \bar{h}^x - f\Delta t \delta_x \bar{h}^x)^{n+1} + \gamma_2^n - f\Delta t \gamma_1^n \}, \quad (4.21b)$$

with

$$\gamma_1^n = 2\Delta t \{ [A]^n - (1/\rho_0) \overline{\delta_x \bar{P}_a^{2t}} \} + [u]^n + f\Delta t [v]^n - g\Delta t (\delta_x \bar{h}^1)^n, \quad (4.22a)$$

$$\gamma_2^n = 2\Delta t \{ [B]^n - (1/\rho_0) \overline{\delta_y \bar{P}_a^{2t}} \} + [v]^n - f\Delta t [u]^n - g\Delta t (\delta_y \bar{h}^1)^n. \quad (4.22b)$$

Substituting expressions (4.21a,b) into (4.19), we get a second-order elliptic equation for h^{n+1} ;

$$\begin{aligned} h^{n+1} - Hg\Delta t^2 \left\{ \left[\frac{1}{1+f^2\Delta t^2} \overline{\delta_x (\delta_x \bar{h}^1 + f\Delta t \delta_y \bar{h}^1)} \right]^{n+1} \right. \\ \left. + \left[\delta_y \frac{1}{1+f^2\Delta t^2} \overline{(\delta_y \bar{h}^1 - f\Delta t \delta_x \bar{h}^1)} \right]^{n+1} \right\} \\ = E^n - H\Delta t \left\{ \frac{1}{1+f^2\Delta t^2} \overline{\delta_x (\gamma_1^n + f\Delta t \gamma_2^n)} + \delta_y \frac{1}{1+f^2\Delta t^2} \overline{(\gamma_2^n - f\Delta t \gamma_1^n)} \right\}, \end{aligned} \quad (4.23)$$

where

$$E^n = \Delta t J^{n-1} + 2\Delta t K^n + h^{n-1}, \quad (4.24)$$

and γ_1^n , γ_2^n , J , and K are defined by (4.22) and (4.22c,d), respectively.

Equation (4.23) is solved iteratively, subject to the condition that $h \rightarrow 0$ at the far boundaries. Then (4.21) can be used to find u^{n+1} and v^{n+1} , by substituting h^{n+1} from (4.23).

The finite differencing of the system of equations (4.14) and (4.15), governing the motion of the baroclinic modes, follows the procedure used for the barotropic system. We rewrite them as

$$\partial u'/\partial t = f v' + A - [A] = M + F_x, \quad (4.25)$$

$$\partial v'/\partial t = f u' + B - [B] = N + F_y, \quad (4.26)$$

where

$$M = -(\mathbf{v} \cdot \nabla) u - (1/\rho_0)(\partial P'/\partial x) - [A] + f v', \quad (4.27)$$

$$N = -(\mathbf{v} \cdot \nabla) v - (1/\rho_0)(\partial P'/\partial y) - [B] - f u', \quad (4.28)$$

with F_x , F_y being the friction terms defined in (4.11). The time differencing of (4.25) and (4.26) becomes an explicit, leap-frog technique, i.e.,

$$(u')^{n+1} = (u')^{n-1} + 2\Delta t (M^n + F_x^{n-1}), \quad (4.29)$$

$$(v')^{n+1} = (v')^{n-1} + 2\Delta t (N^n + F_y^{n-1}). \quad (4.30)$$

The solutions are filtered every 30 timesteps to eliminate grid separation. The spatial differencing of the terms M and N is given by

$$\delta_x (\bar{u}^x \bar{u}^x) + \delta_y (\bar{u}^x \bar{v}^y) + \delta_z (w \bar{u}^x), \quad (4.31a)$$

$$\delta_x (\bar{u}^x \bar{v}^y) + \delta_y (\bar{v}^y \bar{v}^y) + \delta_z (w \bar{v}^y), \quad (4.31b)$$

$$\delta_x (\bar{u}^x \bar{T}^x) + \delta_y (\bar{v}^y \bar{T}^y) + \delta_z (w \bar{T}^x), \quad (4.31c)$$

$$\delta_x(\bar{u}'\bar{s}') + \delta_y(\bar{v}'\bar{s}') + \delta_z(\bar{w}\bar{s}'). \quad (4.31d)$$

It should be noted that the finite difference form of the advective terms has the quadratic conservative property and therefore the model is free from nonlinear instability. The finite difference form of the continuity and hydrostatic equations is given by

$$\delta_x u + \delta_y v + \delta_z w = 0, \quad (4.32a)$$

and

$$\delta_z P = -g\bar{\rho}'.$$

4.3 Results for Three-Dimensional Ocean Simulation

Two numerical experiments have been performed on the same physical model, with the initial and boundary conditions, given by Madala and Piacsek (1977), utilizing an explicit and implicit model, respectively. The formulation of the explicit model is very straightforward; see, e.g., Crowley (1968).

Figures 4-3 and 4-4 illustrate the comparison of the implicit and explicit time integrations for the azimuthal velocity and temperature deviations at various depths, respectively. The results are illustrated at 60 hours of elapsed integration as a function of radial distance. The velocity results show appreciable deviation only near the boundaries, particularly in the lower layer. The errors for h in the upper layer amount to less than 10%, but amount to about 25% in the lower layer. The temperature results show a smaller error, a few percent, in all regions of the flow. The velocity discrepancies are attributed to the different arrival times and reflection of the surface waves as treated by the implicit and explicit model, respectively. Sponge layers containing a Rayleigh viscosity can be inserted to reduce reflection effects. The errors seem to have the same absolute size in each layer and can be attributed to surface effects manifested in layers 3 and 7. The amount of computer time saving achieved using the present method is a factor of 12; the time-step difference of factor 15 was slightly offset by the time necessary to iterate the one two-dimensional Helmholtz equation for the free surface height.

An alternate approach to semi-implicit and split semi-implicit techniques called split explicit integration method has been developed to integrate a multi-layer numerical model for tropical cyclones. In this method, the spectral equations governing each of the eigenmodes of the numerical model are solved explicitly using a time step not exceeding the time step given by the CFL condition for the mode. Then at regular intervals, the eigenmodes are combined to obtain the required solution. The integration of a 5 layer, three-dimensional tropical cyclone prediction model shows that the new method is about thirty percent faster than the semi-implicit method and about twenty percent faster than the split semi-implicit scheme.

5. LAGRANGIAN FLUID DYNAMICS USING TRIANGULAR GRIDS

Lagrangian methods offer the most natural approach to those transient applied hydrodynamics problems which contain free surfaces, interfaces, sharp gradients, or boundaries. In practice, their use in numerical solutions has previously been restricted to one-dimensional problems and "well-behaved" flows, since shear, fluid separation, and large-amplitude motions

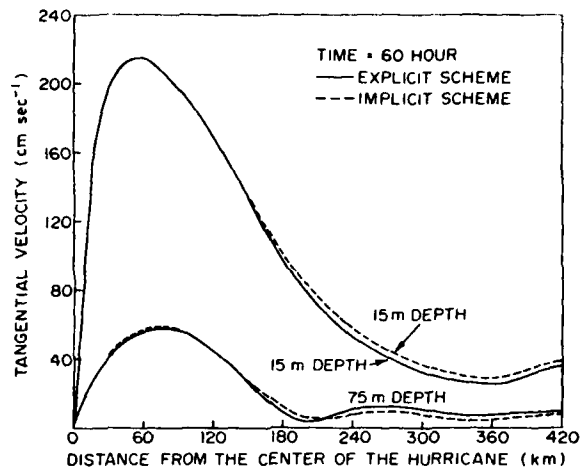


Fig. 4-3 — A comparison of the tangential velocities (15 and 75 m depths) obtained at the end of 60 hours of integration of the implicit and explicit models.

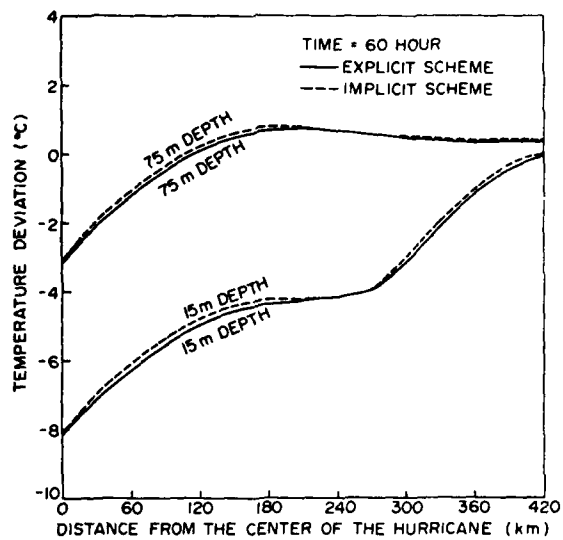


Fig. 4-4 — A comparison of the deviatory temperatures (15 and 75 m depths) obtained at the end of 60 hours of integration of the implicit and explicit models.

produce severe grid distortions which cause large inaccuracies and numerical instability. The distortions arise from the migration of mesh points which were formerly neighboring, but which have crossed or become separated in the flow. Numerical solutions of the physical equations differenced over such a mesh quickly fail because the simple finite-difference approximations used ignore mesh point rearrangement and grid distortion. This problem has been solved at NRL by implementing numerically a general-connectivity grid in which local mesh reconnections are made whenever the grid distorts appreciably.

The techniques described here involve the use of a Lagrangian, finite-difference mesh of connected triangles to represent the fluid motion, the various gradients and interfaces, and any free surfaces or boundaries present. Some of the basic concepts were developed by Crowley (1971) using the code FLAG. Our own work has concentrated on the free-surface and physical consistency aspects of the problem.

A good example of the type of problem requiring improved numerical techniques is the problem of a large-amplitude breaking surface wave. When the wave top separates from the wave due to bottom shallowing, nonlinear wave interaction, or strong surface winds, it falls back into the wave trough in a complicated flow which the usual techniques cannot handle. One goal of our work has been to develop Lagrangian triangular grid algorithms in which the crest of the numerical wave will be able to separate from the body of the wave, fall under the influence of gravity and any strong winds back into the trough of the wave, and become reabsorbed smoothly while following the numerical representation of the fluid at the surface. This extremely difficult free surface problem is conceptually identical to the breaking internal waves expected during strong subsurface disturbances, except that the density ratio is larger, $10^3:1$, and the problem correspondingly more difficult.

Figure 5-1 illustrates the use of a Lagrangian triangular mesh to facilitate the solution of this problem. As the neck of fluid narrows, the corresponding triangle sides shrink to zero and the connection between the two bodies of dense fluid is broken. In this way the topology of the problem has changed self-consistently with the evolution of the system. Clearly, any representation of the problem, to be adequate, must be capable of drastic local changes such as this to reflect complicated motions (Boris, Fritts and Hain 1975). It is equally clear that a numerical free-surface and interface capability which incorporates such reconnection algorithms for the solution of fully nonlinear flows will allow an almost inexhaustible list of important hydrodynamics problems to be solved, including the highly nonlinear flow problems described above (see Barcelona et al., 1974).

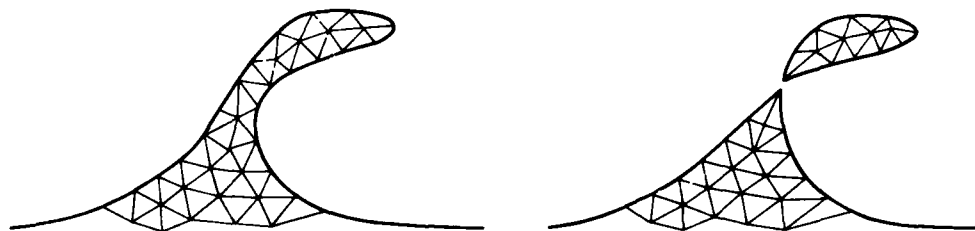


Fig. 5-1 — A schematic illustration of the use of a two-dimensional grid of triangles to represent a large-amplitude, separating, free-surface gravity wave

Previous Lagrangian treatments have been restricted to simple flows or small-amplitude motions because they use a topologically rectangular finite-difference mesh. The major practical problem with these rectangular methods arises from the inflexible connectivity of the various mesh points. In complicated and strongly sheared flows, where one element of fluid may become widely separated from an initially nearby element, the usual Lagrangian treatments break down because a simply structured rectangular mesh becomes too severely distorted to allow an adequate numerical representation of the fluid flow. When the mesh becomes so distorted that no greater Lagrangian motion can be permitted, a process of continual rezoning amounts to a form of numerical diffusion. Thus the usual Lagrangian treatments are capable of extending linear models significantly into the nonlinear regime but are not capable of providing a satisfactorily accurate representation of complicated flows bordering on turbulent phenomena.

Another drawback of such approaches is a difficulty in representing complicated boundaries and topography because of the limited topology of the mesh. It is often necessary to introduce greater resolution simply to obtain a satisfactory initial grid. Rectangular mesh approaches also appear to suffer a serious "even-odd" or computational-mode instability which must be overcome by some form of added numerical damping. This damping destroys the reversibility of the algorithm and limits its usefulness for high Reynolds-number flows even though some care has been given to developing damping algorithms which minimize this non-physical diffusion.

A triangular-element mesh has several advantages. The grid can be restructured. Individual triangles and sides can be bisected or rearranged to give new grid structures which better represent changing fluid flows. Since the number of triangles meeting at a vertex is variable, increased accuracy in one region of the flow does not force unnecessary resolution in other areas of the flow. This versatility also permits both regular and irregular tessellations of the $x-y$ plane with triangles. Triangles, unlike rectangles, can cover a circle symmetrically without cusps or other local representation irregularities. Thus free surfaces, complicated interfaces, sharp gradients, and boundaries of immersed objects can be represented accurately and economically.

The triangle is a much less ambiguous structure than a rectangle or higher-order polygon and hence interpolations and integrals are usually simpler to perform. Because the figures are three-sided (rather than four-sided), there is no unambiguous labeling of grid points as even and odd. Hence, the even-odd problems of rectangular schemes appear to be absent, or at least greatly subdued, in the triangular representation. A vertex which is one point removed from its neighbor along a particular path will be two points removed by another. This does not mean that an even-odd problem cannot occur, only that it is not a topologically natural mode of instability and hence is considerably slower growing than its rectangular-mesh counterpart. Experience supports this; an essentially reversible algorithm is feasible by use of the Lagrangian triangular mesh approach.

Of course there are also problems with the triangles. A general connectivity triangular mesh has nontrivial bookkeeping problems associated with the grid and its connections. Furthermore, numerical experience with triangular meshes is much more limited than experience with rectangular meshes. The statement of the hydrodynamic equations in triangular systems requires that especially close attention be paid to the spatial derivative terms which are needed. There are, as well, some intrinsic numerical complications in triangular systems. The major one, discussed below, is the counting of equations and free unknowns. This difficulty can be

argued to be the price we pay for the relative suppression of the "even-odd" or "computational" modes observed in more standard rectangular approaches.

We have restricted ourselves to the study of fluids which are inviscid and incompressible but have variable density. The conservation integral approach and definitions of divergence we employ, however, allow a natural extension to compressible systems. We also concentrate attention on problems in which the gravity is constant and directed in the negative y direction. These are not necessary restrictions but simplify the analyses and allow the full spectrum of problems of current interest to be solved. The basic equations of the system are:

$$\rho \frac{d\mathbf{v}}{dt} = -\nabla p - \rho g \hat{\mathbf{y}} \quad (5.1)$$

and

$$\nabla \cdot \mathbf{v} = 0 = \frac{dp}{dt} \quad (5.2)$$

The fluid density ρ , pressure p , and velocity $\mathbf{v}$ are assumed to vary only with x and y . Equation (5.2), the condition for incompressibility, removes the sound waves. We will assume that p is a constant along free surfaces.

Figure 5-2 shows a section of a triangular mesh representation with an interface of a fluid of type I connected to a fluid of type II. The basic elements involved in the construction of a triangular mesh are shown. In Fig. 5-2a a particular triangle j is shown in heavy lines and the various elements of that triangle are labeled. Three vertices V_1 , V_2 , and V_3 are connected consecutively by sides S_1 , S_2 , and S_3 . Clearly, triangle j shares these vertices and sides with other triangles. The direction of labeling around each triangle is taken to be counterclockwise and the z direction is out of the page. Since the mesh can be irregularly connected, an arbitrary number of triangles can meet at each vertex. For example, five triangles and sides meet at vertex V_1 . The number of triangles and sides meeting at a vertex is equal except near free surfaces and boundary surfaces where there may be no grid above the surface. Each side is bounded by two triangles (in general) and each triangle shares sides with three other triangles.

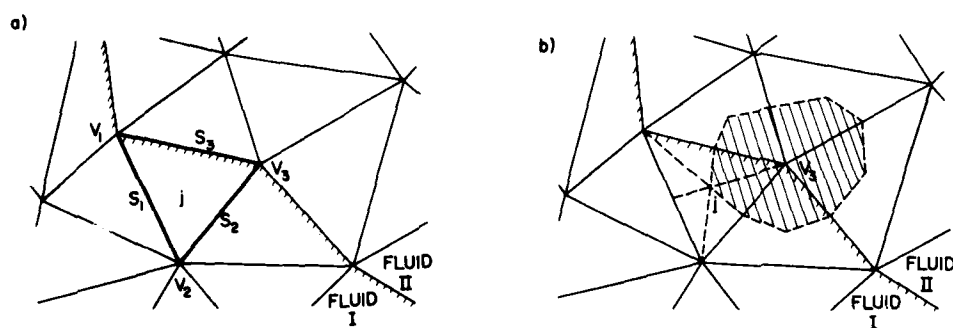


Fig. 5-2 — (a) sides and vertices of a triangle j located at the interface between fluid I and fluid II; (b) the same, with the bisectors of the angles drawn to define cell associated with vertex V_3 .

Figure 5-2b illustrates several important features of triangles which are used in constructing these algorithms. It is convenient to define a cell surrounding a vertex as shown by the shaded region surrounding vertex 3. The borders of such vertex-centered cells are determined by constructing all of the side bisectors for each triangle. Since the three side bisectors all intersect at a point, as shown for triangle j , there is no ambiguity in constructing the vertex cells as indicated. The point of intersection of the side bisectors, the centroid of the triangle, is the center of gravity for the figure; this is true in $r-z$ as well as Cartesian ($x-y$) coordinates. The three side bisectors of any triangle divide the triangle into six subtriangles. These six subtriangles all have the same area and, therefore, each of the three vertex cells receives one-third of the area of the triangle. When a quantity is constant over a triangle, the contributions of that quantity from a triangle to each of the three vertex cells are also equal. These properties of side bisectors make the calculation of cell volumes and cell masses particularly simple.

Any computational representation by a triangular mesh must record all important aspects of the mesh interconnection. If each vertex, each side, and each triangle are numbered, lists of interconnections can simply take the form of an ordered series of integers, which can be stored compactly in the computer. For example, the information might be stored in the form of lists:

1. Each vertex list enumerates (a) the vertices to which the given vertex is connected, (b) the sides to which it is connected, and (c) the triangles which meet at that vertex.
2. For each side, lists could record (a) the starting and the finishing vertex of that side and (b) the triangle to the right and the left of that side, where the side is regarded as a directed vector from the starting vertex to the finishing vertex.
3. Each triangle list could contain (a) the three vertices of the triangle ordered in a counterclockwise direction and (b) the three sides of the triangle between the corresponding vertices, again ordered in a counterclockwise direction.

Since these lists are not all independent, they are not all needed. Knowing only the vertices to which a given vertex is connected, for example, permits all of the other vertex information to be determined. If vertex 2 is connected to vertex 7, the side lists can be searched to determine which side connects vertex 7 and vertex 2, and thus the various sides connected to a given vertex can be determined. However, while maintaining some redundant lists is costly in terms of storage, there are real advantages gained in increased computation speed, clarity, and ease of coding.

It should be clear that the arrays of quantities used to define the grid and its motion involve the storage only of local information. There is no global representation of the mesh in general, although for some specific geometries, such a global representation may be possible. Connection paths between vertices can be determined only by searching sequentially through neighboring vertices. The lack of a global representation in which the vertex numbering denotes a corresponding relative spatial position greatly complicates implicit calculations. Poisson's equation, for example, is solved by iteration, and explicit time and space derivatives are the best that one has any right to hope for. Nevertheless, using the incompressible formulation ensures that timestep limitations due to acoustic transit times are not a problem.

The SPLISH code which embodies these Lagrangian algorithms is currently configured to treat strips of different but constant density fluid separated by sharp interfaces. The bottom is a

rigid boundary, and the top is a free surface. The sides are periodic to facilitate the study of waves and wave interactions. In many problems it may be desired to treat the thermocline as a staircase with a small number of steps connecting the lower-density mixed layer above to the slightly higher density below. In other problems where an initially continuous density profile is preferred, this can be accomplished by defining a density at each vertex and keeping it constant in time as the vertex moves. When free surface gravity waves are not of interest, a lid boundary condition can be placed on the top of the mesh which not only simplifies the boundary treatment but also allows much longer timesteps because the fast-running surface waves are suppressed.

This numerical model is appropriate for simulating internal-surface wave coupling and a host of nonlinear interactions both on and below the surface. An even more difficult test of a Lagrangian code has been carried out with the sign of the density gradient reversed. The Rayleigh-Taylor calculation for two fluids of density ratio 2:1 is patterned on the situation found in a simple laboratory demonstration and displays all of the linear and nonlinear features observed experimentally (Fig. 5-3).

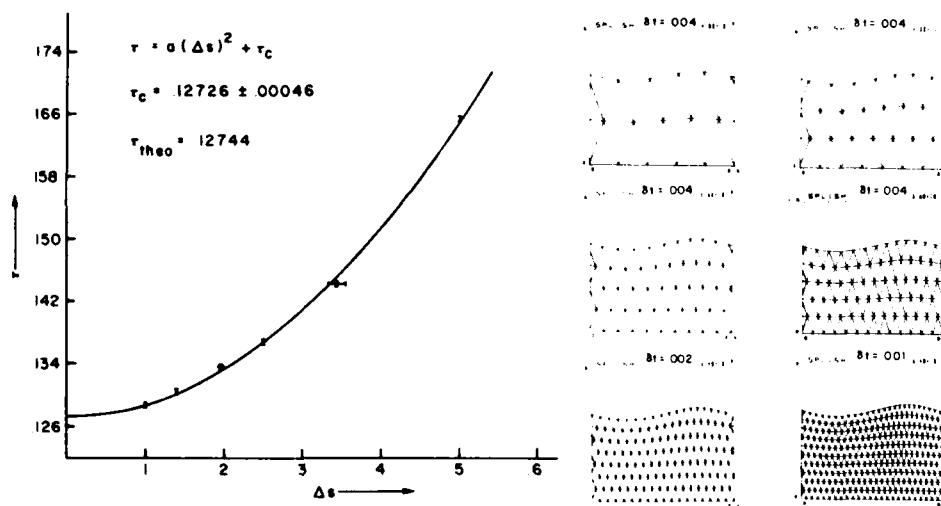


Fig. 5-3 — Wave period versus normalized numerical grid size for four different resolution calculations of a finite-depth free surface wave

An initial grid of triangles was established with a small sinusoidal perturbation of the interface at $t = 0$. The initial velocities were all taken to be zero. The flow field is followed as the instability enters its nonlinear phase. The vertex locations are plotted at successive timesteps to give streaks whose lengths is proportional to the local fluid velocity. Shear flow is established as the interface steepens, giving rise to vortices centered on the interface in a flow which has been likened to the onset of the Kelvin-Helmholtz instability (Chandrasekhar, 1961). The two fluids are observed to mix in the Rayleigh-Taylor vortices, and the vortices are seen to have entrained enough lighter fluid to form "bubbles" which are about to be completely enveloped by the heavier fluid.

Near the end of the calculation most of the grid carried by the lighter fluid has flowed out from below the downward jetting heavier fluid. Similarly, vertices in the heavier fluid have been deleted above the upswelling lighter fluid. A reasonable resolution has been maintained by adding other vertices, particularly along the interface. Originally the interface was resolved with 10 vertices, but 46 were required by the end of the run. New regridding routines are still needed to permit the attachment of fluid through an intervening thinned layer. The heavier fluid has not touched the bottom but rests upon extremely thinned and skewed triangles in the lighter fluid. The presence of these near-zero-area triangles has terminated the calculation due to violation of the Courant condition. Similar problems will soon arise at the free surface near the upswelling lighter fluid and at the cavitation "bubbles" which are about to be formed in both vortices.

This approach may not be the only method of dealing with severe grid distortions (see, e.g., Chan, 1974), but it is the most general, most physical, and the least dissipative technique employed so far. Fritts (1976, 1976a) and Fritts and Boris (1977) have recently employed Lagrangian triangular grid techniques in studying nonlinear aspects of free-surface waves including strongly sheared flows. Ten to fifteen full cycles of waves can be integrated reversibly without significant deterioration of the solution, and the reconnection procedure in the presence of strong shear is reversible (i.e., nondissipative). Figure 5-4 shows a plot of the wave natural period versus grid size for four different resolution calculations of the same physical problem. The second-order accuracy over long times is demonstrated by the parabolic form of the curve. Going to smaller stepsize gives improved accuracy, as expected. An added advantage accrues from using triangular cells. The nonlinear mesh-separation instabilities which plague low-dissipation rectangular cell techniques seem to be absent or damped with triangular cells.

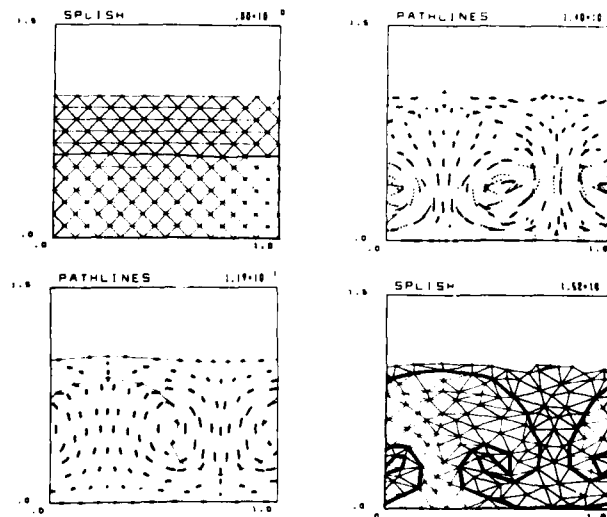


Fig. 5-4 — Development of the Rayleigh-Taylor instability. Note the deformation of the interface occurring between (a) and (d), which show locations of the triangles. In (b) and (c) flow lines are marked to show the onset of shear flow and formation of Kelvin-Helmholtz-like vortices.

The remaining drawbacks to the triangular grid approach are the complexity of the programming required and the fact that the operations which are required seem to hinge strongly on linked lists, random access, and sequential processing. These problems become more pronounced when one attempts to construct a 3D hydro code (based on the use of tetrahedrons instead of triangles), but seem to be amenable to solution by generalizations of the techniques described here. With several pipeline and vector processors working and several more soon to be in operation, efforts toward "vectorization" for parallel processing of these triangular-grid techniques have already met with appreciable success.

6. SOLUTION OF ELLIPTIC EQUATIONS

6.1 Survey of Standard Techniques

Solution of the Poisson equation (2.5) or (2.8) is necessary whenever we employ the vorticity-stream function representation of the incompressible fluid equations. A Poisson equation is also encountered when we seek the solution for the pressure in the primitive equation formulation. Generalizations of the Poisson equation arise in other physical problems with similar features, e.g., motion of plasma in the ionosphere (Scannapieco et al., 1976). All of these are elliptic equations. In most calculations the boundary conditions take the form of Neumann, Dirichlet, or "radiation" conditions; in a few cases of interest, absorbing boundaries which neither transmit nor reflect waves are employed.

The techniques used for solving elliptic equations may be divided into "direct" and "iterative," or into "transform" and "finite difference" types. Since transforms may be regarded merely as maximal-order finite difference procedures, the former distinction is more fundamental for computational purposes. The present section is devoted to a discussion of the standard techniques from this point of view. Following Hockney (1970), we restrict our attention to the simple Poisson equation. Some of the methods discussed generalize straightforwardly to other types of elliptic equations.

Vector computers place a new set of constraints on the choice of numerical algorithms for computational problems. Factors such as the length of data vectors and the time which intervenes between obtaining a result and the subsequent use of that result as an operand in another computation must be considered in optimizing numerical algorithms. Sparse banded matrix equations, like those which arise from the finite-difference solution of elliptic equations on a two- or three-dimensional grid, are examples of the type of problem which can be treated by vectorizable matrix solvers.

A vector computer has two essential features. It achieves high-speed performance through the process-overlapping technique known as "pipelining," and it provides "vector" instructions at *machine* level to maximize the effectiveness of the pipeline concept in processing large arrays.

The pipeline concept is applied at two distinct levels:

1. Overlapping of instructions. (On the TI/ASC, a maximum of 12 instructions may be executed simultaneously within the instruction pipeline.)

2. Overlapping of array indexing. (This feature is especially attractive for large problems of scientific interest which lend themselves to representation by large matrices/arrays.)

Of course vector computers can also operate in scalar mode at reduced efficiency.

Direct methods

The simplest direct method conceptually is solution via a double fast Fourier transform. (Boris and Roberts, 1969). In k space we have for a 2D periodic system

$$\psi_k = -k^{-2} \zeta_k. \quad (6.1)$$

In a finite difference problem the proper representation of this is

$$\psi_k = - (2/\delta x^2 (1 - \cos k_x \delta x) + 2/\delta y^2 (1 - \cos k_y \delta y))^{-1} \zeta_k \quad (6.1)'$$

Inversion of the transform then completes the solution for ψ . The method generalizes readily to 3D [where (6.1) is replaced by three equations for the components of A_k] and to simple nonperiodic boundaries, through the addition of an appropriately chosen solution of Laplace's equation. The number of operations required on an $N_1 \times N_2$ mesh is $\sim 5-10$ times $N_1 N_2 \cdot [\log_2(N_1 N_2) - 3]$, the constant factor depending on the boundary conditions.

The Double Cyclic Reduction (DCR) method starts with the simple five-point 2D finite-difference scheme, which may be written for a uniform mesh in the form

$$\Psi_{j+1} - C^{(1)} \cdot \Psi_j + \Psi_{j-1} = -S_j^{(1)}, \quad (6.2)$$

where Ψ_j and S_j are column vectors and $C^{(1)}$ is the tridiagonal matrix with coefficients $\{-1, 4, -1\}$. The first stage of odd/even reduction converts (6.2) into

$$\Psi_{j+2} - C^{(2)} \cdot \Psi_j + \Psi_{j-2} = -S_j^{(2)}, \quad (6.3)$$

where

$$C^{(2)} = C^{(1)} \cdot C^{(1)} - 2I \quad (6.4)$$

and

$$S_j^{(2)} = S_{j+1}^{(1)} + C^{(1)} \cdot S_j^{(1)} + S_{j-1}^{(1)}. \quad (6.5)$$

The reduction process is repeated until the L th level is reached ($N = 2^L + 1$), when only a single equation remains, which for Dirichlet boundary conditions relates the central vector to two known vectors of boundary values:

$$\Psi_{M+1} = [C^{(L)}]^{-1} [\Psi_N + S_{M+1}^{(L)} + \Psi_1], \quad (6.6)$$

where $M = (N-1)/2 = 2^{L-1}$. We now iterate (6.6) to find the column vector halfway between 1 and $M+1$ and that halfway between $M+1$ and N , then the column halfway between consecutive pairs of those columns, and so on. The expansion procedure at level l ($1 \leq l < L$) can be stated for $j-1$ a multiple of 2^{l-1} as

$$\Psi_j = [C^{(l)}]^{-1} [\Psi_{j+m} + S_j^{(l)} + \Psi_{j-m}], \quad (6.7)$$

with $m=2^{l-1}$ and all quantities on the right previously determined.

The matrices $C^{(l)}$ have $2l+1$ nonzero diagonals. They can be factored, however, as follows:

$$C^{(2)} = [C^{(1)}]^2 - 2I = [C^{(1)} + \sqrt{2} I] [C^{(1)} - \sqrt{2} I]; \quad (6.8)$$

$$\begin{aligned} C^{(3)} &= [C^{(2)} + \sqrt{2} I] [C^{(2)} - I] \\ &= [C^{(1)} + (2 + \sqrt{2})^{1/2} I] [C^{(1)} + (2 - \sqrt{2})^{1/2} I] \\ &\quad [C^{(1)} - (2 - \sqrt{2})^{1/2} I] [C^{(1)} - (2 + \sqrt{2})^{1/2} I], \end{aligned} \quad (6.9)$$

and so on. It turns out that this factorization is impractical on 32-bit word machines for meshes larger than 128×128 , because of overflow problems (the central coefficient of C^l is $\sim 4^l$), and other means of evaluating are preferable (Buneman, 1969).

All of the divisions by $C^{(l)}$ in the series of equations (6.7) represent inversion of a tridiagonal matrix. Buneman's (1969) method involves tridiagonal inversion during both reduction and expansion and is therefore called double cyclic reduction (DCR). Tridiagonal inversion can be performed by a variety of techniques, including odd/even reduction. It turns out that this is a particularly good tridiagonal inversion technique for use on the TI/ASC because it vectorizes readily. The total operation count for Buneman's (1969) symmetric form of DCR is approximately $6N_1N_2(\log_2 N_1 + 2)$, a modest improvement over double FFT. Against this is weighed a slightly more onerous programming task.

A simple form of the Fourier Analysis-Cyclic Reduction (FACR) method begins with FFT in one direction, followed by odd/even (cyclic) reduction along each line in the transverse direction. The procedure for the reduction and expansion exactly parallels that just described. Fourier synthesis is then performed on the result to obtain ψ .

Hockney's (1965) version of FACR proceeds as follows. One stage of odd-even reduction is performed as before. Then a real FFT is performed on the even lines which result from the first step. The resulting equations form a tridiagonal system for each harmonic k . These are then solved by recursive application of cyclic reduction. (Gauss elimination is just as good on scalar computers, but not on pipeline or vector machines.) The Fourier synthesis is carried out, and finally the solutions on the odd lines are found from the original five-point formula by tridiagonal inversion.

The total number of operations is roughly $2.5N_1N_2[\log_2(N_1) + 2.5]$, again depending on the boundaries, details of the FFT, etc. The method can be generalized, however, to include l levels of odd/even reduction before Fourier analysis is performed (Hockney, 1970). When this is done the total number of operations is $N_1N_2[3 + 4.5 + 2^{-l}(5\log_2 N_1 - 4)]$. This has a shallow minimum at $l \geq 2$ (depending on N_1, N_2) and shows that slight improvement over the preceding operation count is possible. Thus for large arrays FACR is the fastest of these methods. All three require only 1D scratch storage, as they overwrite the source array with the potential solution.

Another method for inversion of sparse banded (not necessarily pentadiagonal) $N \times N$ matrices is Crout (1941) reduction. Given an equation of the form

$$Mz = x, \quad (6.10)$$

write M as a product of upper and lower triangular matrices:

$$M = L \cdot U. \quad (6.11)$$

Then solve

$$Ly = x, \quad (6.12)$$

followed by

$$Uz = y. \quad (6.13)$$

In the coded algorithm, the vectorized decomposition operates on the rectangle whose vertices are on the main diagonal and the outermost bands. This rectangle moves down the main diagonal one step at a time. Thus the kernel of the decomposition is a triply nested DO-loop on a vector dot product. This is the fastest vector instruction on the ASC. The solutions are simply row-by-row dot products.

During execution the dot product in the decomposition dominates the calculation. It is approximately $4N^3$ operations (add plus multiply times the length of the main diagonal times the bandwidth). For $N \sim 40$, the computing time is about 10% more than $2N^3$ times the theoretical vector speed. The decomposed matrix occupies $2N^3$ storage locations and dominates the memory requirement.

Iterative methods

Iterative methods are easy to code and (depending on the problem) can accelerate convergence by careful choice of the starting guess. The best known of these is successive overrelaxation (SOR) by points. If we write the Poisson equation in the form

$$R_{i,j} \equiv \phi_{i-1,j} + \phi_{i+1,j} + \phi_{i,j-1} + \phi_{i,j+1} - 4\phi_{i,j} - q_{i,j} = 0, \quad (6.14)$$

the iterative scheme is defined by

$$\phi_{i,j}^{(\text{new})} = \phi_{i,j}^{(\text{old})} + (\omega/4) R_{i,j}, \quad (6.15)$$

where ω is an adjustable number. For this scheme, the rate of convergence is highly sensitive to the order in which the corrections are carried out. If they are done a line at a time, the procedure can be vectorized. Typically 5-10 iterations can be performed during the time required for a direct solution.

The worst-case rate of convergence (based on the poorest possible initial guess) can be improved by varying ω suitably. One technique, the Cyclic Chebyshev method, uses the following prescription to vary ω every half iteration:

$$\omega^n = 1; \quad (6.16)$$

$$\omega^{1/2} = (1 - \mu^2/2)^{-1/2}; \quad (6.17)$$

$$\omega^{n+1/2} = (1 - \mu^2 \omega^n/4)^{-1/2}. \quad (6.18)$$

However, worst-case analyses have little to do with physically well-motivated calculations.

Quite different is the Incomplete Cholesky-Conjugate Gradient (ICCG) method (Kershaw, 1978). For a general sparse matrix operator M , we solve (6.10) for z as follows: First approximate M according to (6.11), where we set $L_{i,j} = 0$ at every location (i,j) for which $M_{i,j} = 0$. Then define $K = L \cdot U$ and find z from

$$K \cdot z = x. \quad (6.19)$$

Set

$$r = z; \quad (6.15)$$

$$p = K \cdot r; \quad (6.16)$$

Now iterate:

$$\gamma = r \cdot K \cdot r; \quad (6.17)$$

$$\alpha = \gamma(p \cdot M \cdot p)^{-1}; \quad (6.18)$$

$$z' = z + \alpha p; \quad (6.19)$$

$$r' = r - \alpha M \cdot p; \quad (6.20)$$

$$\beta = (r' \cdot K \cdot r')^{-1}; \quad (6.21)$$

$$p' = K \cdot r' + \beta p, \quad (6.22)$$

until $\|M \cdot z - x\|$ is sufficiently small. In the algorithm, the iteration is all vector operations, except for two "vector products" $K \cdot (\)$, which are actually solutions of an upper triangular system of equations and a lower triangular system of equations. These operations are totally scalar, since L is sparse by construction.

During execution, the computing time for the vector operations is small compared with the two upper-triangular and two lower-triangular solutions. Thus the required time is approximately $8N^2N_dN_i$ times the scalar operation speed, where N_d is the number of diagonals (above or below) and N_i is the number of iterations. In addition to the diagonals, the right-hand side and the solution vector, the algorithm requires about 9 more vectors. The data storage is thus of order $(N_d + 9)N^2$.

A number of iterative techniques for solving elliptic equations are related to those used to advance diffusion equations in time. Thus if an initial guess for ψ is advanced in time by means of a finite difference approximation to

$$\frac{\partial \psi}{\partial t} = \nabla^2 \psi - \zeta \quad (6.23)$$

with the appropriate boundary conditions until a stationary state is reached, the resulting function is the solution of (2.8). Of the various methods for solving parabolic equations, the one most often used is Alternating Direction—Implicit (ADI). For (6.23) it can be represented as

$$\begin{aligned} \psi_{i,j}^{n+1/2} = \psi_{i,j}^n + \frac{\Delta t}{2} \left[(\psi_{i+1,j}^{n+1/2} - 2\psi_{i,j}^{n+1/2} + \psi_{i-1,j}^{n+1/2})/\Delta x^2 \right. \\ \left. + (\psi_{i,j+1}^n - 2\psi_{i,j}^n + \psi_{i,j-1}^n)/\Delta y^2 \right] - \zeta_{i,j}^n; \end{aligned} \quad (6.24)$$

$$\begin{aligned} \psi_{i,j}^{n+1} = \psi_{i,j}^{n+1/2} + \frac{\Delta t}{2} \left[(\psi_{i+1,j}^n - 2\psi_{i,j}^n + \psi_{i-1,j}^n)/\Delta x^2 \right. \\ \left. + (\psi_{i,j+1}^{n+1} - 2\psi_{i,j}^{n+1} + \psi_{i,j-1}^{n+1})/\Delta y^2 - \zeta_{i,j}^n \right]. \end{aligned} \quad (6.25)$$

On the alternate timesteps, the implicit differencing is performed in the y direction instead of the x -direction, hence the name.

The optimum choice of Δt gives slightly faster convergence than SOR, although with more operations. By varying the interval Δt from one step to the next, convergence can be improved. No simple rules are available for this. The number of iterations required for convergence of ADI varies only weakly with the size of the mesh.

Other iterative techniques are discussed by Roache (1972). Their usefulness, as well as that of many of the variants of SOR, ICCG, and ADI, depends in large measure on the details of the problem being solved.

6.2 A New Direct Solver: The Stabilized Error Vector Propagation Technique

Elliptic partial differential equations may be separable or nonseparable and their solutions are subject to a variety of boundary conditions. If we write the elliptic finite difference form of the differential equation as $Ax = F$, where A is the coefficient matrix, F is the forcing function, and x is the required solution, then A can be diagonally dominant, weakly diagonally dominant, or even nondiagonally dominant. Even though a number of iterative and direct solvers are available in the literature for solving elliptic equations, no one method is efficient for all types of equations. The iterative methods such as SOR, ADI, and hybrid methods such as optimized block implicit relaxation (Dietrich, et al., 1975) are very efficient for diagonally dominant equations. However, when diagonal dominance is weak and residual error much in excess of roundoff cannot be tolerated, convergence rates for these methods may be unattractively small. In addition, many iterative methods will diverge when applied to nondiagonally dominant equations.

Direct solvers based on FACR, DCR, and the method of Rosmond and Faulkner (1976) are very powerful but are restricted to separable equations. For nonseparable equations the direct solvers of Lindzen-Kuo (1969) and Crout (1941; see Section 6.1) are available but require large amounts of computer memory and thus are limited to small arrays. The error vector propagation (EVP) method (Roache, 1971, 1977; Hirota et al., 1970) is applicable to any type of elliptic equation and requires an order of magnitude less memory than Lindzen-Kuo. However, as shown by McAvaney and Leslie (1972), EVP is unstable when the number of grid points is large. Here we describe a method due to Madala (1978), called stabilized error vector propagation (SEVP), which is stable for any number of grid points and retains most of the advantages of EVP.

A general two-dimensional elliptic equation in finite difference form can be written as

$$AX_{I,J}X_{I-1,J} + AY_{I,J}X_{I,J-1} + BB_{I,J}X_{I,J} + CX_{I,J}X_{I+1,J} + CY_{I,J}X_{I,J+1} = F_{I,J}, \quad (6.26)$$

where the coefficients AX , AY , BB , CX and CY may be functions of both I and J , and F is the forcing function. We now review the EVP procedure described by Roache (1971, 1977) for solving (6.26) over a rectangular region (shown in Fig. 6-1) using Dirichlet boundary conditions. By rearranging the terms, (6.26) can be written as

$$X_{I,J+1} = (F_{I,J} - AX_{I,J}X_{I-1,J} - AY_{I,J}X_{I,J-1} - BB_{I,J}X_{I,J} - CX_{I,J}X_{I+1,J})/CY_{I,J}. \quad (6.27)$$

It is clear from (6.27) that if we know the solution on any two consecutive rows, then we can march in J to obtain the required solution. Since the solution is known only on the boundaries B_1 , B_2 , B_3 , and B_4 , the EVP method requires an initial guess of the solution on the row

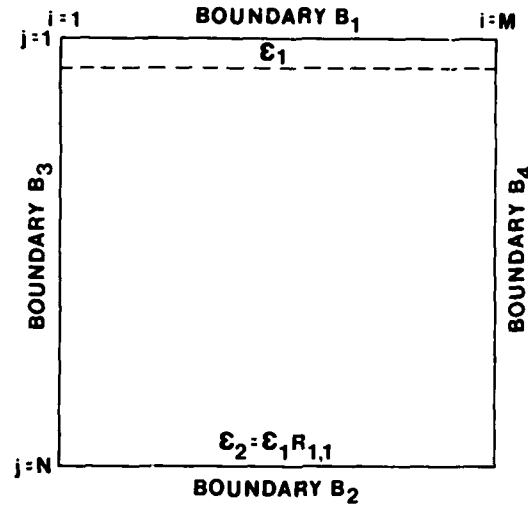


Fig. 6-1 — The region over which the elliptic equation is solved by FVP method. B_1 , B_2 , B_3 , and B_4 are the boundaries.

interior to B_1 to march forward in J . Based on this initial guess a particular solution is obtained which satisfies (6.27) and all the boundary conditions except along B_2 . If we assume that the particular solution deviates from the real solution $X_{l,j}$ by $X''_{l,j}$, then

$$X_{(l,j)} = X^p_{(l,j)} + X''_{(l,j)}. \quad (6.28)$$

Substituting (6.28) into (6.27) and noting that the particular solution satisfies (6.27) and the boundary conditions at B_1 , B_3 , and B_4 , we obtain the following homogeneous equation:

$$X''_{l,j+1} = -(AX_{l,j}X''_{l-1,j} + AY_{l,j}X''_{l,j-1} + BB_{l,j}X''_{l,j} + CX_{l,j}X''_{l+1,j}) / CY_{l,j}. \quad (6.29)$$

with zero boundary conditions along B_1 , B_3 , B_4 . Along B_2 we find from (6.30)

$$X''_{l,N} = X_{l,N} - X^p_{l,N}. \quad (6.30)$$

We can relate the homogeneous solution on any two rows by using $M-2$ independent vectors, where M represents the total number of grid points in I including boundary points. Since the solution along B_2 [given by (6.30)] is known, we will relate it to the solution on the row just interior to the boundary B_1 . In other words, if the vectors ϵ_1 and ϵ_2 (Fig. 6-1) represent the homogeneous solution on the second and last rows respectively, then we can find an influence matrix $R_{1,1}$ such that

$$\epsilon_2 = \epsilon_1 R_{1,1}. \quad (6.31)$$

To obtain $R_{1,1}$, we start with the K th unit vector along the second row and march (6.29) in J , with zero boundary conditions along B_1 , B_3 . The K th row elements of the $R_{1,1}$ matrix are the elements of (6.29) along the boundary B_2 . By varying the values of K from 2 to $M-1$ we complete $R_{1,1}$.

From (6.31) and the error vector [given by (6.30)] we compute the homogeneous solution on the second row. Using these values on the second row and zero boundary conditions on B_1 , B_3 and B_4 we march (6.29) in J to obtain the homogeneous solution throughout the region. Then by superposition of the particular and homogeneous solutions we obtain the complete solution. Due to round off errors in computing the inverse of the matrix $R_{1,1}$, this solution does not satisfy the boundary condition along B_2 exactly. The accuracy of the solution is improved by recomputing the homogeneous part of the solution a few times. (Thus this method, though direct, usually involves iteration.) Normally, about six iterations are required for 20 grid points in the marching direction to obtain an error of 10^{-14} on a computer with 56 bit word precision. The number of iterations required increases as the number of grid points in the marching direction is increased up to about 25; beyond 25 the method becomes unstable.

The influence matrix, $R_{1,1}$, depends only on the coefficients of the elliptic equation and therefore can be computed without knowledge of the forcing function and the boundary conditions. This step is called the preprocessor. If we want to solve (16.26) repeatedly with the same coefficients but with different forcing functions and boundary conditions, the influence matrix needs to be computed only once and stored in the memory for further use.

Although the EVP method is very efficient and requires a very small amount of computer memory, it is unstable for large number of grid points in the marching direction (McAvaney and Leslie, 1972). For computers with 24 bit precision, the limit is about 12 grid points, which may be increased by going to higher precision or by using the two-directional marching method described by Hirota et al. (1970). Even with 56-bit precision and two-way marching the limit is extended only to about 40 grid points.

In the SEVP method, the integration region is divided into blocks, each of which is stable for the EVP method. Further, any two consecutive blocks have two rows in common. The division of the integration region into three blocks is shown in Fig. 6-2. As with the EVP method, SEVP is divided into two steps: the preprocessor step and the solution step. In the preprocessor step, $2 \times \text{NBLK} - 1$ influence matrices are computed, where NBLK is the total number of blocks. NBLK of the influence matrices relate the values of the homogeneous solution on the second and last rows of each block, while the remaining $\text{NBLK} - 1$ matrices relate the homogeneous solution on the second rows of consecutive blocks.

The first $M-2$ unit vectors on the second row of the first block are used to march in the J direction, using (6.29) and zero boundary conditions on B_1 , B_3 , and B_4 to obtain influence matrices for the last two rows of the block. If ϵ_1 , ϵ_2 , and ϵ_3 (shown in Fig. 6-2) represent the homogeneous solution on the second and last two rows of the first block, then

$$\epsilon_2 = \epsilon_1 R_{1,1} \quad (6.32)$$

and

$$\epsilon_3 = \epsilon_1 R_{1,2}, \quad (6.33)$$

where $R_{1,1}$ and $R_{1,2}$ are the influence matrices for the last two rows of the block.

Combining (6.32) and (6.33) to eliminate ϵ_1 we obtain

$$\epsilon_3 = \epsilon_2 R_{1,1}^{-1} R_{1,2} = \epsilon_2 S_1. \quad (6.34)$$

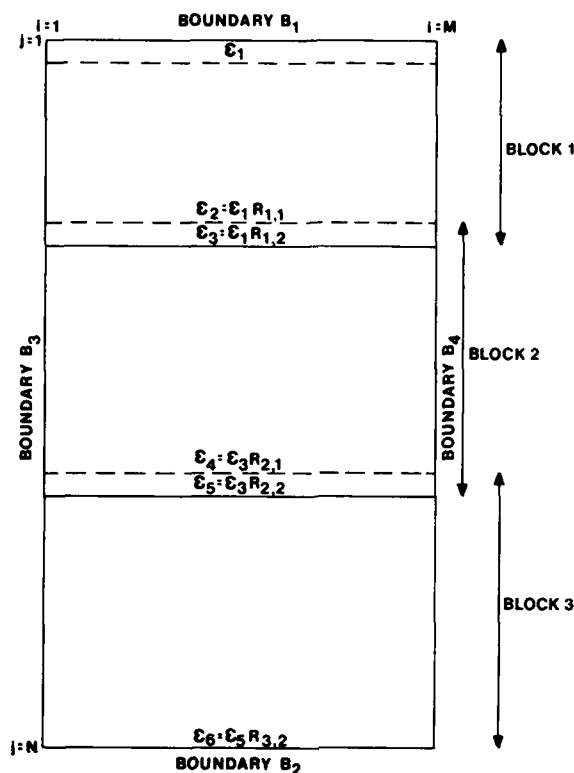


Fig. 6-2 — A three block division of the region for the SEVP method.

The matrices $R_{1,1}$ and S_1 relate homogeneous solutions on the second and last rows of the block and on the last two rows of the block, respectively.

To obtain the influence matrices for the second block, we start with the unit vectors on the second row of the second block, and compute the corresponding values on the first row of the block from (6.34) to get the equivalent of a boundary condition for this block. The homogeneous solution on the first row corresponding to the K th unit vector on the second is the K th row of the matrix S_1 . Using $M-2$ unit vectors on the second row with corresponding vectors on the first row and zero boundary values on B_3 and B_4 , we advance (6.29) in J to obtain influence matrices for the last two rows of the second block. If the vectors ϵ_3 , ϵ_4 , and ϵ_5 represent a homogeneous solution on the second and last two rows of the block, respectively, then

$$\epsilon_4 = \epsilon_3 R_{2,1} \quad (6.35)$$

and

$$\epsilon_5 = \epsilon_3 R_{2,2} \quad (6.36)$$

where $R_{2,1}$ and $R_{2,2}$ are the influence matrices for the last two rows of the block. Eliminating ϵ_3 from (6.35) and (6.36) we obtain

$$\epsilon_4 = \epsilon_5 S_2 S_2 = R_{2,2}^{-1} R_{2,1}. \quad (6.37)$$

Repeating the procedure described above for the third block (last block in Figure 6-2) we will obtain a matrix, $R_{3,2}$, relating the homogeneous solution on the second and last rows of the block. If ϵ_5 and ϵ_6 represent the homogeneous solution on these rows, then

$$\epsilon_6 = \epsilon_5 R_{3,2}. \quad (6.38)$$

As in the EVP method, all the influence matrices depend only on the coefficients of (16.26).

We obtain the required solution by one forward and one backward sweep of the blocks. During the forward sweep, an approximate solution is obtained which satisfies the equation and the boundary conditions everywhere except on boundary B_2 . In the backward sweep, this solution is corrected to obtain the exact solution.

The forward sweep is started with the second row on the first block with an initial guess for the elements of that row and the given boundary values along the first row. This procedure is exactly the same as that for EVP. When the solution has been marched to the end of the first block, arbitrary values are assigned along the block boundary to obtain the correction vector ϵ_3 . The second sweep forward generates a homogeneous solution for the first block. Iterations are applied, if necessary, to reduce roundoff error.

For the second block we take the arbitrary solution assigned along the first block boundary (second row, second block) and the solution along the first row of the second block and sweep forward again. An arbitrary solution is again imposed along the last row of the second block to obtain the correction vector ϵ_5 . On the second (or homogeneous) sweep of the second block we use the correction vectors ϵ_2 and ϵ_3 to form the homogeneous solution. We compute ϵ_3 from ϵ_5 by (6.36) and obtain ϵ_2 from ϵ_3 using (6.34). The procedure continues until all blocks are swept out. For the last block, we use the boundary B_2 instead of a guess value to find ϵ_6 .

The particular solution we have obtained on the forward sweep satisfies (6.26) and the given boundary conditions everywhere except on the block boundaries. More importantly, the last block contains the exact solution since the computation of the correction vector ϵ_6 is based upon the known boundary conditions along B_2 .

The backward sweep now corrects the errors introduced in the particular solution by guessing the solutions along the block boundaries. Using (6.37) and (6.38) we calculate ϵ_4 and ϵ_5 for the homogeneous sweep of the last block and obtain the total solution for the last block, which has two rows in common with the second block. To obtain the homogeneous solution for the second block we need to find the difference ϵ_5 between the particular solution on the last row of the block and the exact solution. One method of obtaining ϵ_5 is to store the particular solution from the forward sweep and subtract it from the exact solution. However, it is much easier to recompute the particular solution by backing up three rows into the second block and repeating a small segment of the forward sweep. After ϵ_5 is obtained we sweep the second block to obtain the homogeneous solution, which is then added to the particular solution. This procedure is repeated until we finish the blocks. Error reduction iterations are applied at this step also.

The stabilization of EVP by this method is achieved by the introduction of the artificial boundaries. Propagation of error is very severe in EVP, and when the error exceeds the accu-

racy of the computer it cannot be corrected by addition of a homogeneous solution. The introduction of artificial boundaries before the error becomes too large limits the error in each block. Since the influence matrices for small blocks have small error levels, the artificial solution generated by the introduction of these boundaries can be easily corrected to obtain an extremely accurate final solution.

As a test problem for SEVP, Poisson's equation is solved over a square region with zero homogeneous boundary conditions. The test problem is also solved with the Lindzen-Kuo (Lindzen and Kuo, 1969), Crout (1941), and SOR methods. The former two are direct solvers. The relaxation by SOR is stopped when the normalized error (normalized with the forcing function) reaches 10^{-4} . A comparison of the computing time taken by these methods and SEVP is given in Fig. 6-3. When the grid dimension exceeds 60 for Lindzen-Kuo (L-K) and 50 for Crout methods the computer memory is exhausted. It is clear from the figure that the SEVP method is more efficient than the other three methods. For $M = N = 40$, the SEVP method is about two times faster than L-K and Crout methods and 20 times faster than SOR. The computation times given in Fig. 6-3 were obtained with double precision (56 bit precision) operations on the Texas Instruments ASC.

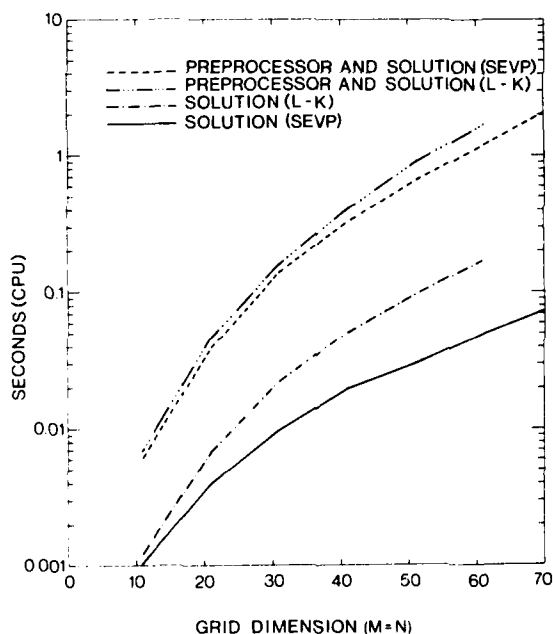


Fig. 6-3 — A comparison of the computing time requirements for the SEVP, Lindzen-Kuo, Crout and SOR methods.

Figure 6-4 shows the computer time required by the preprocessor step and the solution step of SEVP on a vector computer. The ratio of computing time required by the preprocessor step to that required by the solution step is about 3 for $M = N = 10$ and increases with increase in grid dimension. The L-K method can also be separated into a preprocessor step

(computing α matrices) and a solution step. For the test problem single precision on a 32-bit-word computer is adequate for the L-K method.

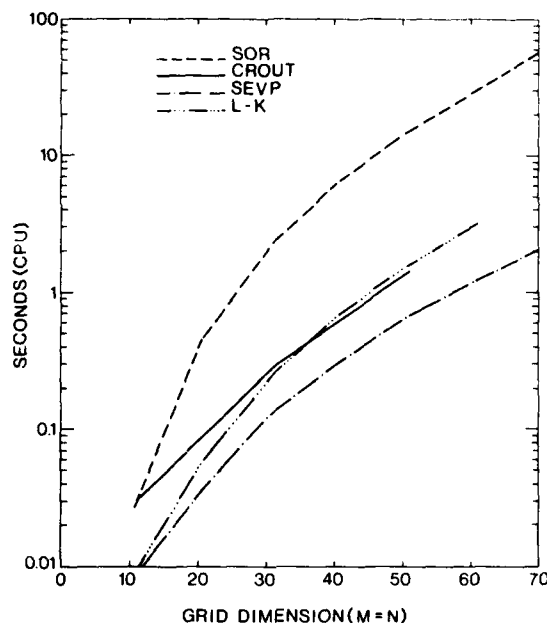


Fig. 6-4 — Computer time taken by the preprocessor and solution steps of the SEVP method.

Figure 6-5 gives a comparison of the computer time taken by the single precision Lindzen-Kuo and double precision SEVP methods to solve the test problem. The top two curves are the total computer time (preprocessor and solution steps) required, while the lower two curves are for the solution steps only. It is clear from the figure that the SEVP method is faster than the Lindzen-Kuo method for both the preprocessor and solution steps. ICCG (not shown) performed slightly worse than Crout because of relatively incomplete vectorization.

Table 1 gives the auxiliary memory requirements for the three direct methods for a 32-bit-word computer using double precision. It is clear from the table that the SEVP method's auxiliary memory requirement is an order of magnitude smaller than that required by the other two direct methods.

If $NB(K)$ represents the maximum number of points in the marching direction of the K th block, then

$$NB(K) \leq PA, \quad (6.39)$$

where P is a constant which depends only on the computer precision and A represents the minimum value of $CY_{I,J} CX_{I,J}^{-1}$ if we march in J or of $CX_{I,J} CY_{I,J}^{-1}$ if we march in I . It is clear from this equation that we can reduce the auxiliary memory required by the SEVP

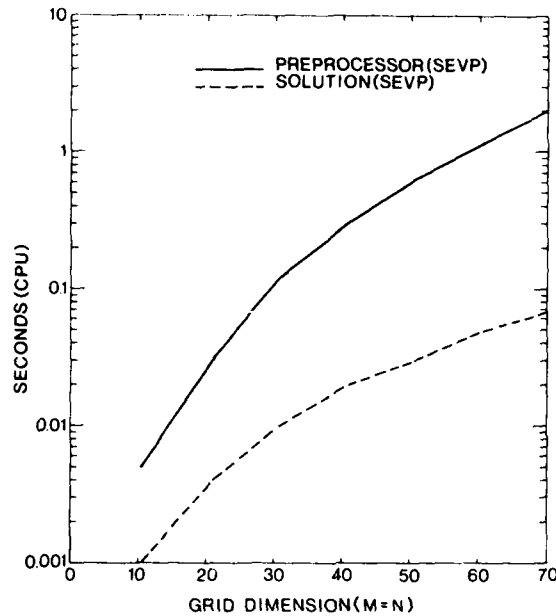


Fig. 6-5 — A comparison of the computer time requirements of single precision Lindzen-Kuo and double precision SEVP of methods using a vector computer with 32 bit word length. The top curves give the total time taken by both steps: the preprocessor and the solution steps. The lower two curves give the time taken by the solution steps.

Table 1. A comparison of the auxiliary memory requirements (in thousands of words) for SEVP, Lindzen-Kuo and Crout methods on 32 bit word computer using double precision. the numbers in parentheses exceed the Total central memory available on the NRL ASC.

Grid size	Crout	SEVP	Lindzen-Kuo
10	4.4	0.2	2.8
20	33.2	2.4	17.6
30	111.6	5.4	57.6
40	262.4	16.0	134.4
50	510.0 ^a	25.0	260.0
60	(878.4)	50.4	446.4
70	(1391.6)	88.2	(705.6)
80	(2073.6)	115.1	(1049.6)
90	(2978.4)	178.2	(1490.4)
100	(4040.0)	220.0	(2040.0)

^aNormalized error is more than 10^{-4} .

method by choosing the marching direction in such a way that $A \geq 1$. It can also be shown that the method requires less computing time to solve the elliptic equation in the case when $A \geq 1$ than in cases $A < 1$. Therefore, the computing speed of the method increases with $|A|$. On the other hand, the Lindzen-Kuo and Crout methods deteriorate with the increasing A and became unstable for $A \geq 100$.

This procedure for solving elliptic equations with Dirichlet boundary conditions can easily be extended to other boundary conditions (Neumann, periodic or mixed). It can also be used to solve elliptic equations over a region with irregular boundaries.

Another kind of generalization is also possible. Two-dimensional elliptic difference equations with nine-point stencils (over three or five consecutive rows and columns) can be solved by using the EVP and SEVP methods with a few minor modifications. For elliptic difference equation with a nine-point stencil over three consecutive rows and columns, a general equation can be written

$$\begin{aligned} & AY_{IJ}^3 X_{I-1,J+1} + BY_{IJ}^3 X_{I,J+1} + CY_{IJ}^3 X_{I+1,J+1} = \\ & F_{IJ} - AY_{IJ}^1 X_{I-1,J-1} - BY_{IJ}^1 X_{I,J-1} - CY_{IJ}^1 X_{I+1,J-1} \\ & - AY_{IJ}^2 X_{I-1,J} - BY_{IJ}^2 X_{I,J} - CY_{IJ}^2 X_{I+1,J}. \end{aligned} \quad (6.40)$$

To obtain the homogeneous and particular solutions we use the marching procedure in the j th direction by computing the solution explicitly at every point in the $(j+1)$ th row in terms of the j th and $(j-1)$ th rows. Since (6.40) couples the solution at three consecutive points on the $(j+1)$ th row, we instead compute the solution at all points on the $(j+1)$ th row simultaneously with a tridiagonal solver. With this minor modification, the EVP and SEVP methods described can be used to solve (6.40). The use of the tridiagonal solver increases the number of operations by $8(M-2)^2(N-1)$ for the preprocessor step and by $8(M-2)^2(2+\alpha)$ for the solution step.

A general elliptic difference equation using nine-point stencil over five consecutive rows and columns can be written

$$\begin{aligned} & CY_{IJ}^2 X_{I,J+2} = F_{IJ} - CY_{IJ}^1 X_{I,J+1} - AX_{IJ}^2 X_{I-2,J} - AX_{IJ}^1 X_{I-1,J} \\ & - BB_{IJ} X_{IJ} - CX_{IJ}^1 X_{I+1,J} - CX_{IJ}^2 X_{I+2,J} - AY_{IJ}^1 X_{I,J-1} - AY_{IJ}^2 X_{I,J-2}. \end{aligned} \quad (6.41)$$

The solution of this fourth-order finite-difference equation requires four boundary conditions in each direction, which may be specified as the solution on the two outermost rows and columns of the region. As before we obtain the required solution by superposing a particular and a homogeneous solutions. The particular solution is obtained by marching in the j th direction using the given solution (boundary condition) on the first two rows and a guess solution on the next two rows. The result satisfies (6.41) with the given forcing function and the boundary condition everywhere except on the boundary representing the last two rows. Since the homogeneous solution is the deviation of the particular solution from the required solution, it will satisfy (6.41) with zero forcing function everywhere and homogeneous boundary conditions on all the boundaries except on the last two rows. On these two rows the boundary conditions are obtained by subtracting the particular solution from given boundary conditions. Following the procedure described earlier using the homogeneous equation, we compute an influence matrix relating a vector representing the homogeneous solution vectors on the third and fourth

rows with that to the last two rows. Since each of these vectors has $2(M - 4)$ grid points, we use $2(M - 4)$ unit vectors to obtaining the influence matrix. Using the influence matrix and the boundary conditions for the homogeneous solution on the last two rows, we can obtain the error solution on the third and fourth rows. With these values for the error solution on the third and fourth rows and zero boundary conditions on the first two rows, first two columns and last two columns, we can obtain the solution for the homogeneous equation everywhere in the interior by marching in the j th direction.

The modifications described above to solve Eq. (6.41) by the EVP method can be easily incorporated in the SEVP procedure. As before, the region is divided into a number of blocks in such a way that each block is stable for the EVP method. The first and last two rows and columns of each block are the boundaries. Thus any two consecutive blocks now have four rows in common. The preprocessor step consists of computing influence matrices for each block that relate the homogeneous solution on the third and fourth rows to the last two rows. We also compute an influence matrix for every two consecutive blocks to relate the homogeneous solution on the first two common rows with the homogeneous solution on the next two common rows. During the forward sweep of the solution step we obtain an approximate solution, given guess values at the block boundaries where the solution is unknown. During the backward sweep these boundary conditions are connected to obtain the required solution. The procedure is similar to the one described earlier.

The size of each influence matrix described above is approximately four times the size of the corresponding influence matrix associated with Eq. (6.27). Thus the EVP and SEVP methods require four times more auxiliary memory to solve a fourth-order finite-difference equation than a second-order equation. It can also be shown that the preprocessor step takes approximately eight times more computing time and the solution step takes approximately four times more computing time.

6.3 Application of Chebychev Iteration to Nonself-Adjoint Equations

For some problems requiring the solution of an elliptic equation, attainment of solution to roundoff accuracy may not be required. If a certain level of residual error can be tolerated, and a good trial solution is available (as in the case of many time integration problems), an iterative solution of sufficient accuracy may in some cases be obtainable in a fraction of the time required for a complete solution. This generally requires that the equation be well conditioned in some sense, and that the residual error not have a cumulative effect on parameters of interest. We will not attempt to give criteria here for determining when iteration is preferable to direct solution, but present a method that has been used successfully in that context (McDonald, et al., 1978).

The computational mathematics literature is well stocked with iterative methods for solution of self-adjoint linear operator equations. However, many fewer approaches are offered for nonself-adjoint problems. One occasionally encounters the suggestion that the equation $L(\mathbf{r})\phi(\mathbf{r}) = S(\mathbf{r})$ (with L the linear operator, and S the known driving term) be made self-adjoint by an extra application of the adjoint of L : $L^*L\phi = L^*S$. There may be cases in which this approach has merit. However, it has two immediate drawbacks: (1) one has to work with a higher order equation, and (2) the ratio of maximum to minimum eigenvalue amplitude, an indicator of the amount of numerical work required, is squared. There are situations in which

it may be desirable to transform a self-adjoint problem into a nonself-adjoint one. For example, in a temperature or salinity diffusion problem, L contains the operator $\nabla \cdot K \nabla$, where K is the diffusivity. If K varies greatly in magnitude, the eigenvalue span of L will be increased accordingly, making the numerical solution more difficult. A possible remedy is to divide through by K , so that L contains the nonself adjoint operator $\nabla^2 + \nabla \log K \cdot \nabla$.

With some adaption, ICGG may hold promise for nonself-adjoint problems. But the factorization stage is awkward for vector computers. As indicated earlier, factorization of a matrix into upper and lower triangular matrices requires many scalar operations. Thus we wish to describe an alternative method developed by McDonald (1977) which allows complete vectorization (parallel processing) for all interior mesh points of a multidimensional grid. This method is a generalization of the Chebychev semi-iterative method (Section 6.1).

We seek an iterative solution to the equation

$$L\phi = S, \quad (6.40)$$

where S is a known source vector, and L is a matrix or finite difference operator. We assume L has complex eigenvalues $\lambda_r + i\lambda_i$, with all λ_i being of the same sign. We also assume all λ fall within an ellipse in the complex plane (see Fig. 6-6) whose major axis coincides with the real axis. The intersections of the ellipse with the real axis are $b - a$ and $b + a$, with

$$b/a > 1. \quad (6.41)$$

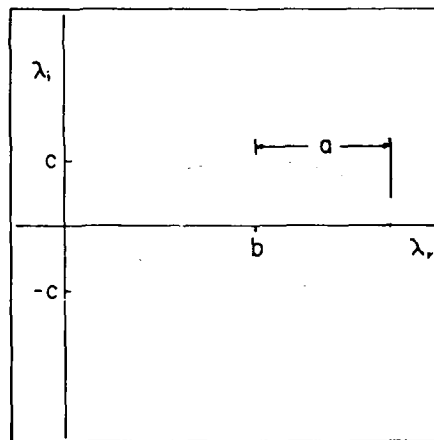


Fig. 6-6 - Ellipse containing complex eigenvalues.

The semiminor axis of the ellipse is c . We require

$$c < |a|, \quad (6.42)$$

and thus

$$\lambda_{i_{\max}} - \lambda_{i_{\min}} < \lambda_{r_{\max}} - \lambda_{r_{\min}}. \quad (6.43)$$

We assume that (6.40) possesses an exact solution Φ . At the end of n iterations, we will have an approximate solution ϕ^n , whose error is defined to be

$$\epsilon^n = \phi^n - \Phi. \quad (6.44)$$

The iterative method is to be such that

$$\epsilon^n = P_n(L)\epsilon^n, \quad (6.45)$$

where P_n is a polynomial of degree n . Substitution of (6.44) into (6.45) gives

$$\phi^n = P_n(L)\phi^n - [P_n(L) - 1]\Phi. \quad (6.46)$$

We do not know Φ in advance, but we do know $L^k\Phi = L^{k-1}S$ for $k > 0$. Thus from (6.46) we require that the constant term of $P_n(L)$ be unity:

$$P_n(0) = 1. \quad (6.47)$$

We wish to choose P_n such that its magnitude is as small as possible everywhere within the ellipse containing eigenvalues of L .

The problem of minimizing the maximum value of $|P_n(x)|$ subject to $P_n(0) = 1$ has a well-known solution when x is restricted to real values between $b - a$ and $b + a$ with $b/a > 1$. The standard argument points out that the desired P_n is such that all maxima of $|P_n|$ have the same value. One immediately determines that P_n is proportional to a Chebychev polynomial. The Chebychev polynomials T_n are such that

$$T_n(\cos \alpha) = \cos n\alpha, \quad (6.48a)$$

and

$$T_n(\cosh \alpha) = \cosh n\alpha. \quad (6.48b)$$

That $\cos n\alpha$ is a polynomial in $\cos \alpha$ and results from the elementary identity

$$\cos(m+1)\alpha = 2\cos \alpha \cos m\alpha - \cos(m-1)\alpha. \quad (6.49)$$

Equation (6.49) is also valid when $\cosh$ is substituted for $\cos$. Thus if $\cos m\alpha$ and $\cos(m-1)\alpha$ are polynomials in $\cos \alpha$, then so is $\cos(m+1)\alpha$. This is the case for $m = 0$ and 1 , so it is also true for all $m > 0$. Equation (6.49) also gives the recursion formula for the Chebychev polynomials:

$$T_{m+1}(x) = 2xT_m(x) - T_{m-1}(x). \quad (6.50)$$

From (6.48) we can see that $|T_n(x)|$ reaches a limiting value of unity $n+1$ times as x varies from -1 to 1 . Thus the solution to the min-max problem for real x is

$$P_n(x) = T_n\left(\frac{x-b}{a}\right) / T_n\left(-\frac{b}{a}\right), \quad (6.51)$$

and

$$|P_n|_{\max} = |T_n(-b/a)|^{-1} = 1/\cosh[n\cosh^{-1}(b/a)] < 1. \quad (6.52)$$

Slightly less well known is the fact that the Chebychev polynomials also satisfy the following problem: With

$$z = \xi + i\eta \quad (6.53)$$

find $P_n(z)$ such that $P_n(0) = 1$, and such that every maximum of $|P_n(z)|$ on the ellipse

$$\left(\frac{\xi - b}{a}\right)^2 + \frac{\eta^2}{c^2} \leq 1 \quad (6.54)$$

has the same value.

Let us now consider the behavior of the Chebychev polynomials in the complex plane. It is convenient to express ξ and η in terms of dimensionless variables ξ' , η' , α , and β as follows:

$$\begin{aligned} \xi &= \lambda_o \xi' \\ \eta &= \lambda_o \eta' \\ \xi' &= \cos \alpha \cosh \beta \\ \eta' &= \sin \alpha \sinh \beta. \end{aligned} \quad (6.55)$$

Here λ_o is a dimensional quantity of arbitrary magnitude having the same dimensions as the operator L . Let us first demonstrate that arbitrary ξ' and η' can be expressed as in (6.45) with real α and β . One can eliminate $\cos \alpha$ from (6.45) and obtain the following equation for $\cosh \beta$:

$$F(u) \equiv u^2 - u(\xi'^2 + \eta'^2 + 1) + \xi'^2 = 0, \quad (6.56)$$

where $u = \cosh^2 \beta$. Incidentally, one obtains the same equation by eliminating $\cosh \beta$ and setting $u = \cos^2 \alpha$. Note from (6.56) that

$$\begin{aligned} F(0) &= \xi'^2; \\ F(1) &= -\eta'^2; \\ F(\xi'^2 + \eta'^2 + 1) &= \xi'^2. \end{aligned} \quad (6.57)$$

We see immediately that there must be one real root with $0 \leq u_1 \leq 1$ and another real root $1 \leq u_2 \leq 1 + \xi'^2 + \eta'^2$. The lesser of the two roots can be taken to be $\cos^2 \alpha$, and the greater to be $\cosh^2 \beta$, with α and β both real. The signs of ξ' and η' can be adjusted properly by proper choice of the sign of β and phase of α modulo π .

Let us return to (6.55) and define

$$\begin{aligned} z' &= \xi' + i\eta' \\ &= \cos \alpha \cosh \beta + i \sin \alpha \sinh \beta \\ &= \cos(\alpha + i\beta). \end{aligned} \quad (6.58)$$

From (6.44) and (6.58) we have

$$\begin{aligned} T_n(z') &= \cos(n\alpha + in\beta) \\ &= \cos n\alpha \cosh n\beta + i \sin n\alpha \sinh n\beta. \end{aligned} \quad (6.59)$$

From (6.59) we find with only a slight amount of algebra

$$|T_n(z')|^2 = \cos^2 n\alpha + \sinh^2 n\beta. \quad (6.60)$$

We can see from (6.55) that surfaces of constant β are ellipses in ξ' - η' space:

$$\xi'^2 + (\eta'/\tanh \beta)^2 = \cosh^2 \beta. \quad (6.61)$$

With β fixed, one can see from (6.55) that the point (ξ', η') scans once around the ellipse as α varies from 0 to 2π . Thus (6.60) shows that $|T_n(z')|$ reaches the same maximum value $2n$ times as z' scans around the perimeter of the ellipse of constant β .

It remains only to show that no maxima occur in the interior of the ellipse (6.61). We can conclude this from the fact that $T_n(z')$ has n distinct real roots:

$$T_n(z') = K_n \prod_{m=1}^n \left[z' - \cos \frac{(m-1/2)\pi}{n} \right], \quad (6.62)$$

where K_n is a normalization coefficient. Thus

$$|T_n(z')|^2 = K_n^2 \prod_{m=1}^n \left[\left(\xi' - \cos \frac{(m-1/2)\pi}{n} \right)^2 + \eta'^2 \right]. \quad (6.63)$$

This shows that $|T_n(z')|$ increases monotonically away from the real axis ($\eta' = 0$), and thus there are no local maxima. Thus all maxima in the region enclosed by the ellipse (6.61) must occur on the boundary. As pointed out by (6.60), these maxima of $|T_n(z')|$ all have the same value. Therefore, we conclude that the Chebychev polynomials solve the problem stated just prior to (6.54).

In order to put the ellipse (6.54) into the form (6.61), set

$$\begin{aligned} \xi &= b + af\xi' \\ \eta &= af\eta' \end{aligned} \quad (6.64)$$

where f is a dimensionless number yet to be specified. Substitution of (6.64) into (6.54) gives for the parameter of the ellipse

$$\xi'^2 + (\eta'a/c)^2 = f^{-2}. \quad (6.65)$$

Thus from (6.51) we take

$$\begin{aligned} \beta &= \tanh^{-1} c/a; \\ f &= 1/\cosh \beta = (1 - c^2/a^2)^{1/2}. \end{aligned}$$

The desired polynomial is proportional to $T_n(\xi' + i\eta')$, and in fact is

$$P_n(z) = T_n\{(z-b)/fa\}/T_n(-b/fa). \quad (6.67)$$

Note that in the limit $c \rightarrow 0$, we have $f \rightarrow 1$, and $z \rightarrow \xi$. In this limit the complex solution (6.67) reduces to the real solution (6.51), as it should.

From (6.60) we find that the maximum amplitude attained by (6.67) for the ellipse of constant β is

$$|T_n(z')|_{\max} = (1 + \sinh^2 n\beta)^{1/2} = \cosh n\beta. \quad (6.68)$$

Substituting this into (6.67) gives

$$|P_n(z)|_{\max} = \frac{T_n(\cosh \beta)}{T_n(b/a \cosh \beta)} < 1, \quad (6.69)$$

where β is given in (6.66). We have dropped the minus sign from the argument of the denominator of (6.67) because $T_n(x)$ is an even or odd function of x when n is even or odd, respectively. We know that (6.69) is less than unity because the argument in the denominator is

larger than that in the numerator and $T_n(x)$ is monotonically increasing for $x > 1$, as one can show from (6.48).

We take the polynomial of (6.67) to be the proper choice for use in constructing the approximate solution to (6.40) according to (6.46). We must choose between two approaches for generating the polynomial $P_n(L)$. The first is factorization of P_n into n linear factors. We have tried this approach and found it subject to roundoff error amplification. This is because the process is equivalent to n overrelaxation steps, some of which reduce the long-wavelength errors at the expense of increasing the short-wavelength errors. The short-wavelength errors are eventually brought back down, but any roundoff error introduced while they were large tends to contaminate the final result.

The second and preferable approach to constructing $P_n(L)$ is through the use of a recursion formula. The only real drawback to this method is that one extra array of storage is required for retention of earlier iterates. We substitute (6.67) into (6.46) to obtain

$$\phi^{n+1} = \frac{T_{n+1}[(L-b)/fa] (\phi^n - \Phi)}{T_{n+1}(-b/fa)} + \Phi. \quad (6.70)$$

In the numerator of this expression let us express T_{n+1} in terms of T_n and T_{n-1} according to the recursion formula (6.50), and then express T_n and T_{n-1} in terms of ϕ and ϕ^{n-1} according to (6.70). Let us define

$$q = -b/fa = -b/\sqrt{a^2 - c^2} \quad (6.71)$$

and use $L\Phi = S$. Then we find

$$\begin{aligned} \phi^{n+1} &= \frac{1}{T_{n+1}(q)} [(2T_n(q)/fa)((L-b)\phi^n - S) - T_{n-1}(q)\phi^{n-1}] \\ &+ \Phi [1 + (-2qT_n(q) + T_{n-1}(q))/T_{n+1}(q)]. \end{aligned} \quad (6.72)$$

Notice that the coefficient of Φ is zero by (6.50). Recall that we imposed this condition at the outset in (6.47).

To use this method we must know the first two terms in the recursion. We get these by direct construction of $P_n(L)$, and obtain

$$\begin{aligned} \phi^0 &= \text{arbitrary trial solution;} \\ \phi^1 &= \phi^0 - (1/b)(L\phi^0 - S); \\ \phi^{n+1} &= \frac{2T_n(q)}{T_{n+1}(q)\sqrt{a^2 - c^2}} ((L-b)\phi^n - S) - \frac{T_{n-1}(q)}{T_{n+1}(q)} \phi^{n-1}, \quad n > 1. \end{aligned} \quad (6.73)$$

If B is an eigenvector of L such that

$$LB = \lambda B, \quad (6.74)$$

where λ is a complex constant, then

$$P_n(L)B = P_n(\lambda)B. \quad (6.75)$$

Let us imagine that the error ϵ^n of (6.44) is expressed as a linear combination of the eigenvectors of L . We have assumed all eigenvalues of L to lie within an ellipse (6.54) in the complex

plane. Then from (6.45), (6.66), and (6.69) the coefficients of the eigenvector expansion of ϵ'' have each been decreased in amplitude by at least a factor

$$|P_n(\lambda)| \leq E_n(a, b, c) \equiv T_n(a/\sqrt{a^2 - c^2})/T_n(b/\sqrt{a^2 - c^2}). \quad (6.76)$$

In Fig. 6-7 we plot this error limit as a function of n for various values of the ellipse axis ratio c/a and the ratio of maximum to minimum real eigenvalue,

$$R \equiv \frac{b+a}{b-a}. \quad (6.77)$$

One notices that the error limit decreases approximately exponentially with n , and that the rate of decrease depends strongly on R . In practice, when R is greater than about 10^3 we use a filtering process to be described later to bring down the effective value of R . The slow convergence at large R is due to the fact that one is attempting to reduce the error by a diffusion process which is slow for long wavelengths.

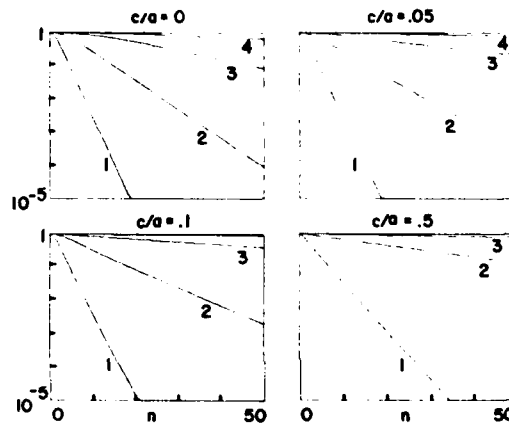


Fig. 6-7 — Error estimate E_n versus n .
Curves are labelled with $\log_{10} R$.

Let us derive an approximate expression for (6.76) in the limit of large R . It is convenient to define

$$\delta = b/a - 1 = 2/(R - 1); \quad (6.78a)$$

$$\rho = c/a. \quad (6.78b)$$

From (6.48) and the identity

$$\cosh^{-1} x = \log(x + \sqrt{x^2 - 1}) \quad (6.79)$$

we have

$$E_n = \frac{\cosh(n \log Q_1)}{\cosh(n \log Q_1 + n \log Q_2)} \quad (6.80)$$

$$= 1/[\cosh(n \log Q_2) + \tanh(n \log Q_1) \sinh(n \log Q_2)],$$

where

$$Q_1 = \frac{1+\rho}{\sqrt{1-\rho^2}} \quad (6.81a)$$

$$Q_2 = \frac{1+\delta+\sqrt{2\delta+\delta^2+\rho^2}}{1+\rho}. \quad (6.81b)$$

Since Q_1 and Q_2 are both greater than 1, (6.80) yields

$$2 \exp(-n \log Q_2) > E_n > \exp(-n \log Q_2). \quad (6.82)$$

Thus the estimate

$$E_n \approx \sqrt{2} \left(\frac{1+\delta+\sqrt{2\delta+\delta^2+\rho^2}}{1+\rho} \right)^{-n} \quad (6.83)$$

is always correct within a factor of $\sqrt{2}$. Defining the convergence rate to be

$$C \equiv -\partial \log E_n / \partial n, \quad (6.84)$$

we can find simple expressions for C in the limit $\delta \ll 1$. For $\rho \ll \delta$, $C \approx \sqrt{2\delta}$. When $\rho \gg \delta$, then $C \approx \delta/\rho$. This shows that convergence slows down when the eigenvalues of the operator L have significant imaginary parts.

As a test of the method (6.72)-(6.73) and the error estimate (6.76) we solve numerically the two-dimensional equation

$$\nabla^2 \phi + \mathbf{A}(x,y) \cdot \nabla \phi = S(x,y), \quad (6.85)$$

on a rectangular grid with constant mesh spacing subject to doubly periodic boundary conditions.

The mesh is assumed to have $K_x \times K_y$ interior mesh points. We choose the x direction such that

$$K_x \geq K_y. \quad (6.86)$$

We also assume the mesh intervals to be

$$\delta x = \delta y = \text{constant}, \quad (6.87)$$

We use redundant guard cells around the perimeter of the mesh for efficient handling of the boundary conditions. This results in a complete mesh of $(K_x + 2)(K_y + 2)$ points. We use the second-order five-point representation of the derivatives in (6.85):

$$L\phi_{IJ} \equiv (\phi_{I+1,J} + \phi_{I-1,J} + \phi_{I,J+1} + \phi_{I,J-1} - 4\phi_{IJ})/\delta x^2 \\ + AX_{IJ}(\phi_{I+1,J} - \phi_{I-1,J})/2\delta x + AY_{IJ}(\phi_{I,J+1} - \phi_{I,J-1})/2\delta x. \quad (6.88)$$

Here I and J are the x and y mesh point indices, and AX and AY are the x and y components of $\mathbf{A}$ respectively.

Optimizing over all ellipses containing the eigenvalues of (6.85), McDonald (1977) finds an approximate convergence rate

$$C \approx (2\delta + \rho^2)^{1/2} - \rho \approx (\sigma/\sqrt{2}) C_o(\mu). \quad (6.89)$$

Optimizing the convergence rate with respect to an oblateness parameter θ , one obtains ellipse parameters as follows: μ

$$b = 4/\delta x^2 \quad (6.90)$$

$$\begin{aligned}\delta &\approx \frac{1}{4} \sigma^2 (1 - 2\theta) \\ \rho &= \mu \sigma \theta^{-1/2} \\ \theta &= (\mu^2/4)^{1/3} [(16\mu^2 + 1)^{1/2} + 1]^{1/3} - [(16\mu^2 + 1)^{1/2} - 1]^{1/3},\end{aligned}$$

where σ is the minimum nonzero value over all wavenumbers k_i , k_j consistent with boundary conditions, of the expression $(\sin^2 k_i + \sin^2 k_j)^{1/2}$.

For doubly periodic boundaries, $\sigma \approx 2\pi/K_1$. The argument μ is defined by

$$\mu = [(AX\delta x)^2 + (AY\delta y)^2]^{1/2}/4\sigma. \quad (6.91)$$

The function $C_o(\mu)$, plotted in Fig. 6-8, satisfies

$$C_o(\mu) = \sqrt{2}[(1/2 - \theta + \mu^2/\theta)^{1/2} - \mu(\theta)^{-1/2}] \approx (9.5\mu + 1)^{-1}. \quad (6.92)$$

The dependence of θ upon μ is shown in Fig. 6-9.

Evaluating (6.90) for large and small μ , we find the following limiting forms for C :

$$\begin{aligned}C &\approx \sigma/\sqrt{2} (1 - 3(\mu^2/2)^{1/3}), \quad \mu \ll 1 \\ C &\approx \sigma\mu^{-1}/6\sqrt{6}, \quad \mu > 1.\end{aligned}$$

This shows that as μ increases from 0 to 1, the convergence rate drops by approximately a factor of 10.

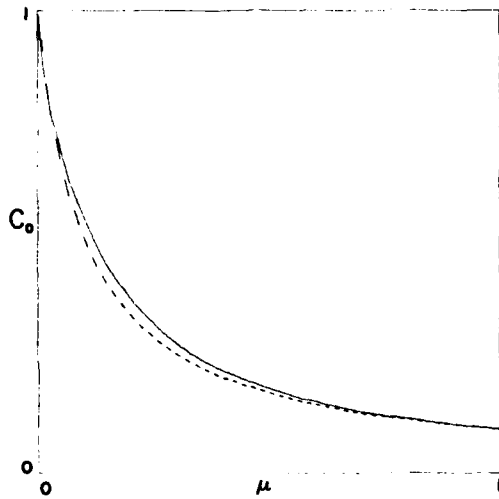


Fig. 6-8 — Normalized convergence rate (6.92) (solid); approximation (7) (dashed).

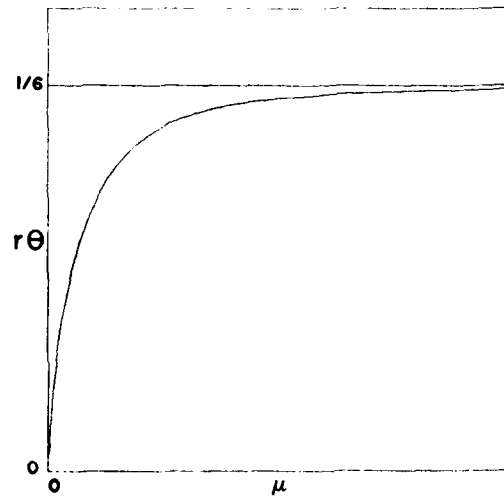


Fig. 6-9 — Optimal tuning parameter as determined from (6.90).

We can see from (6.89) that the number of iterations required to reach a certain level of error reduction is proportional to K_x . Thus the total number of operations required is proportional to $K_x^2 K_z$. Therefore it is advantageous to correct the long-wavelength components of the solution on a coarse mesh, and interpolate onto a fine mesh before completing the solution. The interpolation introduces truncation error into the solution, so that the error estimates derived earlier may not hold on the large mesh. For this reason, it is best to alternate between large and small meshes more than once, attempting only a modest error reduction with each pass. In practice, use of a coarse grid becomes important for $K_x \geq 64$. For a grid of 128×128 interior points, a coarse grid of 32×32 interior points is used. First, the residual $L\phi'' - S$ is extracted on the fine grid. Then this residual and A are defined on the coarse grid by using block averages of 16 fine grid points per coarse grid point. The coarse grid potential correction is initialized to zero and a large number of iterations is performed. Typically we use three times as many iterations on the coarse grid as on the fine grid. Then the potential correction is defined on the fine grid by using bilinear interpolation, and the result is subtracted from ϕ'' . Then iterations are performed on the fine grid, making no attempt to make significant improvements in the lowest 5 to 10 modes of the solution. That is, we arbitrarily increase σ by a factor of 5 to 10. This results in improved convergence of the higher modes. The effectiveness of this fine grid-coarse grid approach may be improved significantly in a time-dependent problem by two-level extrapolation for the trial solution ϕ'' .

As a test of the iterative procedure (6.72)-(6.74) and the convergence estimate (6.93) we have solved (6.85) on meshes of 32×32 and 48×48 interior grid points without regridding; and on a 128×128 mesh with a reduced mesh of 32×32 interior points. All tests used doubly periodic boundary conditions. The numerical convergence rate comparisons were carried out as follows. Arbitrary forms were adopted for A and a reference solution Φ . Then a source term was generated from Φ numerically using the difference operation (6.88). This source term was used in the iteration (6.72)-(6.73), with the approximate solution being initialized to zero. After a large number N of iterations (usually $N = 40$), a relative error E was defined to be the root mean square residual of (6.85) divided by the root mean square of the source S . The average convergence rate was then taken to be $-\log E/N$.

In all cases convergence was fast enough to be consistent with (6.89), and in most cases was faster than (6.89). The one case in which convergence was just equal to (6.89) for all μ was for $\Phi_{ij} = \sin(2\pi i/K_x)$, and $A_x = \text{constant}$, $A_z = 0$. This is consistent with the previous discussion. For determining μ in (6.91), we used the root mean square value for $|A|$, and $r = 1$.

Figure 6.10 shows convergence rates per second of computer time, $-t^{-1} \log E$ obtained by using the two-pipe NRL TI ASC to solve the same set of problems with the Chebychev explicit method (CE, upper curve) and with an alternating direction implicit method (ADI, lower curve). These comparisons were made on a 50×50 mesh using data from the EJET plasma turbulence code (McDonald et al., 1975). The ADI solution used logarithmically spaced iteration parameters and a partially vectorized tridiagonal solver. Although ADI converged faster *per iteration* the limited vectorizability of the method resulted in slow execution. Thus the result in Fig. 6-10 is machine dependent. The execution times per iteration were 25.8 msec for ADI and 1.32 msec for the Chebychev explicit method. To illustrate the computational efficiency of the explicit method, the lower limit on execution time per iteration (neglecting overhead and boundary value resetting) would be $(48 \times 48 \text{ interior points}) \times (11 \text{ operations per point}) \times (40 \text{ nsec per operation}) = 1.01 \text{ msec}$. Thus the explicit method runs at 76%

efficiency on a modest 50×50 mesh. ADI could be made more competitive by the use of cyclic reduction rather than tridiagonal solution in the integration direction, but it is unlikely that any improvement would bring the convergence rate per second up to that of the explicit method. In addition, the boundary conditions are much easier to change in the explicit method than in the implicit one.

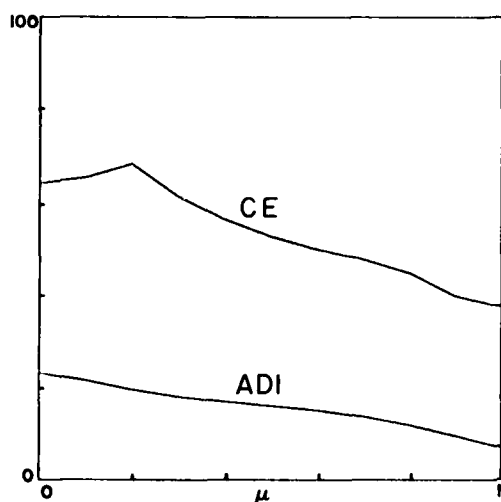


Fig 6-10 — Computational efficiencies of the Chebyshev explicit method (CE) and alternating direction implicit method (ADI) when applied to a series of numerical tests in which $|A|$ and thus μ vary from case to case. Plotted is the logarithm of the error reduction factor after many iterations, divided by the elapsed computer time in seconds.

7. SPECTRAL METHODS

7.1 A Survey of Spectral Methods

Spectral methods for solving initial value partial differential equations involve representing the dependent variables as linear superpositions of a set of basis functions of the independent variables. These functions are chosen so that their properties (e.g., analyticity, orthogonality, the property of satisfying boundary conditions, that of possessing some close connection with the physical processes in the problem, etc.) simplify the task of decomposition, i.e., calculating the series coefficients. The best known and most useful such representation is in terms of sinusoidal functions, i.e., Fourier analysis.

The process of superposition and its inverse, decomposition, involve all N basis functions. In contrast, second-order difference approximations involve just three points at a time. Consequently, to be competitive, spectral methods require very efficient transform methods. Such a method in the case of Fourier transforms is the fast Fourier transform (FFT) developed first by Cooley and Tukey (1965). Fast transform methods have been found for other orthogonal bases (Orszag, 1978), as will be seen.

Gottlieb and Orszag (1977) distinguish three different kinds of spectral methods: the Galerkin, tau, and collocation (pseudospectral) approximations. To understand how they are defined, consider the linear initial value problem of the form

$$\frac{\partial u(x,t)}{\partial t} = L(x,t) u(x,t) + f(x,t), \quad (7.1)$$

where boundary conditions of the form

$$B_{\pm}(x) u(x,t) = 0 \quad (7.2)$$

hold at $x = x_{\pm}$, where the $B_{\pm}$ are linear operators, and

$$u(x,0) = u_0(x). \quad (7.3)$$

We approximate u by a superposition of N basis functions $v_n(x)$ satisfying the boundary condition (7.2),

$$u(x,t) = \sum q_n(t) v_n(x). \quad (7.4)$$

If $\{v_n(x)\}$ form an orthonormal set, the coefficients a_n satisfy

$$a_n(t) = \int_{x_-}^{x_+} dx v_n(x) u(x,t). \quad (7.5)$$

Equation (7.1) then yields

$$\frac{da_n}{dt} = \sum_{m=1}^N \int_{x_-}^{x_+} dx v_n(x) L v_m(x) + \phi_n(t), \quad (7.6)$$

where ϕ_n is defined by

$$f(x,t) = \sum \phi_n(t) v_n(x). \quad (7.7)$$

Equation (7.6), an explicit equation for da_n/dt , is especially simple by virtue of our assumption that $\{v_n(x)\}$ are orthonormal. In the general case, it is necessary to solve simultaneously the N linear equations

$$\sum_{m=1}^N \left\{ \frac{da_m}{dt} \int_{x_-}^{x_+} dx v_n(x) v_m(x) - \int_{x_-}^{x_+} dx v_n(x) L v_m(x) \right\} = \phi_n. \quad (7.8)$$

Together (7.4) and (7.8) constitute the Galerkin approximation.

The tau approximation (Lanczos, 1956) starts again with (7.4), but $\{v_n(x)\}$ are not required to satisfy the boundary conditions. Instead, the two constraints

$$\sum_{n=1}^N a_n B_{\pm} v_n(x) = 0 \quad (7.9)$$

are applied. To leave the problem well posed, only the first $N-2$ of the equations (7.8) [or (7.6) if the basis functions are orthonormal] are retained.

For collocation methods, N points $\{x_n\}$ are chosen such that $x_- < x_1 < \dots < x_N < x_+$. The solution $u(x)$ of (7.1) is approximated by (7.4) where $\{a_n\}$ satisfy

$$\sum_{m=1}^N a_m v_m(x_n) = u(x_n), \quad (7.10)$$

$n = 1, 2, \dots, N$. For example, if $\{x_n\}$ are uniformly distributed between x_- and x_+ and $\{v_n\}$ are harmonic functions (sines and cosines), (7.11) just defines a discrete Fourier transform.

The time derivative may be differenced either explicitly or implicitly. The most popular explicit scheme is simple leapfrog. Runge-Kutta methods are also used. If they are of third or higher order, the linear stability condition has the same form as that for leapfrog,

$$\frac{(N-1)\bar{v}\Delta t}{x_+ - x_-} < C, \quad (7.11)$$

where $\bar{v}$ has the significance of an average or effective velocity and where C is a constant of order unity which depends on the scheme. (Note the similarity to the Courant condition in finite differences.)

Among implicit schemes, Crank-Nicholson and backwards Euler are stable for hyperbolic equations without restriction on the time step. Of course accuracy is another story. Even unconditionally stable schemes must satisfy (7.11) to be accurate. (Cf. the discussion in Section 2.2.)

All three spectral techniques can be generalized to multidimensions without difficulty. Clearly a large selection of different spectral approximations is possible, depending on which kind of technique is adopted, the choice of basic functions, and the order N of the approximation. These should reflect the physics of the problem and the accuracy requirements imposed on the desired solution. As always, an important consideration to be balanced against the latter is running time. This is mainly determined by the time required to perform the transform (7.5) and its inverse (7.4).

Orszag (1978) has concluded that the basis set should be chosen to consist of eigenfunctions of nonsingular Sturm-Liouville equations (e.g., sinusoidal functions) only for problems whose boundary conditions match those of the Sturm-Liouville problem. In all other cases, the basis should consist of eigenfunctions of a singular Sturm-Liouville problem, i.e., those for which the eigenvalue condition is that the solution remain well-behaved at some boundary. (Chebychev and Legendre polynomials are the most familiar examples.) This prescription is made to ensure that the Gibbs phenomenon does not arise in the solution at the boundary.

Such bases give rise to eigenvalue problems

$$\mathbf{L}\mathbf{u} = \mathbf{F} \quad (7.12)$$

whose spectral approximation

$$\mathbf{L}_s \mathbf{u}_s = \mathbf{F}_s \quad (7.13)$$

involves a full $N \times N$ matrix $\mathbf{L}_s$. In a 3D ($\sim 100 \times 100 \times 100$) problem, $N \approx 10^6$. Since storage of the matrix requires N^2 words and straightforward direct inversion involves N^3 operations, recourse must be had to another technique (Orszag, 1978). The trick is to find a sparse approximate operator $\tilde{\mathbf{L}}$ such that $\|\tilde{\mathbf{L}}^{-1} \mathbf{L}_s\| = O(1)$ as $N \rightarrow \infty$. If $\tilde{\mathbf{L}}$ can be inverted cheaply (see Section 6.1), then an iteration scheme, e.g.,

$$\tilde{\mathbf{L}}\mathbf{u}^{\text{new}} = \tilde{\mathbf{L}}\mathbf{u}^{\text{old}} - \omega (\mathbf{L}\mathbf{u}^{\text{old}} - \mathbf{F}), \quad (7.14)$$

where ω is a relaxation constant, yields machine accuracy in a finite number of iterations. Here $\tilde{\mathbf{L}}$ is chosen to be sparse so that storage requires only $O(N)$ words, and at the same time $\tilde{\mathbf{L}}$ is efficiently invertible. In practice $\tilde{\mathbf{L}}$ can be a finite difference approximation, invertible in $O(N)$ operations. Aside from these, the total operation count is proportional to $N \ln N$. (It

should be noted, however, that the constant of proportionality is much larger for, e.g., Bessel or Bernoulli functions than it is for the conventional FFT.)

Spectral methods are applicable to nonlinear problems, but the analysis of stability, accuracy, etc., of course becomes nontrivial. Following Gottlieb and Orszag (1977), we consider the problem of solution of the incompressible Navier-Stokes equation. This is written in the vorticity-stream function formulation. The difficulties arise from the nonlinear term in (2.6). Collocation is recommended for such problems. The key is to evaluate the derivative $\partial u / \partial x$ (where now u denotes the vorticity ζ) using the fast transform. E.g., for complex exponential basis functions,

$$\frac{\partial u}{\partial x} = \frac{2\pi}{x_+ - x_-} \sum_{in} a_n \exp \left(\frac{2\pi i n \Delta x}{x_+ - x_-} \right). \quad (7.15)$$

Then the solution is marched forward in configuration space at the collocation points. The number of operations for this procedure again goes as $N \ln N$. (We will see an example of this technique applied to another problem in the next section.)

Problems involving irregular boundaries (i.e., anything other than squares and circles) can be solved by conformal mapping. There is nothing hard about this in principle, but each reformulation or dynamical motion that results in a change in boundary conditions or topology (or extreme changes in geometry) necessitates recoding "by hand."

Spectral methods allow resolution to be concentrated where it is needed (for example, in a boundary layer), provided the location is known *a priori*. They cannot resolve moving near-discontinuities, e.g., at shock fronts and breaking internal waves. If no extreme gradients occur, spectral codes require much less resolution than finite-difference schemes (since they are in effect N th order accurate) and become proportionately more competitive.

7.2 Solution of One-Dimensional Nonlinear Wave Equations

In many problems of interest it is possible by means of asymptotic analysis to derive an approximate nonlinear equation involving a single fluid quantity, e.g., one component of the flow field. Typically it is necessary to assume for this purpose that the amplitude of the disturbance, though finite, is small and the horizontal scale is large compared with the vertical scale. Such a nonlinear wave equation is useful because, in addition to decreasing the number of dependent variables, it reduces the number of spatial dimensions by requiring the vertical dependence to be that of a linear eigenmode. Thus a three-dimensional process may sometimes be approximated as two-dimensional one; more commonly, a two-dimensional process is approximated by a one-dimensional wave equation.

Examples of such equations are the Korteweg-deVries (KdV) equation for surface water waves, and the Benjamin-Ono equation (Benjamin, 1967) for internal waves. Many of these equations are amenable to exact solution by the inverse scattering method (Ablowitz et al., 1974). Among the solutions so obtained, the most remarkable are those describing solitons. These are single bulges (or troughs), rather than trains of waves, and have the property of retaining their identity even when two or more collide and pass through one another. (Clearly this property is irrelevant to the characterization of a single isolated soliton.)

It is important to note, however, that solitons may result even if the inverse scattering technique and other analytic techniques are inapplicable; indeed, solitons were first observed by Zabusky and Kruskal (1965) in numerical solutions of the KdV equation. More significantly, solitons are seen in nature and in solutions of "exact" hyperbolic systems, i.e., systems of equations like (2.1)-(2.3) which have not been treated with the weak nonlinearity, long-wavelength approximations. Thus solitons are more general and more fundamental than our analytic techniques for deriving them. This means that for many purposes, simulation is an indispensable tool for their investigation.

For integrating nonlinear wave equations, a variety of techniques has been employed. Although explicit finite-difference schemes were used in the early numerical work (e.g., Zabusky, 1967) and to solve a modified form of the KdV equation having a dissipative term and a linear forcing term (Ott et al., 1973), the most efficient and accurate methods are spectral or quasi-spectral.

A straightforward technique for finding periodic solutions is to work in k space. For example, the Benjamin-Ono equation becomes

$$a \frac{\partial v_k}{\partial t} + b \sum_{k'} ik' v_{k'} v_{k-k} + ik |k| c v_k = 0, \quad (7.16)$$

where

$$v(x,t) = \sum_k e^{ikx} v_k. \quad (7.17)$$

Equation (7.16) can be treated as an array of ordinary differential equations, with a linear driving term or a quadratic one (Ott et al., 1973). Even though explicit, this technique is efficient when FFT's are used. Moreover, it is highly accurate, because the spatial derivatives are M th order accurate, where N is the number of modes retained in the sum (7.17). It has the additional advantage of being easily convertible into a 2D solver, provided one is willing to accept a limitation on the number of modes so that the matrix multiplication operation remains within manageable size.

Perhaps the best technique developed to date is the split-step Fourier method of Tappert (1974). [See also the discussion by Hyman in Lax (1975).] Suppose u satisfies

$$\frac{\partial u}{\partial t} + \frac{\partial}{\partial x} f(u) + L \frac{\partial u}{\partial x} = 0, \quad (7.18)$$

where f is some function of u , and L is a differential operator with constant coefficients,

$$L = \sum a_n \frac{\partial^n}{\partial x^n}. \quad (7.19)$$

Let F denote the Fourier transform operation. Then

$$F(Lu) = L(k) F(u), \quad (7.20)$$

where

$$L(k) = \sum a_n (ik)^n. \quad (7.21)$$

An important feature of the method, of course, is the use of fast transform routines for F .

To go from time t to $t + \delta t$, we proceed in two steps. First, advance the solution using only the nonlinear term. An implicit finite-difference scheme is used for this purpose:

$$\tilde{u}(t + \delta t) = u(t) - \frac{1}{2} \delta t \{Q[\tilde{u}(t + \delta t)] - Q[\tilde{u}(t)]\}, \quad (7.22)$$

where $Q(u)$ approximates $\frac{\partial f(u)}{\partial x}$. Then advance the solution exactly, using only the linear term, according to

$$u(t + \delta t) = F^{-1} [e^{iL(k)\delta t} F(\tilde{u})]. \quad (7.23)$$

This method is second-order accurate and linearly stable.

One concludes that ID nonlinear wave equations require careful treatment of the (non-linear) advective term, but do not suffer much from errors in approximating the time derivative. In this connection, Fornberg (1975) has shown that even high-order finite-difference approximations to the advection term are qualitatively less accurate than spectral methods, a result that seems specific to hyperbolic equations.

Tappert (1974) went on to study the "morphological stability" of the KdV equations with his method. That is, he verified that small perturbations added to $f(u)$ or to $L(k)$ did not affect the qualitative features of the solutions (number of solitons, their stability in mutual interactions, the existence of recurrence, etc.). Such perturbations can destroy the property (Miura, 1974) enjoyed by the KdV equation (and others soluble by the inverse scattering method) of possessing an infinite number of polynomial conserved quantities. This is another indication that soliton behavior is more general than the inverse scattering method.

Similar stability studies have apparently not yet been carried out with the Benjamin-Ono equation. A simple change in the analysis carried out by Tappert (1974) would permit this. More significant, however, are transverse perturbations. Tappert (unpublished) has recently performed some calculations on the KdV equation which indicates that transverse perturbations of (7.18) are stable if $L(k)$ is concave downward and unstable otherwise. If this were to hold in general, solutions of the Benjamin-Ono equation would always be unstable. More work is needed here.

REFERENCES

- Ablowitz, M.J., D.J. Kaup, A.C. Newell, and H. Segur, "The Inverse Scattering Transform—Fourier Analysis for Nonlinear Problems," *Stud. Appl. Math.* **53**, 249 (1974).
- Barcilon, A., D.L. Book, J. Boris, A.L. Cooper, K. Hain, P.C. Liewer, A.E. Robson, R. Shanny, P.J. Turchi, and N.K. Winsor, *Plasma Physics and Controlled Nuclear Fusion Research*, 1974, IAEA-CN-33, Vol. II, p. 567.
- Benjamin, T.B., "Internal Waves of Permanent Form in Fluids of Great Depth," *J. Fluid Mech.* **29**, 559 (1967).

- Birdsall, C.K. and D. Fuss, *J. Chem. Phys.* **3**, 494 (1969). See also A.B. Langdon and B.F. Lasinski, "Electromagnetic and Relativistic Plasma Simulation Models," chapter in *Methods in Computational Physics* (Academic Press, New York, 1976), Vol. 16.
- Book, D.L., J.P. Boris, and K.H. Hain, "Flux-Corrected Transport II: Generalizations of the Method," *J. Comp. Phys.* **18**, 248 (1975).
- Boris, J.P., "Flux-Corrected Transport Modules for Generalized Continuity Equations," NRL Memo. Rept. 3237 (1976)
- Boris, J.P., in *Proceedings of the Second European Conference on Computational Physics* (North Holland, 1976a).
- Boris, J.P. and D.L. Book, "Flux-Corrected Transport I: SHASTA—A Fluid Transport Algorithm That Works," *J. Comp. Phys.* **11**, 38 (1973).
- Boris, J.P. and D.L. Book, Flux-Corrected Transport. III. Minimal-Error FCT Algorithms, *J. Comp. Phys.* **20**, 397 (1976).
- Boris, J.P. and D.L. Book, "Solution of Continuity Equations by the Method of Flux Corrected Transport," *Methods in Computational Physics* (Academic Press, New York, 1976a), Vol. 16, Chap. 11.
- Boris, J.P., M.J. Fritts, and K.L. Hain, "Free Surface Hydrodynamics Using a Lagrangian Triangular Mesh," in *Proc. First Int'l. Conference on Numerical Ship Hydrodynamics*, NBS, Gaithersburg, Md., October 1975.
- Boris, J.P. and K.V. Roberts, "The Optimization of Particle Calculations in 2 and 3 Dimensions", *J. Comp. Phys.* **4**, 552 (1969).
- Boris, J.P. and R.A. Shanny, eds., *Proceedings of the Fourth Conference on Numerical Simulation of Plasmas*, Nov. 2-3, 1970, U.S. GPO #0851-00059. See also R.L. Morse, "Multidimensional Plasma Simulation by the Particle-in-Cell Method," *Methods in Computational Physics*, Alder, Fernbach, and Rotenberg, eds. (Academic Press, New York, 1970), Vol. 9.
- Brennan, C. and A.K. Whitney, "Unsteady Free Surface Flows; Solutions Employing the Lagrangian Description of the Motion," *Eighth Symposium on Naval Hydrodynamics*, Pasadena, August 1970.
- Buneman, O., "Time-Reversible Difference Procedures," *J. Comp. Phys.* **1**, 517 (1967).
- Buneman, O., "A Compact Non-iterative Poisson Solver," SUIPR Report No. 294, Inst. for Plasma Physics, Stanford University (1969).
- Chan, R., "A Generalized Arbitrary Lagrangian-Eulerian Method for Incompressible Flows with Sharp Interfaces," *J. Comp. Phys.* **17**, 311 (1974). See also Hirt, Amsden, and Cook, *J. Comp. Phys.* **14**, 227 (1974).
- Chandrasekhar, S., *Hydrodynamic and Hydromagnetic Stability* (Clarendon Press, 1961), p. 428.

- Christiansen, J.P., "Numerical Simulation of Hydrodynamics by the Method of Point Vortices," *J. Comp. Phys.* **13**, 363 (1973).
- Cooley, J.W. and Tukey, J.W., "An Algorithm for the Machine Computation of Complex Fourier Series," *Math. Comp.* **19**, 297 (1975).
- Crout, P.D., "A Short Method for Evaluating Determinants and Solving Systems of Linear Equations with Real or Complex Coefficients," *Trans. AIEE* **60**, 1235 (1941).
- Crowley, W.P., "Flag, A Free-Lagrange Method for Numerically Simulating Hydrodynamics Flows in Two Dimensions," in *Proc. Second International Conference on Numerical Methods in Fluid Dynamics* (Springer-Verlag, 1971).
- Crowley, W.P., "A Global Numerical Ocean Model: Part 1," *J. Comp. Phys.* **3**, 111 (1968).
- Dietrich, D., B.E. McDonald, and A. Warn-Varnas, "Optimized Block-Implicit Relaxation," *J. Comp. Phys.* **18**, 421 (1975).
- Forester, C.K., "Higher Order Monotonic Convective Difference Schemes," *J. Comp. Phys.* **23**, 1 (1977).
- Fornberg, B., "On a Fourier Method for the Integration of Hyperbolic Equations," *SIAM J. Num. Anal.* **12**, 509 (1975).
- Fritts, M.J., "A Numerical Study of Free-Surface Waves," SAI Report SAI-76-528-WA, March 1976.
- Fritts, M.J., "Lagrangian Simulations of the Kelvin-Helmholtz Instability," SAI-76-632-WA (1976a).
- Fritts, M.J. and J.P. Boris, "Solution of Transient Problems in Free-Surface Hydrodynamics," NRL Memo. Rept. 3446 (1977); *J. Comp. Phys.* (to be published).
- Garrett, C.J.R. and W. Munk, "Space-Time Scales of Internal Waves," *Geophys. Fluid Dyn.* **2**, 225 (1972).
- Gottlieb, D., "Strang-Type Difference Schemes for Multidimensional Problems," *SIAM J. Num. Anal.* **9**, 650 (1972).
- Gottlieb, D. and Orszag, S.A., *Numerical Analysis of Spectral Methods: Theory and Applications* (SIAM, Philadelphia, 1977).
- Hain, K., "The Partial Donor Cell Method," NRL Memo. Rept. 3713 (1978).
- Harlow, F.H., "The Particle-in-Cell Computing Method for Fluid Dynamics," *Methods in Computational Physics*, Alder, Fernbach, and Rotenberg, eds. (Academic Press, New York, 1964), Vol. 3, p. 319.

- Harten, A., "The Method of Artificial Compression," CIMS Rept. COO-3077-50 (1974).
- Hirota, I., T. Tokioka, and M. Nishiguchi, "A Direct Solution of Poisson's Equation by Generalized Sweep-out Method," J. Meteor. Soc. Japan **48**, 161 (1970).
- Hockney, R.W., "A Fast Direct Solution of Poisson's Equation Using Fourier Analysis," Comm. Assoc. Comp. Mach. **12**, 95 (1965).
- Hockney, R.W., SUIPR Rept. No. 202, Inst. for Plasma Research, Stanford Univ., Stanford, Calif. (1966).
- Hockney, R.W., "The Potential Calculation and Some Applications," *Methods in Computational Physics*, B. Alden, S. Fernback, and M. Rotenberg, eds. (Academic Press, New York, 1970), Vol. 9.
- Jones, W.W. and J.P. Boris, "Flame and Reactive Jet Studies Using a Self-Consistent Two-Dimensional Hydrocode," J. Phys. Chem. **81**, 2532 (1977).
- Kershaw, D., "The Incomplete-Cholesky-Conjugate-Gradient Method for the Iterative Solution of Systems of Linear Equations," J. Comp. Phys **26**, 43 (1978).
- Kreiss, H.O., "A Comparison of Numerical Methods Used in Atmospheric and Oceanographic Applications," in *Proc. of the Symposium on Numerical Models of Ocean Circulation*, National Academy of Sciences (U.S.G.P.O., Washington, D.C., 1972). See also in the same volume B. Wendroff, "Problems of Accuracy with Conventional Finite-Difference Methods," and G. Fix, "A Survey of Numerical Methods for Selected Problems in Continuum Mechanics."
- Kreiss, H.O. and J. Oliger, "Comparison of Accurate Methods for the Integration of Hyperbolic Equations," *Tellus* **24**, 199 (1972).
- Kulikovskiy, A.G. and G.A. Lyubimov, *Magnetohydrodynamics* (Addison-Wesley, Palo Alto, Calif., 1965).
- Lanczos, C., *Applied Analysis* (Prentice-Hall, Englewood Cliffs, N.J., 1956).
- Launder, B.E. and D.B. Spalding, *Mathematical Models of Turbulence* (Academic Press, London, 1972).
- Lax, P.D., "Periodic Solutions of the KdV Equation," Comm. Pure and Appl. Math. **28**, 141 (1975).
- Liewer, P.C. and J.P. Boris, "Numerical Modeling of the Linus Flying Cusp," Bull. Am. Phys. Soc. **19**, 483 (1974).
- Lindzen, R.S. and H.-L. Kuo, "A Reliable Method for the Numerical Integration of a Large Class of Ordinary and Partial Differential Equations," Mon. Weath. Rev. **97**, 732 (1969).

NRL MEMORANDUM REPORT 4095

- McAvaney, B.J. and L.M. Leslie, "Comments on a Direct Solution of Poisson's Equation by Generalized Sweep-out Method," *J. Meteor. Soc. Japan* **50**, 136 (1972).
- McDonald, B.E., S.L. Ossakow, S.T. Zalesak, and N.J. Zabusky, "A Fluid Model for Estimating Minimum Scale Sizes in Ionospheric Plasma Cloud Striations," *NRL Memo. Rept.* 3864 (1978).
- McDonald, B.E., "Explicit Chebychev-Iterative Solution of Non-Self-Adjoint Elliptic Equations on a Vector Computer," *NRL Memo. Rept.* 3541 (1977).
- McDonald, B.E., T.P. Coffey, S.L. Ossakow, and R.N. Sudan, "Numerical Studies of Type 2 Equatorial Electrojet Irregularity Development," *Radio Science* **10**, 247 (1975).
- Madala, R.V., "An Efficient Direct Solver for Separable and Non-Separable Elliptic Equations," *Mon. Weath. Rev.* **106**, 1735 (1978).
- Madala, R.V. and S.A. Piacsek, "A Semi-Implicit Numerical Model for Baroclinic Ocean," *J. Comp. Phys.* **23**, 167 (1977).
- Marder, B.M., "GAP—A PIC-Type Fluid Code," *Math. Comp.* **29**, 434 (1975).
- Miura, R.M., "The Korteweg de Vries Equation: A Model Equation for Nonlinear Dispersive Waves," in *Nonlinear Waves*, S. Leibowitz and A.R. Seebass, eds. (Cornell Univ. Press, 1974).
- Moretti, G. and M. Abbott, "A Time-Dependent Computational Method for Blunt Body Flows," *AIAA J.* **4**, No. 12, 1966.
- O'Brien, J.J. and H.F. Hurburt, "A Numerical Model of Coastal Upwelling," *J. Phys. Ocean.* **2**, 14 (1972).
- Orszag, S.A., "Spectral Methods in Complex Geometries," *Fluid Dynamics Seminar*, Johns Hopkins Univ. Applied Physics Lab., Nov. 7, 1978.
- Orszag, S.A., "Spectral Methods for Problems in Complex Geometries," *Fluid Dynamics Seminar*, Applied Physics Lab., Johns Hopkins Univ., Nov. 14, 1978.
- Orszag, S.A. and M. Israel, "Numerical Flow Simulation by Spectral Methods," in *Proc. of the Symposium on Numerical Models of Ocean Circulation*, National Academy of Science (U.S.G.P.O., Washington, D.C., 1972).
- Ott, E., W.M. Manheimer, D.L. Book, and J.P. Boris, "Model Equations for Mode Coupling Saturation in Unstable Plasmas," *Phys. Fluids* **16**, 855 (1973).
- Richtmyer, R.D. and K.W. Morton, *Difference Methods for Initial Value Problems* (Interscience, J. Wiley and Sons, New York, 1967).
- Roache, P.J., "A New Direct Method for the Discretized Poisson Equation," in *Proceedings of the Second International Conference on Numerical Methods in Fluid Dynamics*, University of California, Berkeley, September 15-19, 1970 (Springer-Verlag, New York, 1971).

- Roache, P.J., "Marching Methods for Elliptic Problems: Part I," submitted to Numerical Heat Transfer (1977).
- Rosmond, T.E., and F.D. Faulkner, "Direct Solution of Elliptic Equations by Block Cyclic Reduction and Factorization," *Mon. Weath. Rev.* **104**, 641 (1976).
- Scannapieco, A.J., S.L. Ossakow, S.R. Goldman, and J.M. Pierre, "Plasma Cloud Late Time Striation Spectra," *J. Geophys. Res.* **81**, 6037 (1976).
- Scott, A.C., F.Y.F. Chu, and D. McLaughlin, "The Soliton: A New Concept in Science," *Proc. IEEE*, **61**, 1443 (1973).
- Strang, G. and G. Fix, *An Analysis of the Finite Element Method* (Prentice-Hall, Englewood Cliffs, N.J., 1973).
- Taggart, K.A., R.L. Morse, R.L. McCrory, and R.N. Remund, "Two Dimensional Calculations of Asymmetric Laser Fusion Targets Using IRIS," *Bull. APS* **20**, 1378 (1975).
- Tappert, F., "Numerical Solutions of the KdV Equation and Its Generalizations by the Split-Step Fourier Method," *Lect. Appl. Math.* **15**, AMS, Providence, R.I. (1974).
- Van Leer, B., "Toward the Ultimate Conservative Difference Scheme," *J. Comp. Phys.* **14**, 361 (1974).
- Van Leer, B., "Toward the Ultimate Conservative Difference Scheme: III. Upstream Centered Finite-Difference Schemes for Ideal Conservative Flow," *J. Comp. Phys.* **23**, 263 (1977).
- Van Leer, B., "Toward the Ultimate Conservative Difference Scheme: IV. A New Approach to Numerical Convection," *J. Comp. Phys.* **23**, 276 (1977a).
- Winsor, N.K. and J.M. Pierce, "Vectorizable Sparse Matrix Solution Methods, *Bull. APS* **23**, 898 (1978).
- Wurtele, M.G., "On the Problem of Truncation Error," *Tellus* **13**, 379 (1961).
- Zabusky, N.J., "A Synergetic Approach to Problems of Nonlinear Dispersive Wave Propagation and Interaction," *Proc. Symp. on Nonlinear Partial Differential Equations* (Academic Press, 1967).
- Zabusky, N.J. and M.D. Kruskal, "Interaction of 'Solitons' in a Collisionless Plasma and the Recurrence of Initial States," *Phys. Rev. Lett.* **15**, 240 (1965).
- Zalesak, S.T., "Fully Multidimensional Flux-Corrected Transport," *NRL Memo. Rept.* 3716 (1978).