NASA Grant NSG-1529

A SPLIT FINITE ELEMENT ALGORITHM FOR THE COMPRESSIBLE NAVIER-STOKES EQUATIONS

FINAL TECHNICAL REPORT

PREPARED FOR

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

LANGLEY RESEARCH CENTER
HAMPTON, VIRGINIA 23665

BY

UNIVERSITY OF TENNESSEE

DEPARTMENT OF ENGINEERING SCIENCE AND MECHANICS
KNOXVILLE, TENNESSEE 37916

Dr. Allen J. Baker
Principal Investigator

July 18, 1979
Date

SUMMARY

An accurate and efficient numerical solution algorithm is established for solution of the high Reynolds number limit of the Navier-Stokes equations governing the multi-dimensional flow of a compressible essentially inviscid fluid. The theoretical basis employs finite element interpolation theory within a dissipative formulation established using Galerkin criteria within the Method of Weighted Residuals (MWR). An implicit iterative solution algorithm is developed, employing tensor product bases within a fractional steps integration procedure, that significantly enhances solution economy concurrent with sharply reduced computer hardware demands. The algorithm is evaluated for resolution of steep field gradients and coarse grid accuracy using both linear and quadratic tensor product interpolation bases. Numerical solutions for linear and non-linear, one-, two- and three-dimensional examples confirm and extend the linearized theoretical analyses, and results are compared to competitive finite differenced-derived algorithms.

INTRODUCTION

Finite element concepts burst upon the computational fluid dynamics scene about a decade ago in the guise of a triangle. The primary motivation was the profuse geometric flexibility, in contrast to the then-current finite difference limitation to regular grids. In the ensuing interval, the inherent versatility of the basic finite element concept has proven difficult to master within a computationally economical framework. In the same period, development of regularizing coordinate transformation (ref. 1, 2) has markedly extended the applicability of efficient finite difference recursion formulae to non-regular shaped solution domain closures.

What was patently obscure in the early work, but is becoming convincingly transparent, is that the finite element/weighted residuals theoretical basis provides a foundation for derivation of optimally-accurate (in the appropriate norm) numerical algorithms for solution of general categories in fluid mechanics. The theoretical support for finite element solution of linear elliptic equations is complete, and in particular one is assured that a finite element potential flow solution is optimally accurate in the L_2 (energy) norm in comparison to all other methods (ref. 3). Solution of initial-valued problem descriptions is quite typical in fluid mechanics, and in the L_2 norm the finite element algorithm is confirmed optimally accurate for linear parabolic equations (ref. 4). The numerical extension to non-linear parabolic equations, as appropriate for boundary layer flows, has confirmed extremization of the energy norm for both laminar (ref. 5) and turbulent flows (ref. 6). The latter is of particular interest, since the energy norm is a strongly non-linear function of the mean flow field gradients through the effective turbulent viscosity. While by no means constituting a theoretical proof, these results to indicate that the linear finite element theory may be extensible to the more interesting problem classes.

A particularly difficult problem class in fluid dynamics corresponds to high Reynolds number flow prediction. The governing Navier-Stokes equations are generally a non-linear elliptic boundary value description, but the importance of the viscosity term is completely dwarfed by the non-linear convective acceleration everywhere away from a wall. Furthermore, the continuity equation possesses no viscosity-like term, hence exhibits uniformly the hyperbolic description pervading the entire flowfield region away from walls. The primary objective of the research reported herein is to derive and numerically evaluate a finite element/weighted residuals solution algorithm applicable to large Reynolds number flow prediction. It was decided that the algorithm must be implicit so as to handle physical viscosity effects as appropriate. However, primary emphasis rests on determination of accuracy and convergence phenomena for dominantly inviscid forms of the Navier-Stokes equations, ie. the Euler equations. Since three-dimensional flow field prediction is the eventual goal, the developed algorithm must be computer core and CPU efficient. Furthermore, since multi-dimensional fluid mechanics predictions are of necessity almost universally performed on coarse computational grids, particular emphasis is placed on coarse-grid, accuracy assessment. The accurate and efficient finite element tensor-product algorithm that has been derived to meet these requirements is reported herein.

PROBLEM STATEMENT

The basic requirement is to establish a numerical solution algorithm to accurately and efficiently model the substantial time derivative associated with all flow field descriptions (save potential flow). This hyperbolic operator dominates high Reynolds number flows, as governed by the Navier-Stokes equations.

The conservation form of this familiar partial differential equation system is

$$L(\rho) = \frac{\partial \rho}{\partial t} + \frac{\partial}{\partial x_j}(\rho u_j) = 0 \tag{1}$$

$$L(\rho u_i) = \frac{\partial(\rho u_i)}{\partial t} + \frac{\partial}{\partial x_j}\left[\rho u_j u_i + p\delta_{ij} - \sigma_{ij}\right] = 0 \tag{2}$$

$$L(\rho e) = \frac{\partial(\rho e)}{\partial t} + \frac{\partial}{\partial x_j}\left[\rho u_j\left(e + \frac{p}{\rho}\right) - \sigma_{ij}u_i + \theta_j\right] = 0 \tag{3}$$

In equations (1)-(3), ρ is the density, ρu_i the momentum vector, e is the specific energy, and p is the static pressure defined by the equation of state,

$$p = (\gamma - 1)\left[e - \frac{1}{2}\rho u_i u_i\right] \tag{4}$$

where γ is the ratio of specific heats. For present purposes, the heat flux vector θ_j and the Stokes stress tensor σ_{ij} are both assumed negligible; hence, equations (1)-(3) are hyperbolic. Equations (2), $1 \le i \le 3$, are explicitly non-linear, while equations (1) and (3) are quasi-linear in the expressed dependent variable, and each require characterization.

Equations (1)-(3) describe the transient evolution of the element q_k of $\{q\}$, $1 \le k \le 5$, on the n-dimensional space R^n spanned by the x_i coordinate system, $1 \le i \le n$. The domain of the solution is $\Omega \equiv R^n \times t \in x_i \times [(t_o, t)$ with closure $\partial\Omega \equiv R^{n-1} \times t$. On Ω, each member of $\{q\}$ is the solution to

$$L(q) = \frac{\partial q}{\partial t} + \frac{\partial}{\partial x_j}\left[u_j q + f\right] = 0 \tag{5}$$

where $f(q_k)$ is specified in equations (2)-(3). All applicable boundary conditions on $\partial\Omega$ are contained within the expression

$$\ell(q) = a_1 q + a_2 \frac{\partial q}{\partial x_\ell}\hat{n}_\ell + a_3 = 0 \tag{6}$$

where $\hat{n}_j$ is the local outward-pointing unit normal vector and the a_i are specified coefficients. An initial condition on $\Omega_o \equiv R^n \times t_o$ is required, hence

$$q(x_i, t_o) = q_o(x_i) \tag{7}$$

Since regularizing coordinate transformations are available, no generality is lost in assuming x_i a Cartesian coordinate system spanning R^n.

FINITE ELEMENT SOLUTION ALGORITHM

A dissipative finite element solution algorithm is established for equations (5)-(7), hence the Navier-Stokes and Euler equations. Assuming $U\Omega_e = \Omega$, the domain of $L(q)$, that the Ω_e are non-overlapping, and that $\Omega_e \equiv R_e^n \times t$, where UR_e^n is the finite element discretization of R^n, let each member of $\{q\}$ be interpolated on Ω_e as

$$q_e(x_i, t) \equiv \{N_k(x_i)\}^T \{Q(t)\}_e \tag{8}$$

The elements of $\{N_k(x_i)\}$ are polynomials in x_i, complete to degree k, and form a cardinal basis (ref. 7). The expansion coefficients $\{Q(t)\}_e$ are unknown, a solution algorithm for which is required established to determine the temporal evolution of the dependent variable system $\{q\}$. To accomplish this, substitute equation (8) into equations (5)-(6), and set to zero the integral of each over R_e^n and ∂R_e^n after weighting by $\{N_k(x_i)\}$. In addition, as suggested in reference 8, set the weighted integral of the vector gradient of equation (5) to zero. Identifying the vector and scalar multipliers β_i and λ, which must be determined, combine these expressions into the matrix equation system.

$$S_e\left[\int_{R_e^n}\{N\}L(q_e)d\tau - \beta_i\int_{R_e^n}\{N\}\frac{\partial}{\partial x_i}L(q_e)d\tau + \lambda\int_{\partial R_e^n\bigcap\partial R^n}\{N\}\ell(q_e)d\sigma\right] \equiv \{0\} \qquad (9)$$

Equations (9) are systems of ordinary differential equations written on the temporal evolution of the discrete approximation $\underset{e}{\Sigma}\{Q\}_e$ to each $q_e(x_i, t)$ of $\{q\}$. These differential equations are uncoupled in the temporal derivatives, and S_e is the familiar finite element assembly operator mapping local operations to the global reference frame (cf., ref. 4). The scalar multiplier λ is conventionally employed to enforce the discretized boundary condition statement, equation (6).

The β_i are scalar components of an n-dimensional vector, the determination of which is required. This term represents the additional requirement that the gradient of the solution error in $L(q_e)$ be orthogonal to the interpolation basis $\{N_k\}$. The desired form for the augmented MWR algorithm is achieved using a Green-Gauss theorem. Letting $\beta_i \equiv \nu_i\Delta_e$, where Δ_e is the measure of the finite element domain R_e^n, yields

$$S_e\left[\int_{R_e^n}\left[\left(1 + \Delta_e\frac{\partial}{\partial x_j}\nu_j\right)\{N\}\right]L(q_e)d\tau + \lambda\int_{\partial R_e^n\bigcap\partial R^n}\{N\}\ell(q_e)d\sigma\right] \equiv \{0\} \qquad (10)$$

The n scalar components of ν_j can be element-dependent in the general case, and the closure surface integral stemming from use of the Green-Gauss theorem vanishes identically. Independent of the dimension n of R^n, equation (10) yields an ordinary differential equation system for solution of $\underset{e}{\Sigma}\{Q(t)\}_e \equiv \{Q\}$, of the form

$$S_e\left[[C]_e\{\dot{Q}\}_e + [U]_e\{Q\}_e + \{f(Q)\}_e\right] = \{0\} \qquad (11)$$

The superscript prime denotes (ordinary) differentiation with respect to time. The first two terms, common for all q_k, see equations (5)-(6), are

$$[C]_e \equiv \int_{R_e^n} \left[\left(1 + \Delta_e \frac{\partial}{\partial x_i} v_i \right) \{N_k\} \right] \{N_k\}^T d\tau \tag{12}$$

$$[U]_e \equiv \int_{R_e^n} \left[\left(1 + \Delta_e \frac{\partial}{\partial x_i} v_i \right) \{N_k\} \right] \frac{\partial}{\partial x_j} \left[\{U_j\}_e^T \{N_k\} \{N_k\}^T \right] d\tau \tag{13}$$

Here, $\{U_i\}$ is the nodal distribution of the discrete representation of the velocity field $u_i(x,t)$. The source term $\{f(q)\}_e$ is distinct for each q_k, and represents a non-homogeneity at the minimum.

As mentioned, an implicit integration algorithm for equation (11) is required. A familiar single-step procedure is

$$\{Q\}_{j+1} = \{Q\}_j + h\left[\theta\{Q\}'_{j+1} + (1 - \theta)\{Q\}'_j\right] \tag{14}$$

where j is the time-step index, h is the integration stepsize, and θ is a parameter $0 < \theta \leq 1$ controlling implicitness. Following the usual manipulations (ref. 4), insertion of equation (11) into (14) yields a large order, nonlinear algebraic equation system. The Newton matrix iteration algorithm for solution of this system is

$$\left[J\left(\{Q\}_{j+1}^p\right)\right] \{\delta Q\}_{j+1}^{p+1} = -\left\{\left[F\left(\{Q\}_{j+1}^p\right)\right]\right\} \tag{15}$$

The dependent variable is the iteration vector,

$$\{Q\}_{j+1}^{p+1} \equiv \{Q\}_{j+1}^p + \{\delta Q\}_{j+1}^{p+1} \tag{16}$$

where p is the iteration index. The right side of equation (15) is the homogeneous form of equation (14) evaluated with the $p^{\underline{th}}$ iterate.

$$\{F\}^p_{j+1} = S_e\left[[C]_e\left(\{Q\}^p_{j+1} - \{Q\}_j\right) + h\left(\theta\{g_e\}^p_{j+1} + (1 - \theta)\{g_e\}_j\right)\right] \tag{17}$$

where

$$\{g_e\}^p_\ell \equiv [U]_e\{Q\}^p_\ell + \{f\}_e \tag{18}$$

Note that equations (17)-(18) are defined solely in terms of inner products on elements, with the assembly operator yielding the equivalent global contribution. The vanishing of {F} to within definition of a computed zero yields equation (15) homogeneous, hence convergence of the iteration process. By definition, the Jacobian is the derivative of equation (17) with respect to $\{Q\}^p_{j+1}$. Hence,

$$[J] = S_e\left[[C]_e + h\theta[U]_e + h\theta\,\frac{\partial\left([U]_e + \{f\}_e\right)}{\partial\{Q\}_e}\,\{Q\}_{j-\ell}\right] \tag{19}$$

where the final term accounts for contributions stemming from explicit nonlinearity. All operations involve matrix inner products of an elemental basis, hence implicitly independent of the dimension n of R^n. The rank of [J] at least equals the order of {δQ}; specific (Dirichlet) boundary constraints are applied within the evaluation of {F}.

As opposed to the conventional use of multidimensional, finite element interpolation functions $\{N(x_i)\}$, the three-dimensional requirements demand a spatial factorization that permits replacement of the large, sparse-matrix Jacobian operations with elementary banded-matrix procedures. The theoretical operations are to replace the multi-dimensional interpolation bases with

tensor product bases (ref. 9), and to implement the method of fractional steps (ref. 10) to resolve solution operations onto scalar components parallel to coordinate axis. Hence, the interpolation basis in equation (8) becomes, for three-dimensional space

$$\{N_k(x_i)\} \Rightarrow \{N_k(x_\alpha)\} \otimes \{N_k(x_\beta)\} \otimes \{N_k(x_\gamma)\} \tag{20}$$

where $\otimes$ signifies the tensor product. The equivalent tensor matrix products are similarly expressed; for example, the matrix equivalent of the initial-value operator, equation (12) becomes

$$[C]_e \Rightarrow [C_1]_e \otimes [C_2]_e \otimes [C_3]_e \tag{21}$$

where

$$[C_\alpha]_e \equiv \int_{R_e^\alpha} \left[\left(1 + \Delta_e \frac{\partial}{\partial x_\alpha} v_\alpha\right)\{N_k(x_\alpha)\}\right]\{N_k(x_\alpha)\}^T dx_\alpha \tag{22}$$

The similar operations for the convection operator, equation (13) yields

$$[U_\alpha]_e \equiv \int_{R_e^\alpha} \left[\left[1 + \Delta_e \frac{\partial}{\partial x_\alpha} v_\alpha\right]\{N_k\}\right] \frac{\partial}{\partial x_\alpha}\left[\{U_\alpha\}_e^T\{N_k\}\{N_k\}^T\right] dx_\alpha \tag{23}$$

and α is not a tensor summation index in equations (22)-(23).

With the finite element matrix equivalents of the terms in equation (11) recast as tensor products, the method of fractional steps (ref. 10) is employed to establish the desired operations for the Newton matrix iteration algorithm, equation (15). It is elementary operation to evaluate the Jacobian as the tensor matrix product,

$$[J] \Rightarrow [J_1] \otimes [J_2] \otimes [J_3] \tag{24}$$

THE UNIVERSITY OF TENNESSEE
Department of Engineering Science and Mechanics
317 Perkins Hall
Knoxville, Tennessee 37916
(615) 974-2171

August 3, 1979

NASA Scientific and Technical Information Facility
PO Box 8757
Baltimore & Washington International Airport
Maryland 21240

REF: NASA Grant NSG-1529, Final Report

Persuant to requirements for NASA Grant NSG-1529, enclosed are two (2) copies of the final technical report entitled, "A Split Finite Element Algorithm For The Compressible Navier-Stokes Equations." This report constitutes satisfaction of all reporting requirements for the referenced grant.

A. J. Baker
Associate Professor
Principal Investigator

mlk

cc Mr. Robert E. Smith, Jr.
NASA LaRC, M/S 125

and each of the $[J_\alpha]$ exhibit the desired banded matrix structure. Specifically, for k = 1 in equation (20), $[J_\alpha]$ is tridiagonal, while for k = 2 it is dominantly pentadiagonal. Hence, the tensor matrix solution algorithm equation (15) becomes

$$\left[J_3(Q)^p_{j+1}\right] \otimes \left[J_2(\bar{\bar{Q}})^p_{j+1}\right] \otimes \left[J_1(\bar{Q})^p_{j+1}\right]\{\delta Q\}^{p+1}_{j+1} = - \left\{F_1(\bar{Q})^p_{j+1}\right\} \otimes \left\{F_2(\bar{\bar{Q}})^p_{j+1}\right\} \otimes \left[F_3(Q)^p_{j+1}\right] \qquad (25)$$

Here, $\bar{Q}$ and $\bar{\bar{Q}}$ represent intermediate iterates and δQ is interpreted as the iteration vector for each respective iterate.

THEORETICAL ANALYSIS

A von Neumann stability analysis (ref. 11) can quantize the formal order of accuracy of the develoepd algorithm for a linearized one-dimensional equation, hence predict an appropriate value for ν. Therefore, consider the x-momentum equation with constant advection velocity U_o, ie

$$L(u) = \frac{\partial u}{\partial t} + U_o \frac{\partial u}{\partial x} = 0 \qquad (26)$$

The analytical solution to equation (26) is the Fourier expansion

$$u(x,t) = V \exp\left[i\,\omega(x - U_o t)\right] \qquad (27)$$

where $i \equiv \sqrt{-1}$, $\omega = 2\pi/\lambda$ is the wave number where λ is wavelength and V is the initial velocity distribution u(x,0). This solution corresponds to the diffusion- and dispersion-free advection of the initial wave form parallel to the x^α axis with velocity U_o.

The Fourier analysis of the discrete algorithm is best accomplished using equation (11). It is readily facilitated only for specification of k = 1 in equation (20), wherein the assembly operation yields an elementary recursion relation form of the algorithm. The resultant expression for equation (22) is

$$[C_\alpha]_e = \frac{\Delta_e^\alpha}{6}\begin{bmatrix}2 & 1\\ 1 & 2\end{bmatrix} + \frac{\nu\Delta_e^\alpha}{2}\begin{bmatrix}-1 & -1\\ 1 & 1\end{bmatrix} \tag{28}$$

for ν a constant. For a uniform discretization, the assembly of the first term of equation (11) over the two elements sharing node j yields

$$S_e\left[[C_\alpha]_e\{Q\}_e\right] = \frac{\Delta^\alpha}{6}\left[(1+3\nu)\,Q_{j-1}' + 4Q_j' + (1-3\nu)Q_{j+1}'\right] \tag{29}$$

where Δ^α is the uniform measure of the discretization in the direction of x^α. Similarly, the convection term in equation (23) becomes

$$[U_\alpha]_e = \frac{1}{6}\{U^\alpha\}_e^T\begin{bmatrix}-\begin{Bmatrix}2\\1\end{Bmatrix} & \begin{Bmatrix}2\\1\end{Bmatrix}\\ -\begin{Bmatrix}1\\2\end{Bmatrix} & \begin{Bmatrix}1\\2\end{Bmatrix}\end{bmatrix} + \frac{\nu}{2}\{U^\alpha\}_e^T\begin{bmatrix}\begin{Bmatrix}1\\1\end{Bmatrix}-\begin{Bmatrix}1\\1\end{Bmatrix}\\ -\begin{Bmatrix}1\\1\end{Bmatrix} & \begin{Bmatrix}1\\1\end{Bmatrix}\end{bmatrix} \tag{30}$$

For the constant advection velocity, $\{U^\alpha\}_e = U_o\{1\}$, and

$$S_e\left[[U_\alpha]_e\{Q\}_e\right] = \frac{U_o}{2}\left[-(1+2\nu)Q_{j-1} + 4\nu Q_j + (1-2\nu)Q_{j+1}\right] \tag{31a}$$

$$= \frac{U_o}{2}\left[-Q_{j-1} + Q_{j+1}\right] + U_o\nu\left[-Q_{j-1} + 2Q_j - Q_{j+1}\right] \tag{31b}$$

The second form of equation (31) emphasizes the action of the dissipative algorithm in introduction of a viscosity-like term, ie. the difference operation

equivalent of a second derivative term in equation (26) except for the omission of a power of the measure Δ^{α}. Alternatively, the first form suggests the action of $\nu \neq 0$ as a sort of upwind difference operation. It is important to note, however, that these elementary interpretation are valid only for the selected case, ie. U_0 a constant and Δ_e^{α} uniform.

The von Neumann stability analysis assumes the solution for the semi-discrete equivalent of the continuous Fourier form as

$$u^*(\Delta x, t) \equiv \sum_{\varepsilon} u_e = V \exp\left[i\,\omega(j\Delta x - \lambda t)\right] \tag{32}$$

where $\Delta x \equiv \Delta_e^{\alpha}$, the uniform discretization of R^{α}, j is the node indicator, and $\lambda \equiv \beta + i\delta$, where β and δ are real numbers. Comparing equations (27) and (32), a difference between β and U_0 constitutes a disparity in the phase speed of propagation of V, hence phase error in the discrete solution. Correspondingly, $\delta \neq 0$ introduces a real exponential argument yielding a damping (or growth) of the amplitude of the initial distribution V. Direct substitution of equations (29) and (31) into (32) and expanding the resultant expressions for β and δ in a Taylor series yields (ref. 12)

$$\beta = U_0\left[1 + \left(-\frac{1}{180} + \frac{\nu^2}{12}\right)d^4 + O(d^6)\right] \tag{33}$$

$$\delta = U_0\left[-\frac{\nu d^3}{12} + O(d^5)\right] \tag{34}$$

Here, $d \equiv \omega\Delta x$ and $O(\ )$ indicates the order of the truncated term. Since β is the real component of λ, it can be made identical to U_0 to order $(\Delta x)^6$ by requiring $\nu^{-1} \equiv \sqrt{15}$. Then, the phase accuracy of the discrete solution u*

agrees with U_o to sixth order accuracy in Δx, i.e., $O(\Delta x)^6$. For $\nu > 0$, $\delta < 0$ and an artificial damping is introduced. The resultant specific form for equation (32) is

$$u^*(\Delta x,t) = V \exp\left[i\,\omega\left[j\Delta x - \left[U_o + O(\Delta x^6)\right]t\right]\right]\exp\left[-\,\omega^4 kt + O(\Delta x^5)\right] \qquad (35)$$

where $k \equiv U_o(\Delta x)^3/12\sqrt{15}$ is the damping coefficient. Note that the damping is quite selective, occuring only for sufficiently large wave numbers ω (small wave lengths) due to the ω^4 factor.

Two additional comments are warranted. Setting $\nu \equiv 0$ eliminates δ and β is a fourth order accurate representation of the differential equation. This high order accuracy accrues with use of the simplest linear interpolation. It is obvious that the convection term in equation (31b) is the central difference equivalent; the improvement to fourth order results directly from the finite element derived form for $[C_\alpha]_e$, ie. equation (29). The normalized (1,4,1) weighting on the derivatives corresponds identically with a spline interpolation. The conventional finite difference practice, eg. reference 13, is to replace equation (19) with $\Delta^\alpha Q_j$ which yields directly a degradation to overall second order accuracy. Secondly, the conventional finite difference practice to introduce dissipation is to add the "artificial viscosity" term $\mu\,\partial^2 u/\partial x^2$ to the parent differential equation (29). Repeating the semi-discrete Fourier analysis with this added term, and setting $\nu \equiv 0$ but retaining the finite element derived initial-value term, yields (ref. 12).

$$u^*(\Delta x,t) = V \exp\left[i\,\omega\left[j\Delta x - \left[U_o + O(\Delta x^4)\right]t\right]\right]\exp\left[-\,\omega^2\mu t + O(\Delta x^2)\right] \qquad (36)$$

The basic fourth order phase accuracy remains intact, but the artificial damping is less selective due to the diminution of the wave number exponent to 2 and the appearance of the term of order Δx^2.

The stability of the algorithm for the linearized equation can be assessed using a fully discrete Fourier analysis of equation (14) combined with (11). In this instance, letting n denote the time index, the approximate solution form is

$$u^{*n}(\Delta x,\Delta t) = g^n \exp\,[i\omega j\Delta x] \tag{37}$$

The form of the amplification matrix g^n is sought, since the discrete solution will propagate and damp/grow dependent upon its real and imaginary arguments. Previous analyses (ref. 14), for the non-dissipative algorithm, indicate that the trapezoidal rule ($\theta \equiv \frac{1}{2}$) is the sole suitable selection. Retaining this definition, the amplification factor for the dissipative finite element algorithm is

$$g = \frac{1+\frac{1}{2}\cos(\omega\Delta x) - 3C\nu\sin^2(\frac{1}{2}\omega\Delta x) - i\frac{3}{4}(C+2\nu)\sin(\omega\Delta x)}{1+\frac{1}{2}\cos(\omega\Delta x) + 3C\nu\sin^2(\frac{1}{2}\omega\Delta x) + i\frac{3}{4}(C-2\nu)\sin(\omega\Delta x)} \tag{38}$$

$C \equiv U_0\Delta x/\Delta t$ is the Courant Number, and the numerator and denominator of g are complex conjugates for $\nu \equiv 0$. Therefore, the basic non-dissipative algorithm is neutrally stable, ie. $|g| = 1$ for all Δx and Δt, hence error induced by the solution will propagate undamped and unmagnified throughout the solution domain. Selecting $\nu \neq 0$ destroys this neutral stability; therefore, define g in terms of real and imaginary parts as

$$g = \gamma + i\Gamma \tag{39}$$

Hence, $\gamma > 0$ quantizes the dissipative mechanism, while Γ determines the phase accuracy. In particular,

$$\theta = \frac{1}{\lambda} \tan^{-1}\left[\frac{i\Gamma}{\gamma}\right] \tag{40}$$

yields the normalized phase velocity of the approximate solution u^{*n}. Figure 1 graphs equation (40) for the non-dissipative form; the analytical solution corresponds to the horizontal line at $u^*/U_o = 1$. The solid curves represent the finite element algorithm, while the dashed curves correspond to equation (38) as modified by the finite difference diagonalized initial-value matrix form. The superior performance of the present theory is clearly evident. At a modest Courant number, eg. $0.01 < C < 0.5$, the algorithm accurately resolves all wavelengths $\lambda > 5\Delta x$, while for the latter this occurs only for $\lambda > 15\Delta x$.

The action of the added dissipative mechanism ($\nu \neq 0$) is to improve the phase accuracy in the short wave length region. Table 1 lists select evaluations of equation (38) for $C = 0.25$ for various ν and λ. The dissipation level μ is defined as

$$\mu \equiv \frac{-1}{\lambda^4} \ln|g| \tag{41}$$

It is evident that ν exerts a profound correction to θ on the interval $2 \leq \lambda \leq 5$, hence produces a closer approximation to the correct solution regarding phase accuracy. The penalty for improved phase accuracy is the corresponding introduction of dissipation. The dissipation level μ is modestly sensitive to λ and nearly linearly dependent on ν.

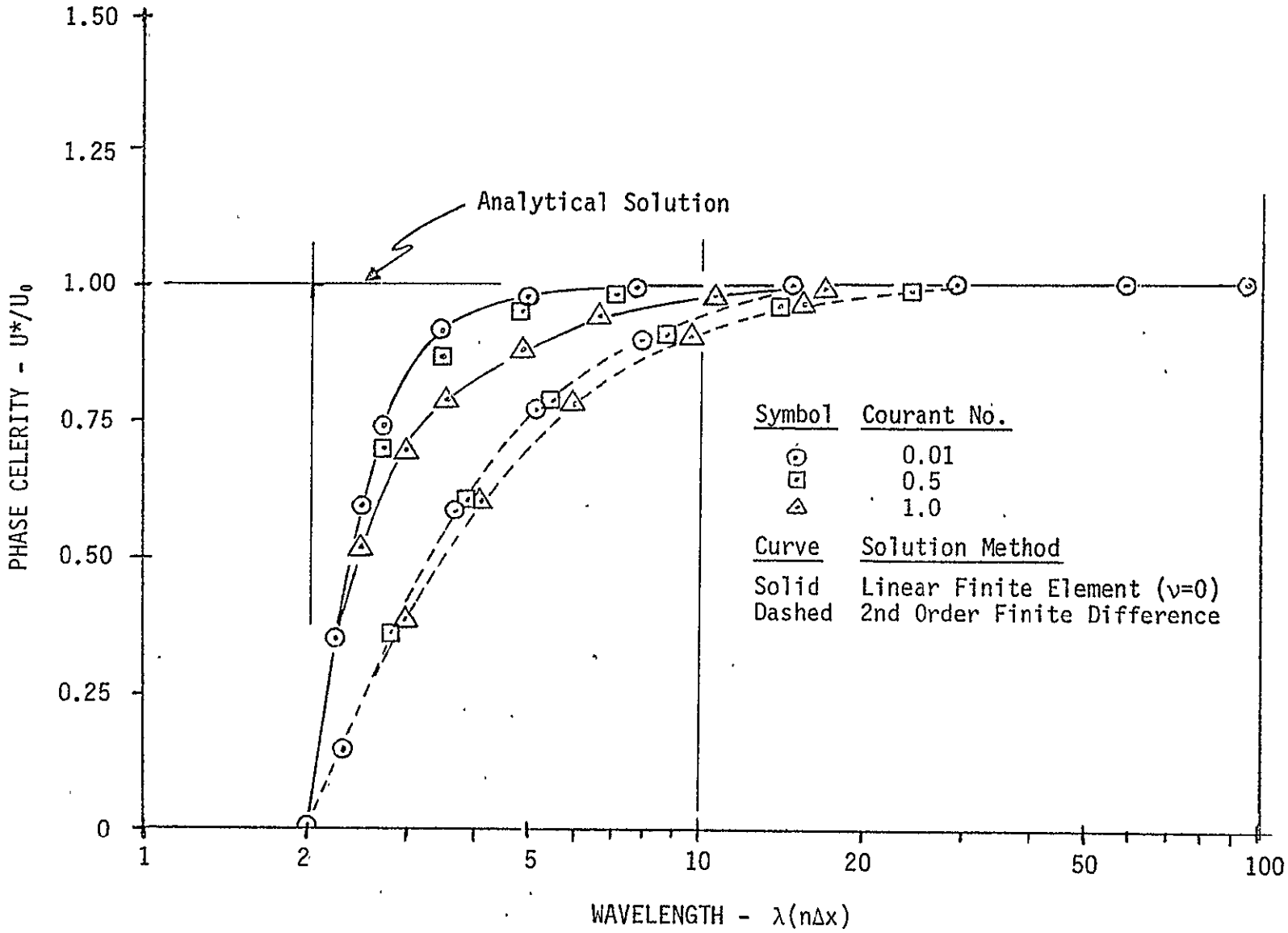


Fig. 1. Phase Error of Non-Dissipative Finite Element & Finite Difference Algorithms.

Table 1

Influence of Dissipation Level ν on Phase Celerity and Dissipation Level, C = 0.25

Wavelength $\lambda(n\Delta x)$	Phase Celerity = θ				Dissipation Level - μ			
	ν=0.	ν=0.1	ν=0.25	ν=0.5	ν=0	ν=0.1	ν=0.25	ν=0.5
2.00	0.00000	0.00000	0.00000	0.00001	0.0	0.01241	0.03438	0.07991
2.25	0.34486	0.38455	0.59418	1.26043	0.0	0.01615	0.03854	0.06108
2.50	0.58252	0.62651	0.82294	1.26599	0.0	0.01637	0.03611	0.04907
2.75	0.72707	0.76034	0.90723	1.20781	0.0	0.01547	0.03322	0.04358
3.00	0.81442	0.83843	0.94355	1.15717	0.0	0.01443	0.03093	0.04075
3.50	0.90399	0.91639	0.97178	1.09111	0.0	0.01275	0.02786	0.03839
4.00	0.94397	0.95081	0.98222	1.05509	0.0	0.01163	0.02601	0.03777
5.00	0.97560	0.97815	0.99037	1.02238	0.0	0.01036	0.02400	0.03799
6.00	0.98687	0.98803	0.99373	1.00994	0.0	0.00971	0.02298	0.03860
8.00	0.99455	0.99490	0.99664	1.00208	0.0	0.00909	0.02201	0.03962
10.00	0.99705	0.99719	0.99789	1.00018	0.0	0.00881	0.02158	0.04025
15.00	0.99891	0.99894	0.99908	0.99954	0.0	0.00854	0.02116	0.04099

Anticipating the results of numerical experiments, to be discussed, the phase-accurate optimum value of $\nu^{-1} = \sqrt{15}$ introduces entirely too much artificial diffusion, for both linear and non-linear example equation systems. In addition, these linearized analyses are valid only for the algorithm using linear interpolation, and performance assessment with at least quadratics is required. As an indication of expected performance, the quadratic interpolation basis yields formally fourth order accurate difference representation for both convection and diffusion differential operators at the elemental vertex nodes of a uniform discretization of x^{α}. The additional required assessment of the tensor product algorithm basis is also facilitated by numerical experiment.

NUMERICAL RESULTS

One-Dimensional Solutions

The primary requirement is to assess acceptable bounds on $\nu > 0$ that facilitate accurate solutions without introduction of excessive artificial diffusion. An appropriate example for examining the important non-linearity of the momentum equation (2) is the inviscid form of Burgers equation

$$L(u) = \frac{\partial u}{\partial t} + u \frac{\partial u}{\partial x} = 0 \tag{42}$$

For an initial condition corresponding to a square wave, see Figure 2a), the exact solution to equation (42) is propagation of the original wave form parallel to u with a celerity of $\frac{1}{2}u$. Figures 2b)-c) show the results obtained from the dissipative finite element algorithm for $k = 1$, $C = 0.125$ and $\nu^{-1} = \sqrt{30}$. The propagation speed of the wave is exactly correct, ie., the

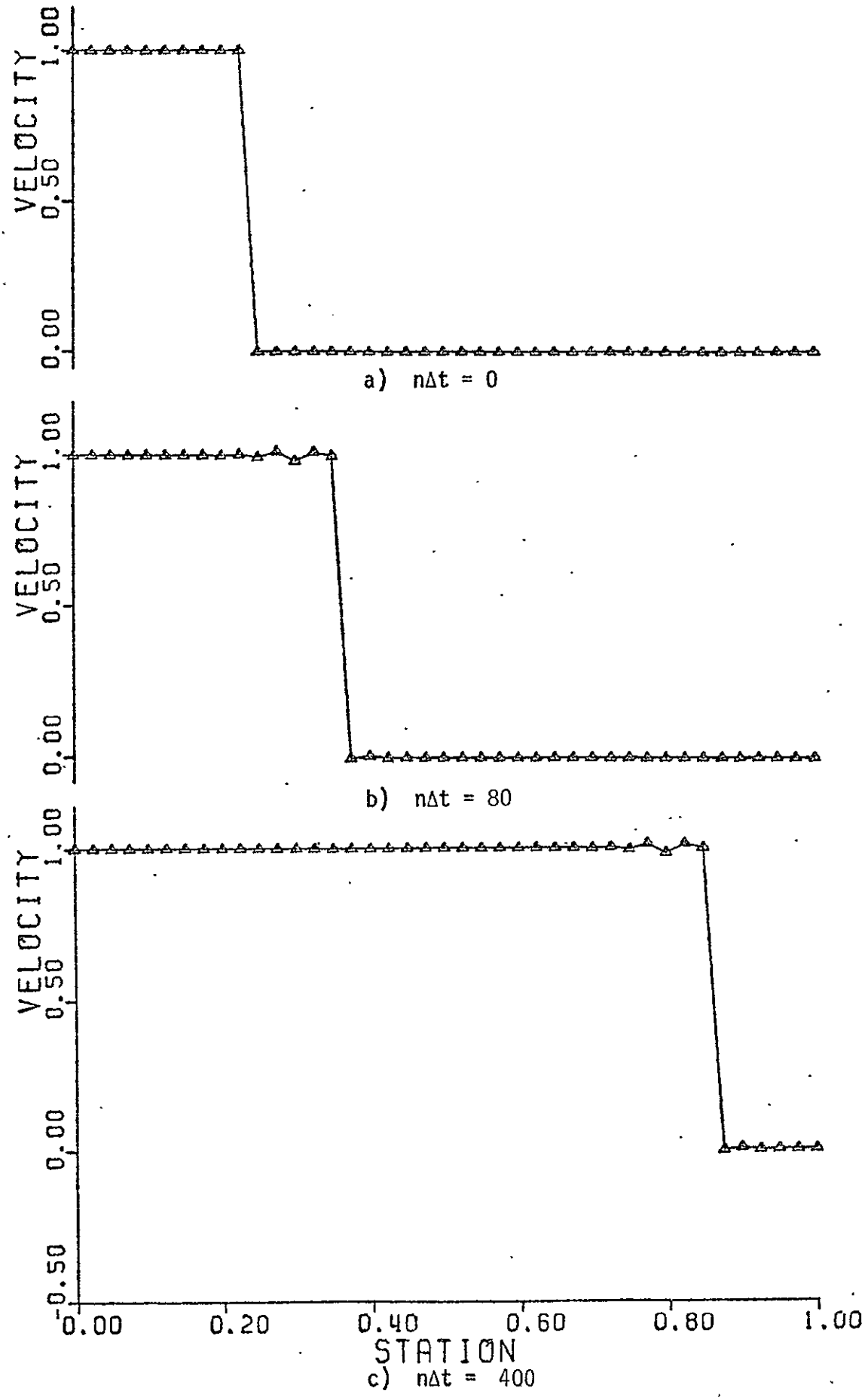
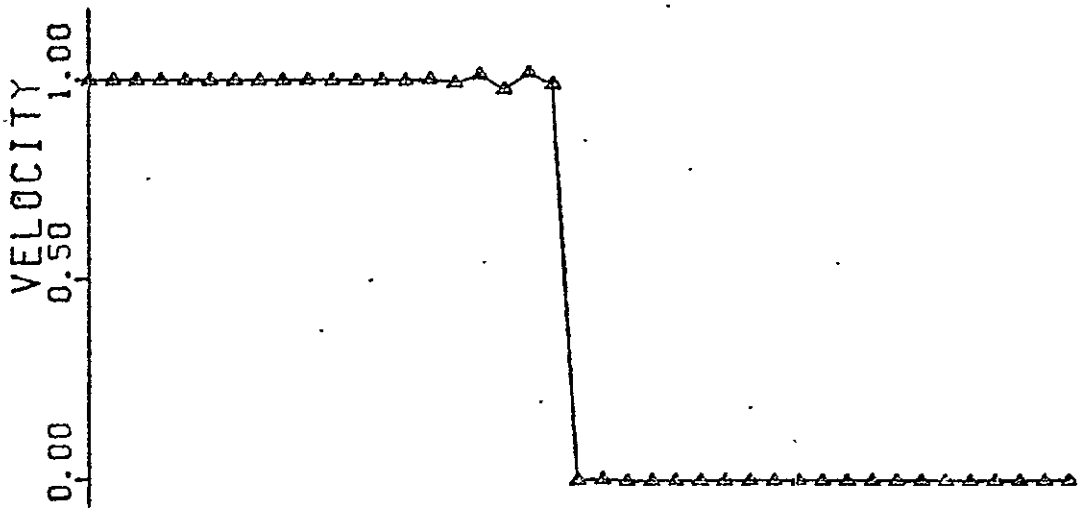


Fig. 2. Solution of 1-D Burger's Equation, C = 0.125, $\nu^{-1} = \sqrt{30}$ k = 1.

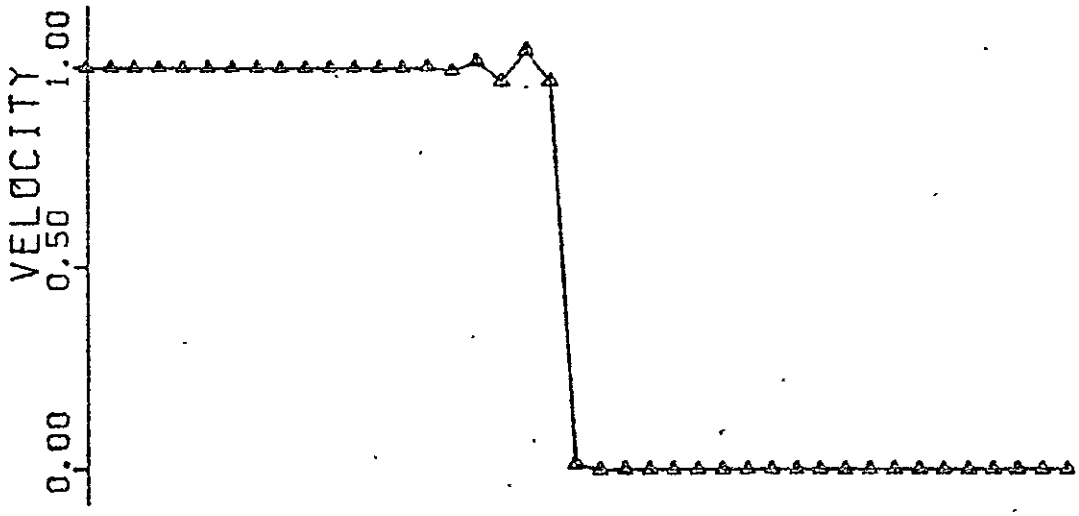
wave numerical distribution is identically repeated every 16 time steps, ie. 2/C. The solution approximation to the original square wave is excellent, ie., the step remains interpolated across one element domain only, the leading phase error is only 0.3%, while the lagging phase error extremum is 1.3%. For this value of ν, there is no perceptible diffusion of the step; however, diffusion is introduced when the "optimal" linear analysis value of $\nu^{-1} = \sqrt{15}$ is used, and the square wave becomes interpolated across three element domains.

Figure 3 summarizes the influence of Courant number (time intergration step size) and level of ν on the square wave solution. The results in Figure 3a) were obtained for C = 0.5 and $\nu^{-1} = \sqrt{30}$. The fidelity of the original square wave is excellently maintained with no evidence of numerical diffusion. The leading phase error is reduced to 0.1%, compare to Figure 2b), while the lagging error extremum is increased to 1.9%. The algorithm is stable to unit Courant Number; the fidelity of the original square wave is degraded further (lagging error extremum is 4.6%), as induced by the truncation error associated with the trapezoidal rule. However, the wave remains interpolated across only one finite element. For comparison, setting $\nu^{-1} = \sqrt{15}$ and C = 1.0 yields the results shown in Figure 3c). The lagging error peaks have become completely diffused, the leading error peak is 1.9%, and the wave has become interpolated over four element domains.

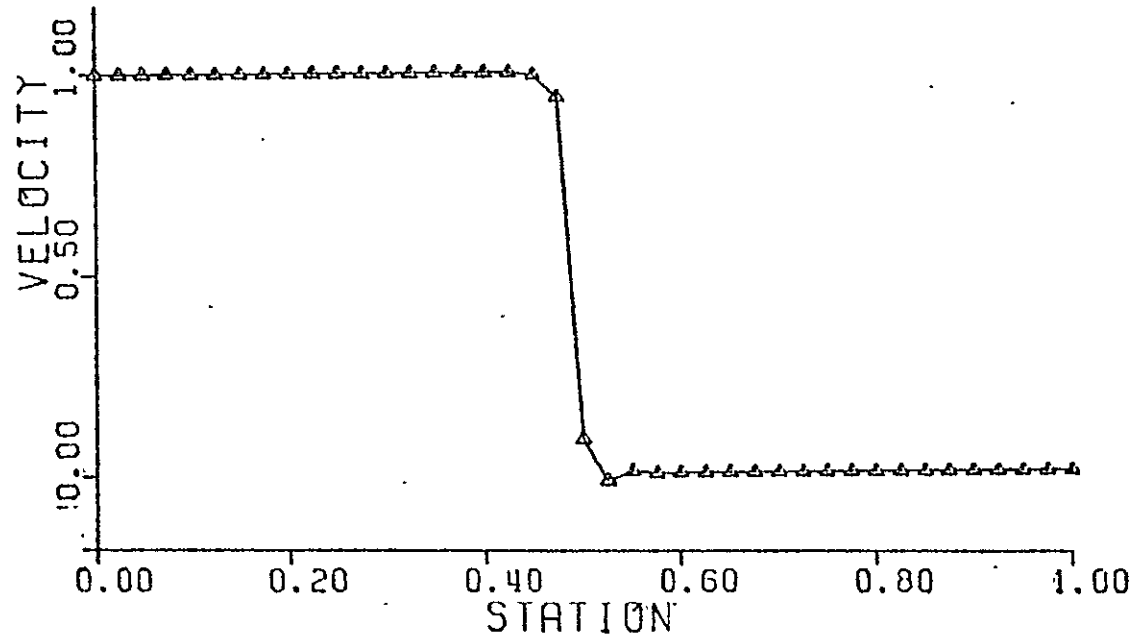
These results verify that the developed dissipative implicit finite element algorithm, employing linear interpolation bases, exhibits excellent accuracy control for the sample non-linear problem. Additional numerical tests have quantized the relative importance of the derived matrix structures. By and large, all modification degrade performance of the basic algorithm.



a) $\nu^{-1} = \sqrt{30}$, C = 0.5



b) $\nu^{-1} = \sqrt{30}$, C = 1.0

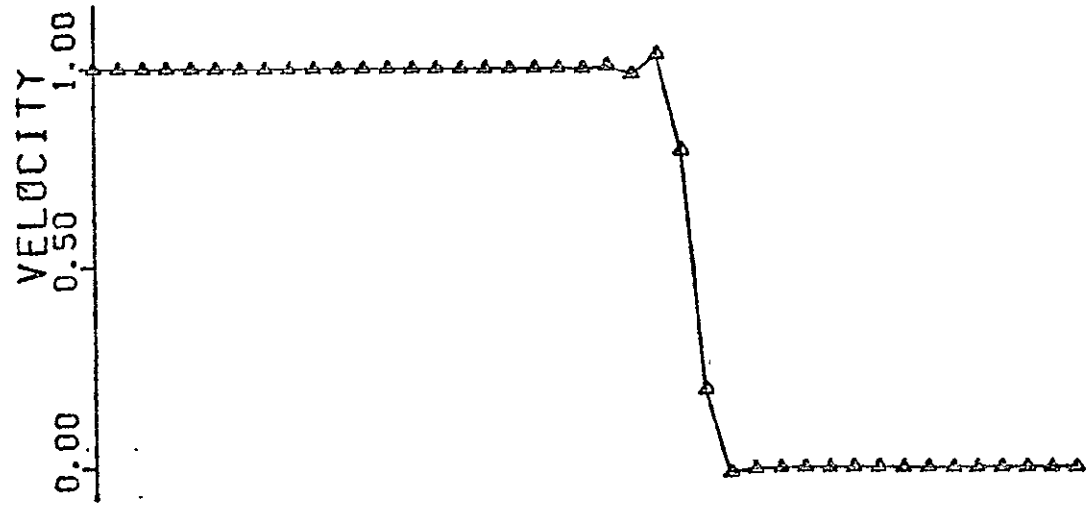


c) $\nu^{-1} = \sqrt{15}$, C = 1.0

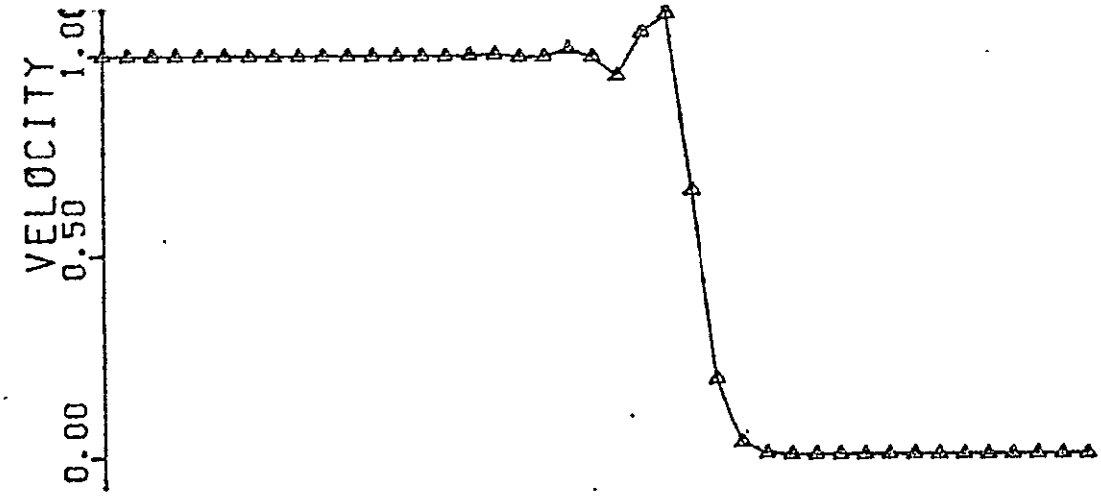
Fig. 3. Solution of 1-D Burger's Equation, Various Courant Numbers and Viscosity, Linear Elements, $n\Delta t = 100$

For example, the results in Figure 4a) were obtained following replacement of the diffusion term $\nu\{U^{\alpha}\}_e^T$ with the scalar $\nu\{1\}^T$ in equation (30). The results are clearly inferior to the comparison test, Figure 2. For Figure 4b), this modification was retained, and in addition, the finite element-derived initial value matrix structure, equation (49), was collapsed to the finite difference diagonal form. A large lagging dispersion error peak is introduced, and these results are definitely poorer than the comparison case, Figure 3a). Figure 4c) corresponds to the exact duplicate of the Beam and Warming (ref. 13) implicit finite difference algorithm with an added fourth-order dissipation term, ie., a fourth order accurate finite difference equivalent of a viscosity term. The value of μ, the artificial diffusion coefficient, was numerically optimized, and Figure 4c compares almost exactly with Figure 1c of reference 13. By comparison, these results are clearly much poorer than those of Figure 3. Hence, these tests firmly quantize the superior performance of the finite element-based algorithm for this test case.

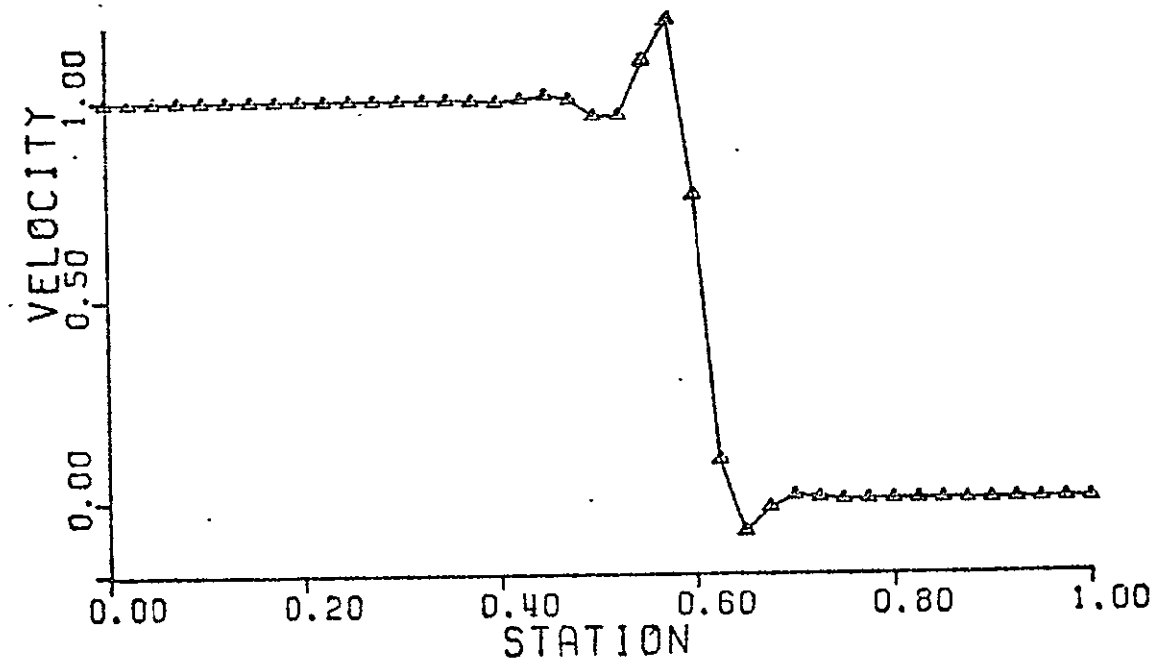
Similar results are obtained for the quadratic element embodiment of the algorithm, as obtained setting $k = 2$ in equation (20). Figure 5a) illustrates the square wave after 80 time steps at $C = 0.125$ and for $\nu^{-1} = \sqrt{15}$. Since the quadratic possesses a non-vertex node, the original wave (interpolated across one element) possesses a nodal mid-value. The quadratic algorithm maintains an adequately accurate representation of the original wave, see Figure 5a), with lagging and leading dispersion error extrema of $\pm$ 4%. The phase celerity is again exactly correct, as the numerical solution is repeared every 16 time steps. For Figure 5b), the dissipation term was scalarized, as discussed for the $k = 1$ solution in Figure 4b), which induces sufficient additional dissipation to eliminate the dispersion error and



a) k = 1; S = 0; $\nu^{-1} = \sqrt{30}\ \{1\}$; C = .125



b) k = 1; S = 2; $\nu^{-1} = \sqrt{30}\ \{1\}$; C = 0.5



c) k = 1; B & W + $4^{\underline{th}}$; μ = .5(-3); C = 0.5

Fig. 4. Solution of 1-D Burger's Equation, Various Assembly and Viscosity, Linear Elements.

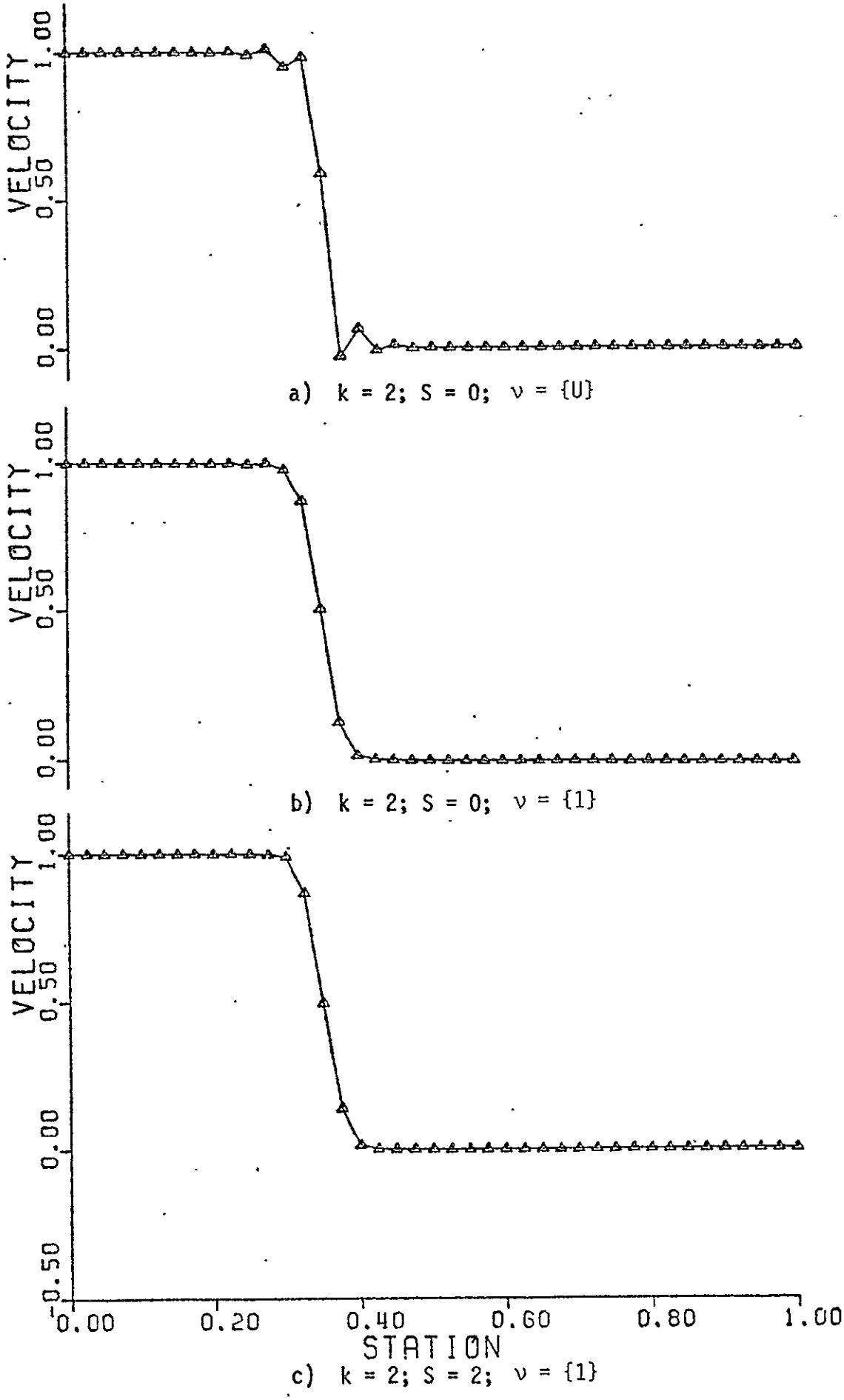
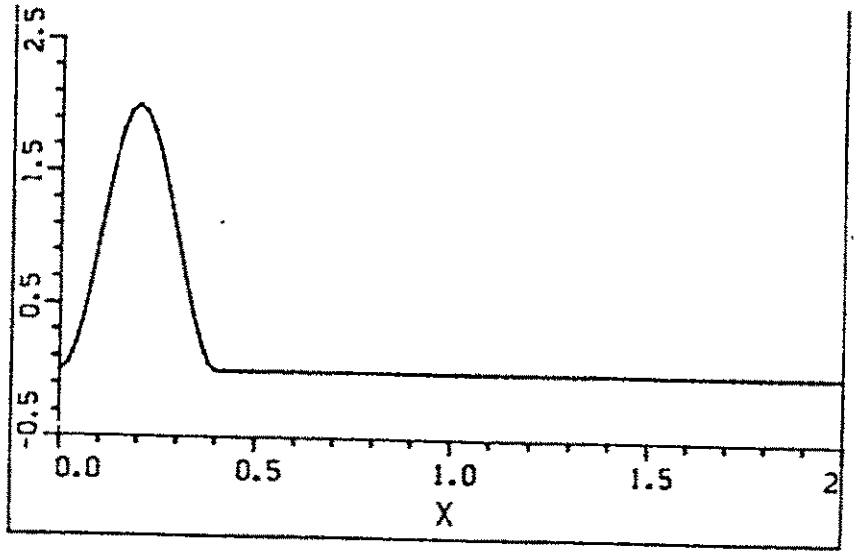


Fig. 5. Solution of 1-D Burger's Equation, Quadratic Elements, Various Assembly and Viscosity, $n\Delta t = 80$

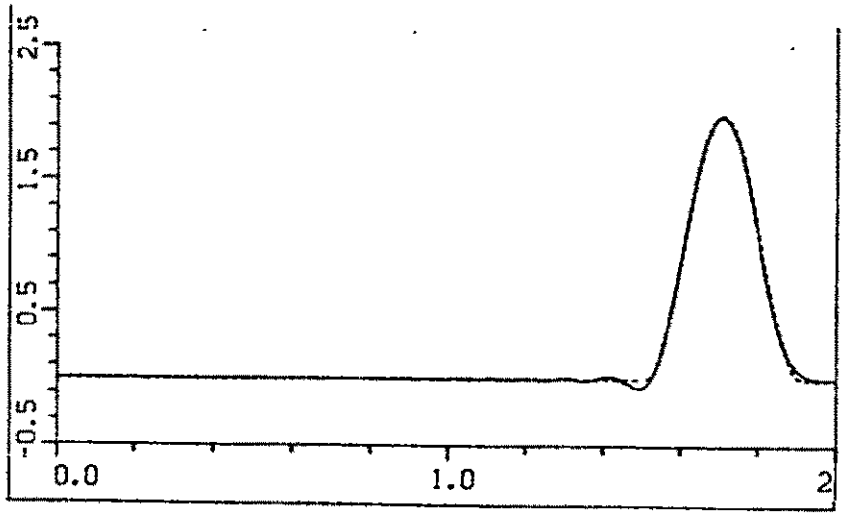
smear the wave over two element domains. The additional diagonalizing of the initial value matrix does not markedly alter this solution, Figure 5c, in contrast to the linear element solution results.

The influence of the dissipative mechanism within the derived algorithm is less demonstrative for a one-dimensional linear equation solution, eg. the continuity equation (1). A test case is advection of a cosine wave by a constant imposed velocity, for which the theoretical analysis is exact. Figure 6a) shows the initial-condition, and Fig. 6b) the non-dissipative linear ($\nu = 0$, $k = 1$) finite element solution following propagation over three wave-lengths. The dashed curve is the exact solution, and for $C = 0.4$ the finite element result agrees almost exactly. A modest leading phase error is exhibited, a trailing $2\text{-}3\Delta x$ wave is induced by the relatively poorer phase accuracy, and the peak value remains at 100%. The diagonal initial-value matrix form yields exactly the Crank-Nicolson algorithm. Figure 6c) shows the corresponding results, which are substantially poorer in comparison. As predicted by the theoretical analysis, phase fidelity is much poorer, to the extent that the marginally diffused peak celerity becomes in substantial error. In contrast to the non-linear square wave test case, this solution if continued will produce totally erroneous trash.

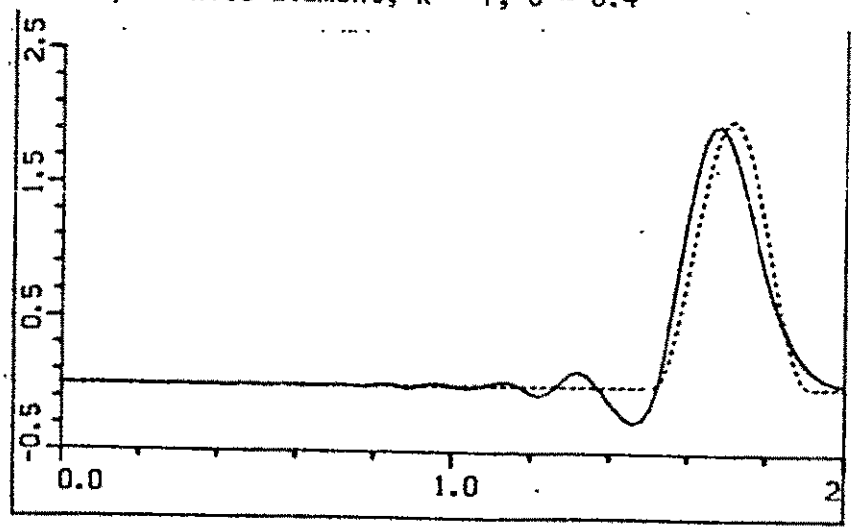
Table 2 summarizes the accuracy of the non-dissipative linear element algorithm solutions as a function of integration time-step, ie. Courant number. A peak value of 100 is retained up to $C = 0.5$, and solution dissymetry progressively increases with larger Courant number. The last column denotes the magnitude of the largest trailing wave peak, which always occurs immediately behind the cosine wave. (Not until $C = 1.4$ is the finite element solution distribution similar in appearance to the Crank-Nicolson result for $C = 0.4$.) Table 3 summarizes the influence of the dissipation factor level



a) Initial Condition, Dashed Line is Analytical Solution



b) Finite Element, k = 1, C = 0.4



c) Crank-Nicolson, C = 0.4

Fig. 6. Convection of a Cosine Wave, Linear Elements, $\nu = 0$.

Table 2

Influence of Courant Number on
Cosine Hill Distribution, $k = 1$, $\nu = 0$

Courant No.	Solution Distribution							Dispersion Error Peak
Analytical	15	50	85	100	85	50	15	0
0.1	16	50	85	100	84	47	15	-2
0.2	16	50	85	100	83	46	15	-2
0.4	17	52	87	100	81	45	15	-3
0.5	18	54	90	100	78	43	16	-5
0.6	20	56	90	98	75	42	16	-6
0.7	21	59	91	96	73	41	17	-7
0.8	23	61	92	94	70	40	17	-8

Table 3

Influence of Dissipation Level on Solution
For Cosine Hill, C = 0.4, k = 1

Dissipation Level (ν)	Solution Distribution							Dispersion Error Peak
Analytical	15	50	85	100	85	50	15	0
0.0	17	52	87	100	81	45	15	-3
0.1 E-03	18	52	85	98	80	45	17	-3
0.1 E-02	19	52	85	98	80	45	17	-3
0.1 E-01	19	52	84	97	79	45	17	-3
0.2 E-01	20	52	84	95	78	46	18	-2
0.5 E-01	21	52	81	92	76	46	20	-1
0.9 E-01	23	52	78	87	74	47	22	0
0.1291	25	51	75	84	72	47	23	0

ν on phase accuracy for the test case at C = 0.4. Levels of $\nu < 0.01$ do not measureably alter the solution. Increasing $\nu < 0.01$ tends to progressively symmetrize the solution while continuously adding diffusion of the peak level. At $\nu = 0.1291$, which is one-half the linear analysis optimum value, the solution is nearly symmetric, but the level of peak diffusion is unacceptably large. Therefore, it is confirmed that the sixth-order phase accuracy is unacceptable in terms of induced artificial diffusion.

The corresponding results for the quadratic element form of the algorithm are summarized in Tables 4 and 5. Up to C = 0.5, the non-dissipative algorithm produces essentially identical results using k = 1 and 2, compare Tables 2 and 4. For larger Courant number, the inaccuracy and phase shift of the quadratic form becomes progressively poorer in comparison. Comparing Tables 3 and 5, there is little performance difference between k = 1 and k = 2 for $\nu > 0$ at C = 0.4 for this test case. Definitive differences will result for multidimensional solutions, however.

Table 4

Influence of Courant Number on
Cosine Hill Distribution, $k = 2$, $\nu = 0$

Courant No.	Solution Distribution							Dispersion Error Peak
Analytical	15	50	85	100	85	50	15	0
0.1	14	49	85	98	87	50	14	-1
0.2	14	50	86	99	86	49	14	-1
0.3	15	52	87	99	84	47	13	-2
0.4	16	53	89	99	82	45	13	-3
0.5	19	58	92	98	77	41	14	-7
0.6	21	60	94	97	74	40	15	-9
0.7	26	66	96	91	67	38	16	-11
0.8	26	67	98	92	67	37	16	-13

Table 5

Influence of Dissipation Level on Solution
For Cosine Hill, C - 0.4, k = 2

Dissipation Level (ν)	Solution Distribution							Dispersion Error Peak
Analytical	15	50	85	100	85	50	15	0
0.0	16	53	89	99	82	45	13	-3
0.1 E-3	16	53	89	98	82	45	13	-3
0.1 E-2	17	53	88	98	82	45	14	-3
0.1 E-1	19	53	85	94	79	46	16	-2
0.2 E-1	21	53	82	90	77	46	19	-1
0.5 E-1	26	51	73	79	70	47	24	0
0.9 E-1	30	49	65	70	63	46	28	0
0.1291	31	47	59	63	58	45	30	0

Multi-Dimensional Solutions

The one-dimensional solutions have quantized acceptable levels of $\nu > 0$ for which adequate accuracy can be maintained within acceptable diffusion levels. The present requirement is embedding the tensor-product basis into a multi-dimensional algorithm and evaluate non-linear and linear solutions. Figure 7 illustrates, for two-dimensions, the manner in which fractional steps are employed for the grid sweeps associated with use of the tensor product basis. For linear interpolation, Figure 7a), the elements of the linearly independent cardinal basis are contracted on the first sweep (parallel to x_1 axis) with node numbers

$$\{1, 2; 2, 3; 3, 4; ..; 29, 30; 31, 32; 32, 33; ...\}$$

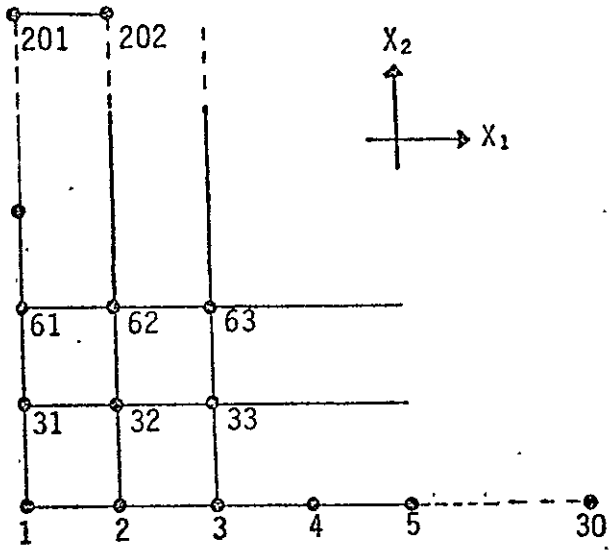
to form the vectors on the interpolation subspace. Within the fractional steps concept, the first (inner) tensor matrix statement in equation (25) is iterated to convergence of $\{\overline{Q}_{j+1}\}$. For the second sweep, parallel to the x_2 axis, the elemental contraction vectors are ordered

$$\{1, 31; 31, 61; ...; 200, 201; 2, 32; ...\}$$

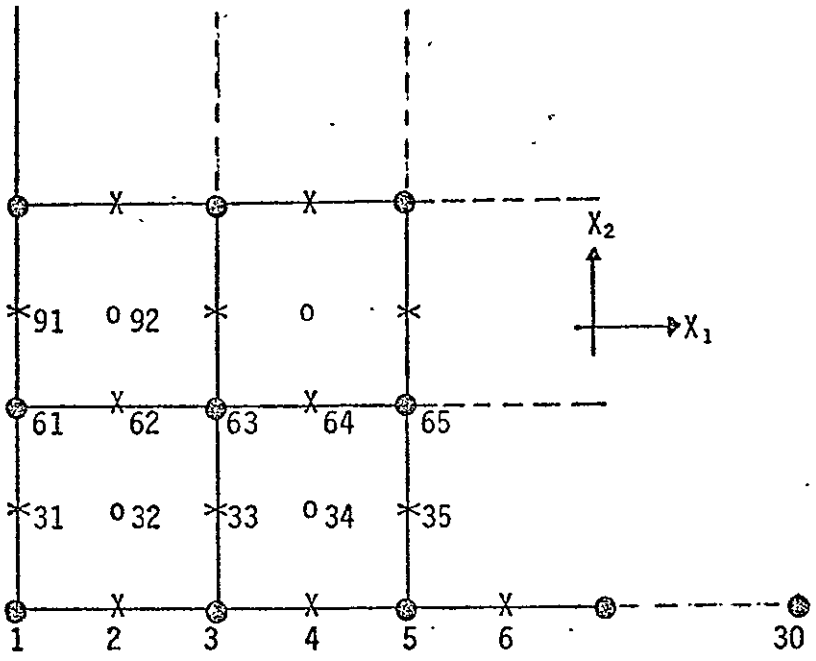
and the independent variables in the interpolation basis are linearly dependent upon x_2. The iteration of the interior matrix product then yields the converged value of $\{\overline{\overline{Q}}\}_{j+1}$. The extension to the third direction is obvious, whereupon the matrix iteration converges to $\{Q\}_{j+1}$, the solution at time-step t_{j+1}.

The linear element tensor product algorithm is rather comparable to a finite difference alternating direction framework. The quadratic basis algorithm is somewhat more complex. Referring to Figure 7b), for the first sweep the contraction nodal vector is

$$\{1, 2, 3; 3, 4, 5; ..; 28, 29, 30; 31, 32, 33; ...\}$$



a) Linear Basis



b) Quadradic Basis

Fig. 7. Illustrations for Tensor Product Basis Finite Element Algorithm

The even numbered nodes 2, 4, 6, .., 32, ... are non-vertex in the first sweep, which yields the solution $\{\overline{Q}\}_{j+1}$. For the second sweep parallel to x_2, the contraction vector becomes

{1, 31, 61; 61, 91, 121; ...; 2, 32, 62; ...}

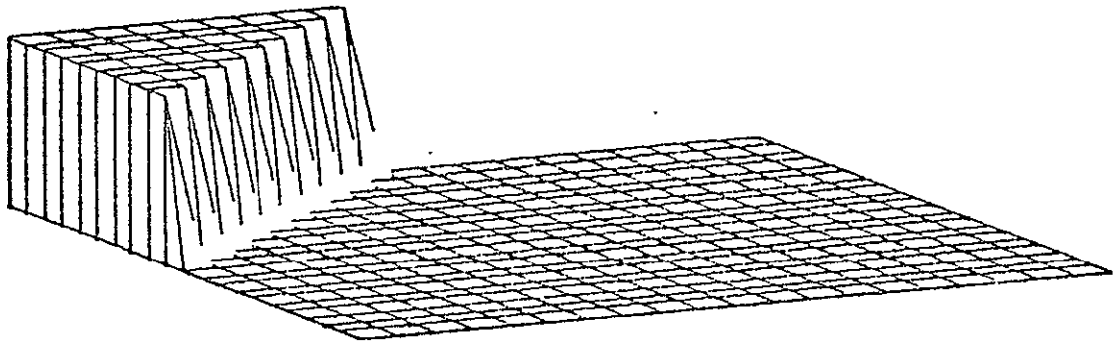
The odd-number nodes 31, 91, 33, etc. are now non-vertex, while 2, 62, 4, etc. have become vertex, and the solution is $\{\overline{Q}\}_{j+1}$. Note that only nodes 32, 34, 92, etc. are always non-vertex for a two-dimensional problem. Hence, the tensor product algorithm employs an interpolation basis essentially comparable to the non-Serendipity multi-dimensional quadratic with an interior (centroidal) node. The extension of the quadratic tensor product algorithm to three-dimensions is again direct. To avoid the one-sided accumulation of round-off error, the tensor matrix algorithm sweeps are sequenced with each coordinate direction cyclically alternating position within equation (25).

The comparable non-linear test case is the two-dimensional inviscid Burgers equation system

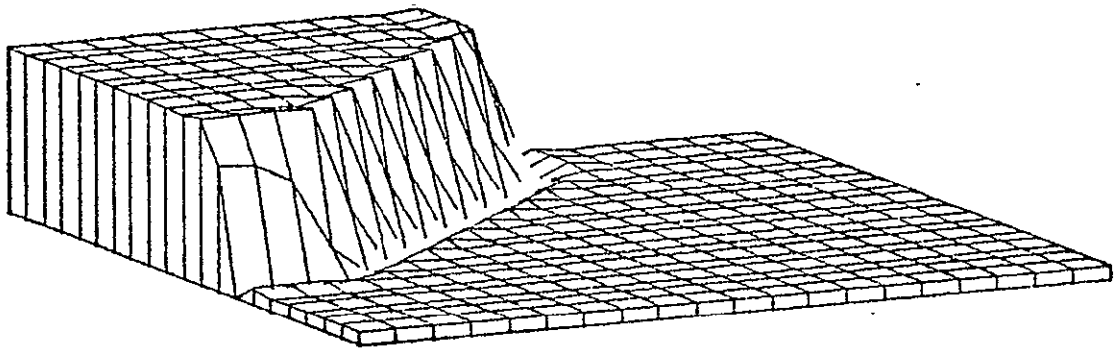
$$L(u) = \frac{\partial u}{\partial t} + u\,\frac{\partial u}{\partial x} + v\,\frac{\partial u}{\partial y} = 0$$

$$L(v) = \frac{\partial v}{\partial t} + u\,\frac{\partial v}{\partial x} + v\,\frac{\partial v}{\partial y} = 0 \qquad (43)$$

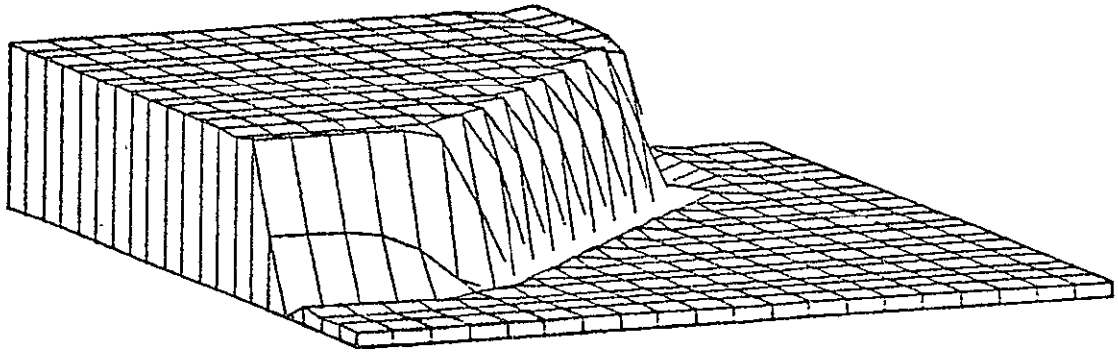
Figure 8a shows the initial condition for a square wave impinging on the upper left corner of the solution domain. The initial step distribution is interpolated across one element domain only; the correct solution is pure advection of the initial distribution, parallel to the domain diagonal with celerity $U_o/\sqrt{2}$, and maintenance of unit and zero level plateaus. Vanishing normal derivative boundary conditions were applied everywhere except for those nodes possessing non-zero initial values which were fixed. Figures 8b) -c) show the linear element computed results for C = 0.125 and $\nu^{-1} = \sqrt{30}$. They are excellent approximations to the correct solution away from the



a) Initial Condition, t = 0



b) Downstream, $n\Delta t = 60$



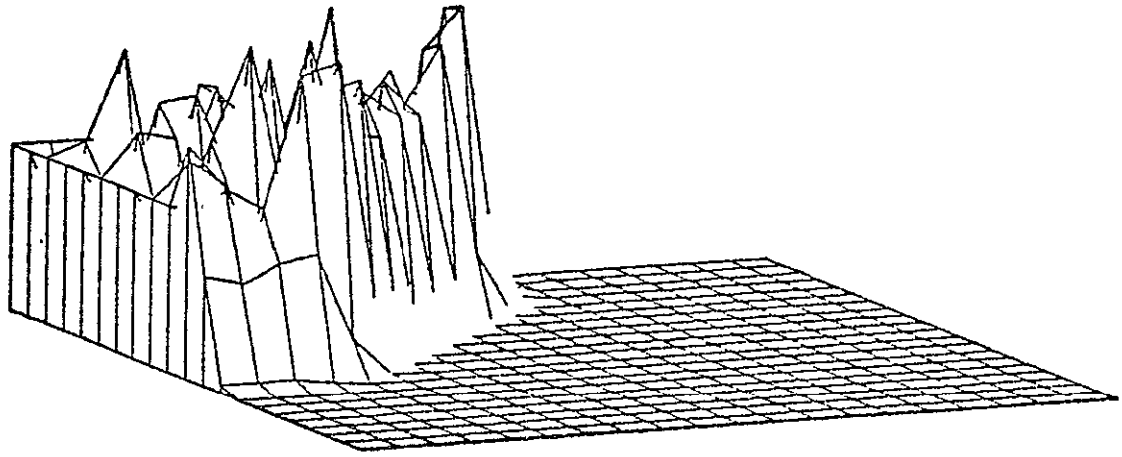
c) Downstream, $n\Delta t = 120$

Fig. 8. Solution of Two-Dimensional Burgers Equation, Linear Tensor Product Finite Element Algorithm, C = .125, $\nu^{-1} = \sqrt{30}$

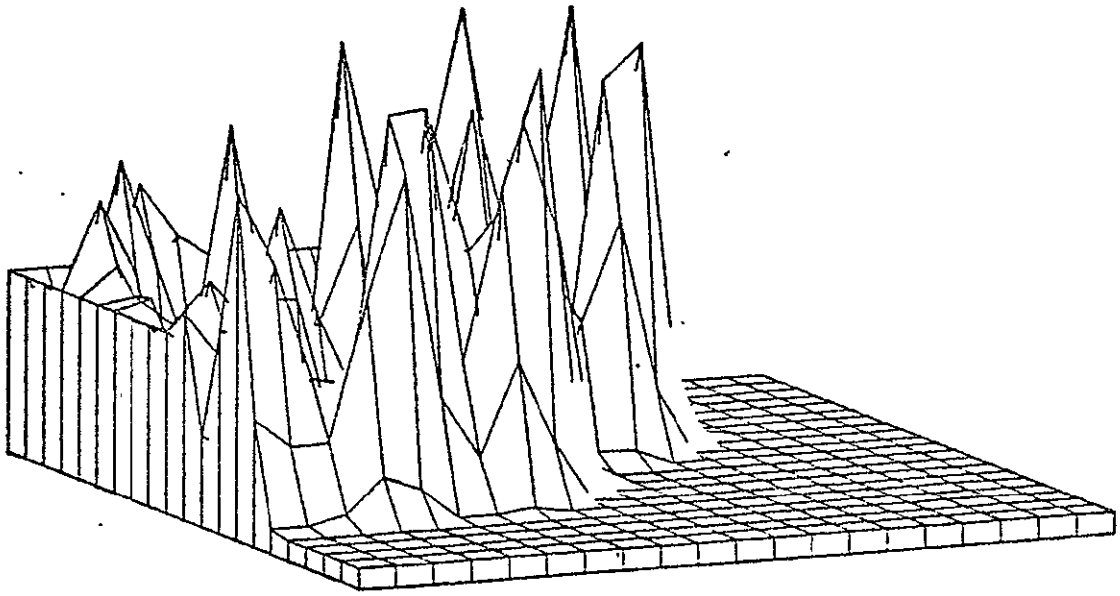
boundaries. Thereupon,the numerical solution generates an acceptable approximation to the vanishing normal derivative (which cannot be explicitly enforced, see equation (6), since $\ell(q) = 0$ is a homogeniety in equation (9)). The center wave celerity is exact, and the dispersion error-induced peaks are nominally $\pm$ 4% and are limited to the immediate vicinity of the front. For comparison, Figure 9 shows the identical test case but with $\nu \equiv 0$ within the algorithm. The importance of the dissipative mechanism is graphically evident.

As was done for the one-dimensional cases, comparison results are obtained with finite difference matrix structure modifications. The diagonalization of the initial-value matrix degrades the algorithm to second-order accuracy. The concomitant artificial diffusion is further enhanced by reducing the phase selectivity of the dissipation action by setting $\nu\{U\} \Rightarrow \nu\{1\}$. The resultant solution, Figure 10 has diffused the front over approximately six element domains. The wave front is considerably sharpened, see Figure 11, by restoring the theoretical matrix statement $\nu\{U\}$, with an associated increase (to 12%) of the lagging phase error peak. These multi-dimensional results appear quite comparable to the finite difference experience discussed in Figure 4. In all cases, the restructuring of the basic theoretical statement degrades solution performance.

Figure 12 shows the corresponding test case solutions generated by the quadratic tensor product algorithm for C = 0.125 and $\nu^{-1} = \sqrt{30}$. The solutions are basically identical to those obtained by the linear element form, see Figure 8, with only modest leading and lagging error peaks, excellent plateaus and good approximations to vanishing normal boundary gradients. In particular, the one-dimensional wave form in Figure 5a) is almost identical with the intersection of the two-dimensional solution with the plane y = 0, Figure 12b).

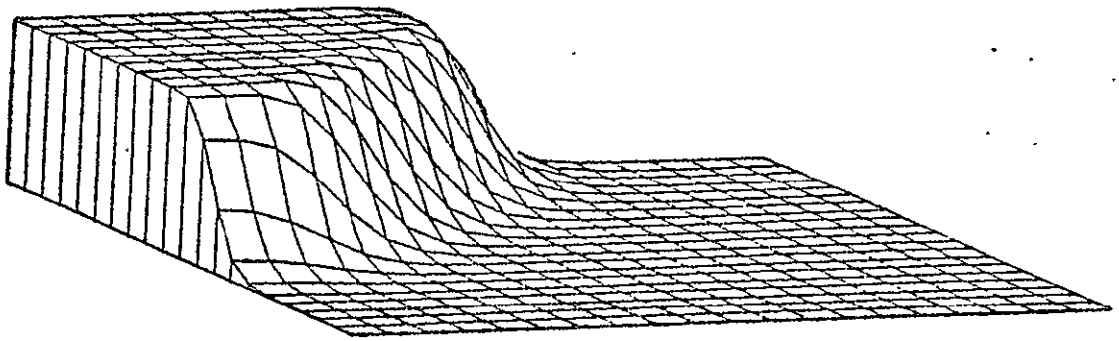


a) Downstream, $n\Delta t = 60$

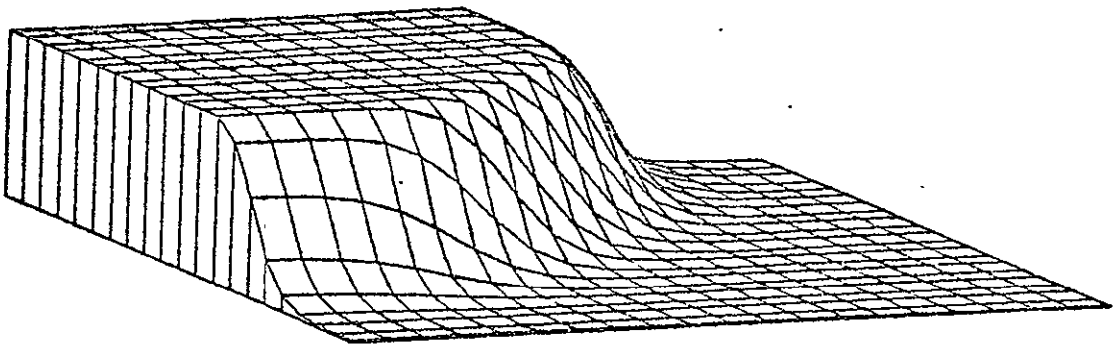


b) Downstream, $n\Delta t = 120$

Fig. 9. Solution of Two-Dimensional Burgers Equation, Linear Tensor Product Finite Element, $C = .125$, $\nu \equiv 0$

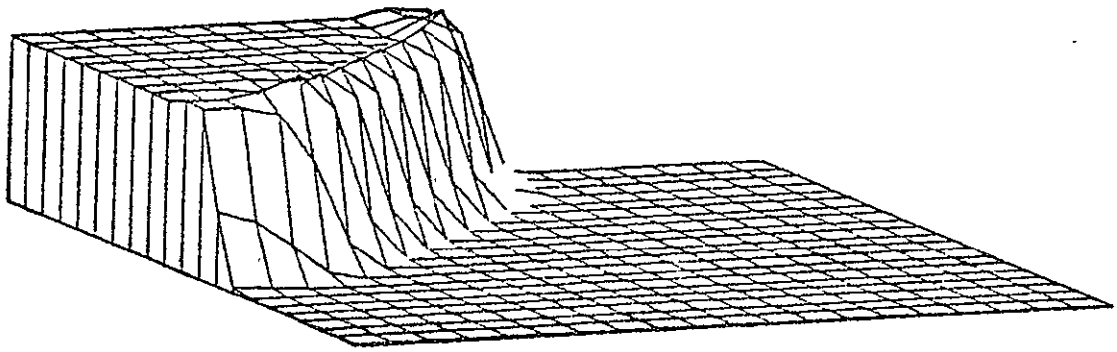


a) Downstream, $n\Delta t = 60$

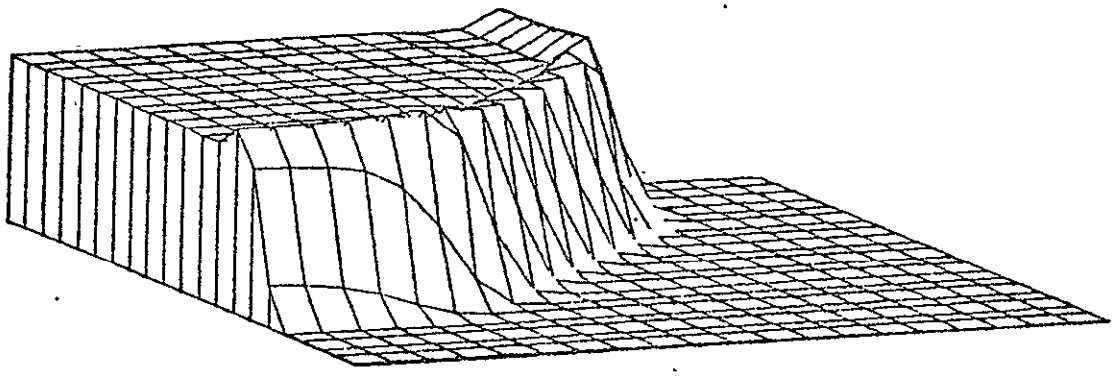


b) Downstream, $n\Delta t = 120$

Fig. 10. Solution of Two-Dimensional Burgers Equation, Diagonalized Linear Tensor Product Algorithm, C = 0.125, $\nu^{-1} = \sqrt{30}$ ({1}).



a) Downstream, $n\Delta t = 60$



b) Downstream, $n\Delta t = 120$

Fig. 11. Solution of Two-Dimensional Burgers Equation, Diagonalized Linear Tensor Product Algorithm, C = 0.125, $\nu^{-1} = \sqrt{30}$ ({U}).

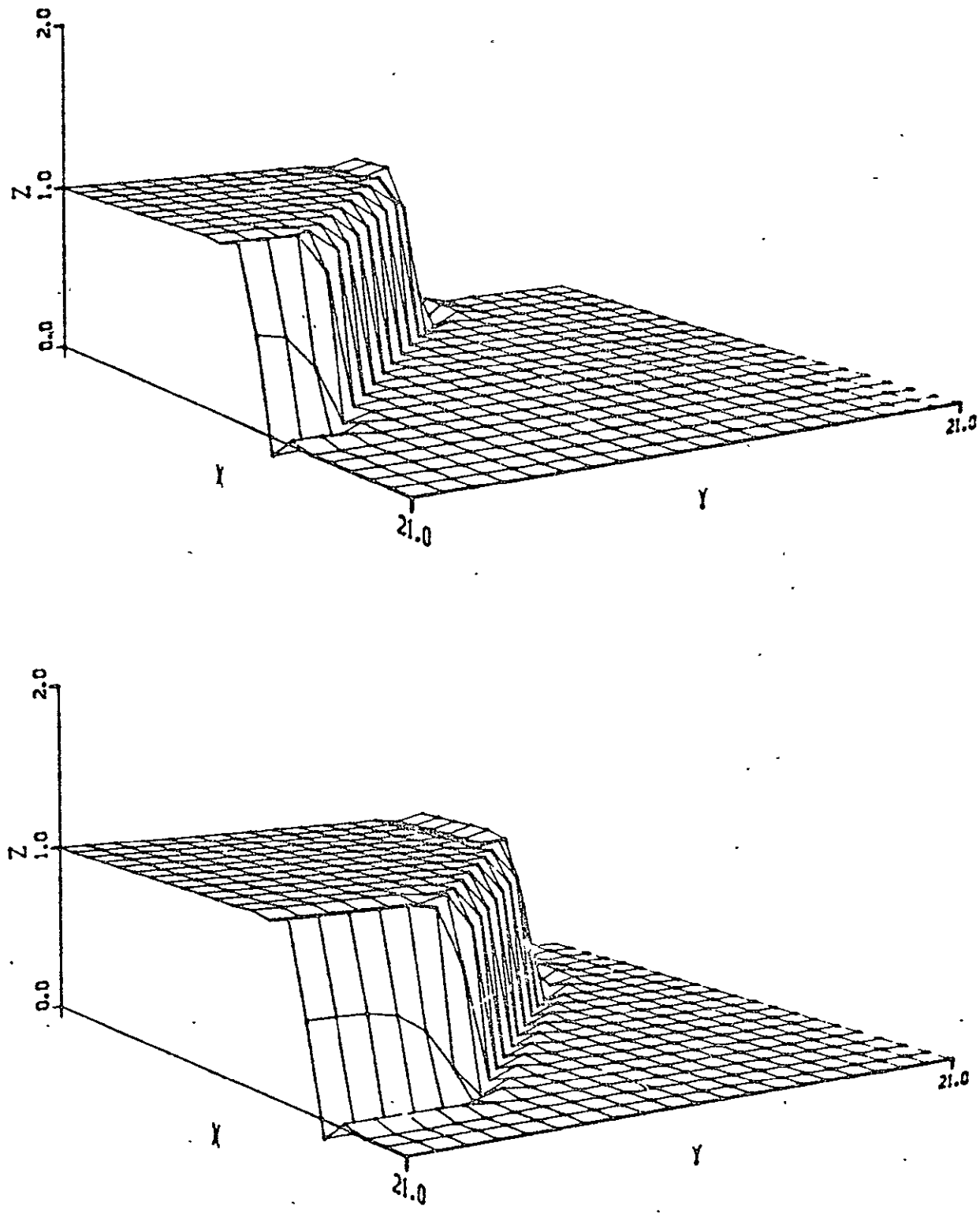


Fig. 12. Solution of Two-Dimensional Burgers Equation, Quadratic Tensor Product Algorithm, C = 0.125, $\nu^{-1} = \sqrt{30}$

Solution improvement with the quadratic tensor product basis occurs for the multi-dimensional solution of the linear equation. The two-dimensional test case equivalent is the "cosine hill", obtained by rotating the one-dimensional cosine wave about the peak. A demanding test is pure advection by an imposed solid body rotation, cf. reference 16. The correct solution is exact propagation of the initial distribution. Fibures 13a) -c) illustrate the linear element solution results for $C = 0.15$ and $\nu \equiv 0$. After one-full turn, the peak value would remain at 100 if the phase distortion did not move it off a node location, and the lagging phase error peak is 10%. Figure 14 quantizes the data; Figure 14a) shows the initial condition distribution, which is identical with the exact solution following one full revolution. Figure 14b) shows the $k = 1$ results obtained after one rotation at $C = 0.15$, and the circled value corresponds to the correct peak location. The generally lagging phase has retarded the computed peak about one-half cell, and the phase dispersion error is firmly quantized. Figure 15a) shows the comparison solution, obtained with the non-dissipative $k = 2$ algorithm form at $C = 0.15$, the accuracy of which is excellent. The solution distortion due to lagging phase is nominally absent, the lagging wake, peak is -2%, and the solution peak is undiffused (it is actually modestly increased) and occurs at the exact nodal location. Doubling the integration step size to $C = 0.3$ produces the solution shown in Fig. 15b), which is essentially comparable to the linear element solution at $C = 0.15$. Due to the pentadiagonal Jacobian, the quadratic algorithm is about 16% slower than the linear; therefore, the net CPU savings for the $k = 2$ solution at double the Courant Number is about 35%. The linear element solution at $C = 0.3$ further decreased the peak value to 82, from the 93 in Figure 14b), and the corresponding enhancement of dispersion error produced an unacceptable -17 in the trailing wake.

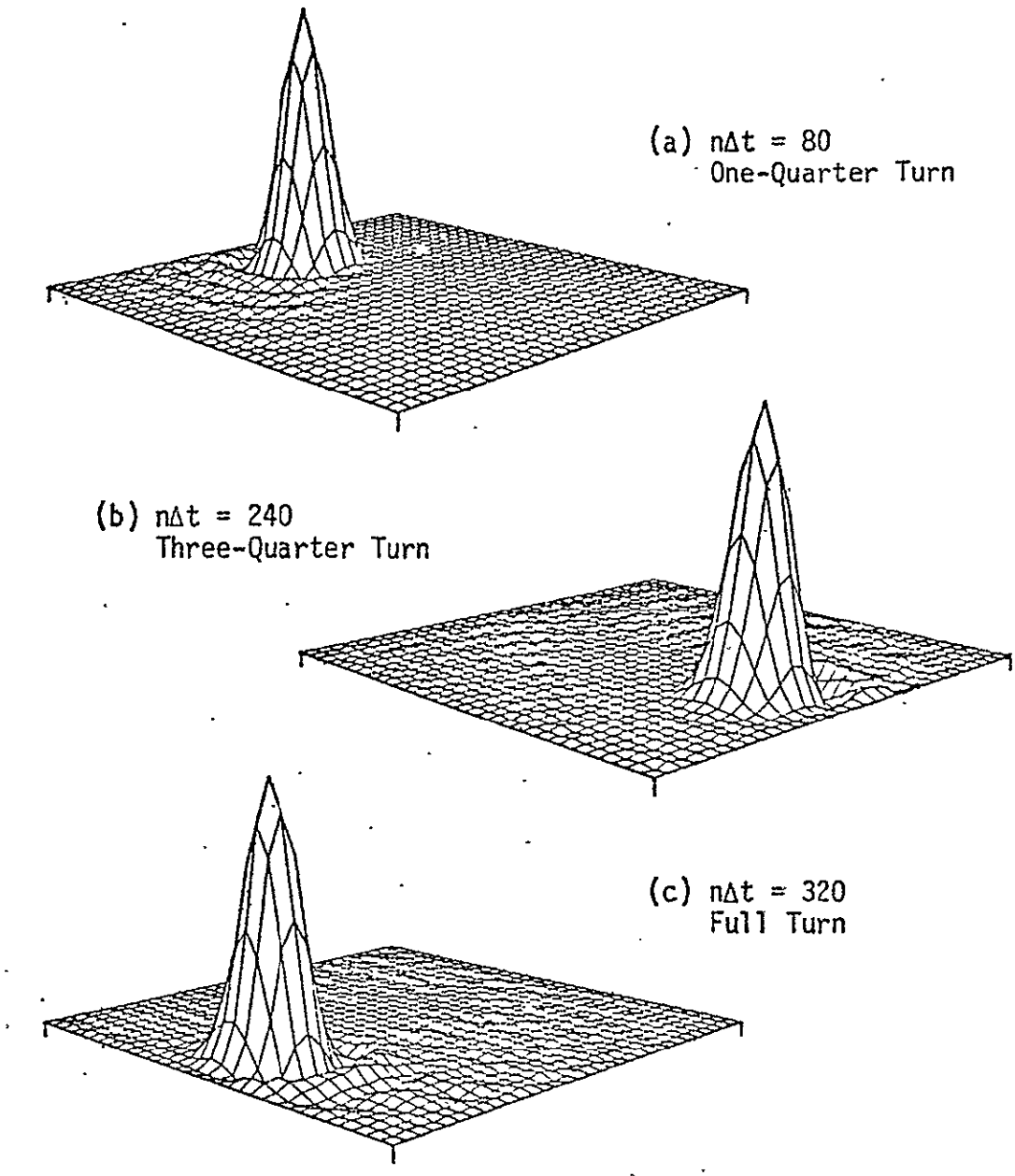
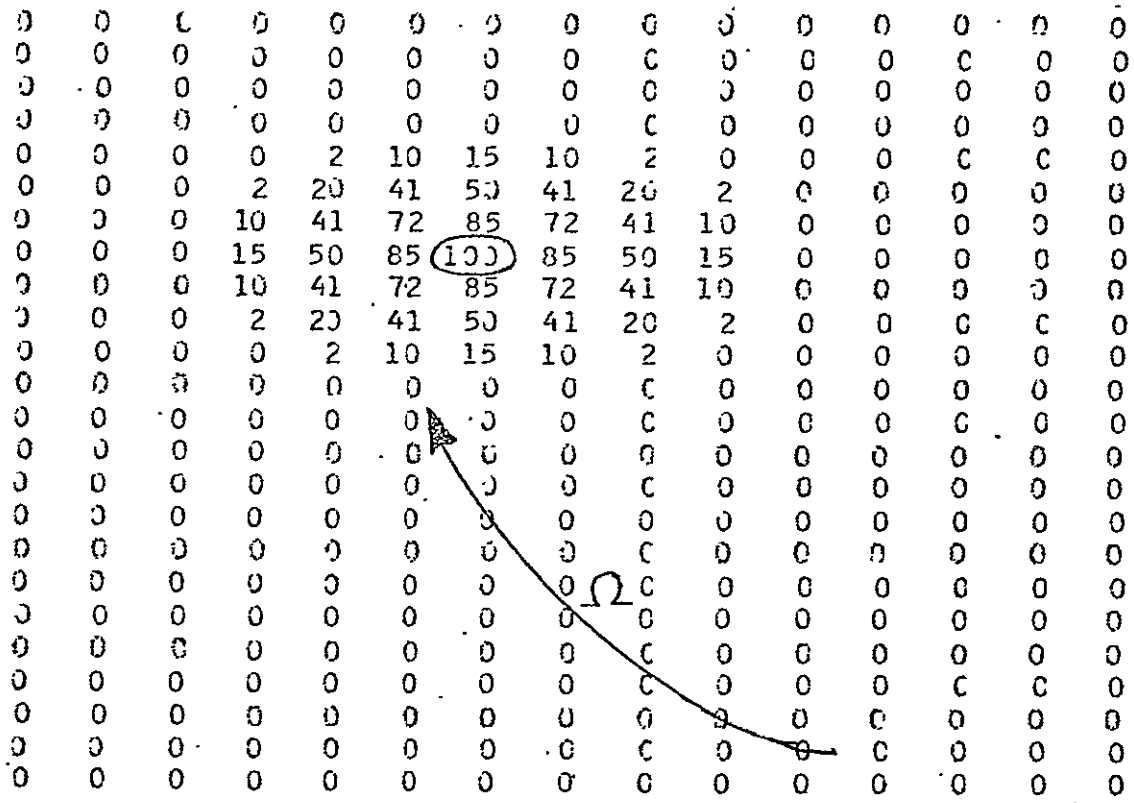
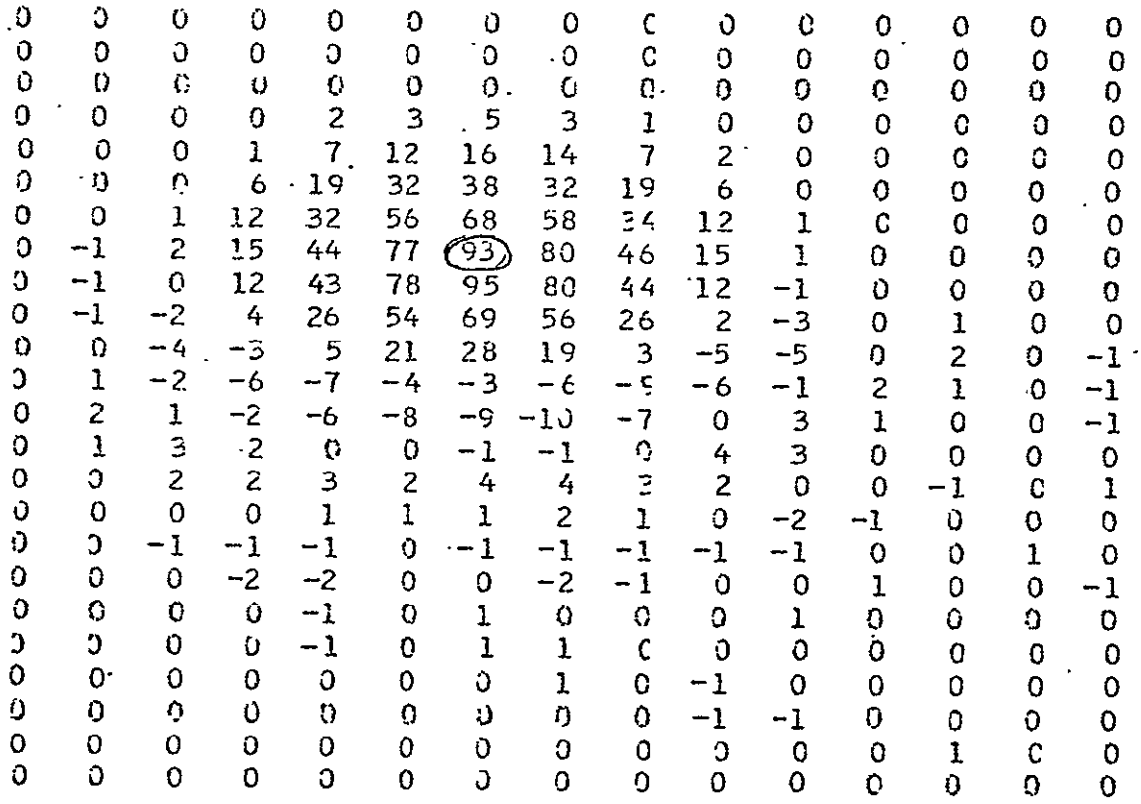


Fig. 13. Advection of Cosine Hill in Solid-Body Rotation Velocity Field, C = 0.15, k = 1.



a) Initial Condition, Exact Final Solution



b) Linear Tensor Product Solution, One Rotation, C = 0.15, ν = 0.

Fig. 14. Advection of Cosine Hill in Solid Body Rotation

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	2	10	14	10	4	0	-1	0	0	0	0
0	0	0	2	15	36	44	37	19	2	-2	0	0	0	0
0	0	0	10	34	68	84	70	40	9	-1	0	0	0	0
→ 0	0	0	15	50	87	(105)	87	50	14	0	0	0	0	0
0	0	0	11	41	73	84	70	40	10	0	0	0	0	0
0	0	-2	4	22	43	48	40	20	3	0	0	-1	0	0
0	0	0	-1	5	14	17	12	4	-1	0	0	-1	0	0
0	0	0	-2	-3	-2	-1	-2	-1	-2	1	0	-1	0	0
0	0	1	0	0	-2	-2	-1	0	0	1	0	-1	0	0
0	-1	0	1	1	1	1	1	2	0	0	0	0	1	0
0	0	-1	0	0	0	1	0	0	-1	-1	0	0	0	-1
0	0	-1	0	-1	-2	-1	-1	-1	0	1	1	0	-1	0
0	0	0	0	1	0	0	1	1	1	1	0	0	0	0
0	0	0	0	1	0	0	0	0	0	-1	-1	1	0	0
0	0	0	0	0	0	-1	0	-1	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	2	0	0	0	0
0	0	0	0	0	0	0	0	1	0	0	0	-1	0	0
0	0	0	0	0	0	0	0	0	-1	-1	0	0	0	1
0	0	0	0	0	0	0	0	0	0	0	0	1	0	-1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

a) Quadratic Tensor Product Solution, C = 0.15, ν = 0

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	1	2	1	0	0	0	0	0	0	0
0	0	0	1	3	4	5	3	1	1	0	0	0	0	0
0	0	1	2	6	11	14	12	7	1	0	0	0	0	0
0	0	0	5	13	26	31	28	17	3	-1	0	0	0	0
0	0	1	9	25	50	66	58	33	9	0	0	0	0	0
→ 0	0	2	13	38	75	(95)	84	49	13	0	0	0	0	0
0	0	0	12	42	80	99	81	44	12	0	0	0	0	0
0	0	-1	7	32	62	74	53	26	4	-1	0	-1	0	0
0	-1	-4	0	12	29	30	16	5	-2	-1	1	0	0	1
0	0	-3	-6	-5	1	0	-6	-6	-4	-1	1	0	0	1
0	0	0	-4	-8	-7	-9	-9	-4	-2	0	1	0	-1	0
0	0	1	0	-4	-6	-5	-4	0	1	1	0	0	0	1
0	0	1	1	1	0	1	1	2	2	0	0	-1	0	0
0	0	0	1	3	2	3	3	1	0	-1	-1	0	0	0
0	0	-1	-1	0	1	0	0	0	-1	-1	0	1	0	-1
0	0	0	-1	-1	-1	-1	-2	-1	-1	0	1	1	0	-2
0	0	1	0	-1	0	0	0	0	1	1	0	0	-1	0
0	0	0	0	0	0	1	2	0	1	0	-1	-1	0	1
0	0	0	0	0	0	0	0	0	0	0	0	0	1	1
0	0	0	0	0	0	0	0	0	0	-1	0	1	0	0
0	0	0	0	0	0	0	0	0	0	0	0	1	0	-1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

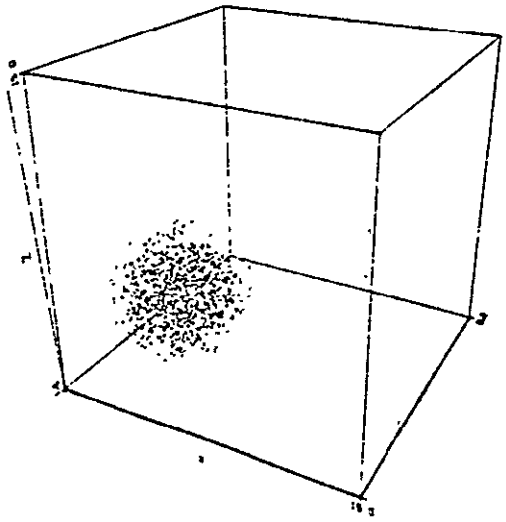
b) Quadratic Tensor Product Solution, C = 0.3, J = 0

Fig. 15. Advection of Cosine Hill in Solid Body Rotation

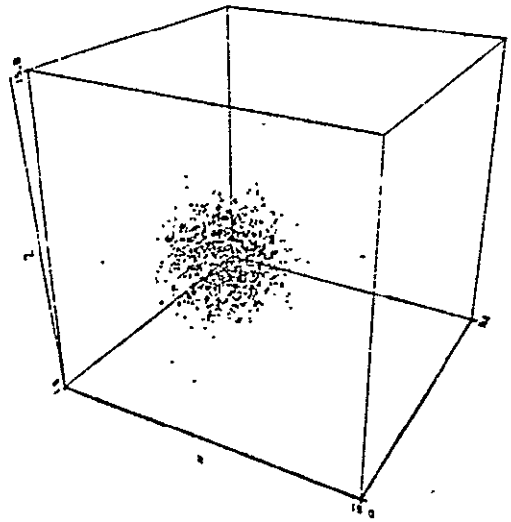
A three-dimensional solution for which an exact solution is known is pure advection of the "cosine-sphere", the three-dimensional equivalent of the "cosine-hill." The selected case is linear convection along the domain diagonal, and Figure 16 visualizes the solution in terms of particle density distributions. Here, the lagging phase of the k = 1 algorithm progressively sheds particles into the distribution wake. The accuracy of the tensor product algorithm is quantized in the remaining figures, which are printouts of the computed two dimensional distributions in the three x_3-planes centered about the exact solution mid-plane. For comparison, Figure 17 contains the initial distributions; the correct solution is pure translation to the upper right corner, and preservation of all symmetries.

The result for the non-dissipative linear algorithm, as obtained for C = 2/3, are shown in Figure 18. The peak level is almost retained (99), and the lagging phase distortion produces overall lower solution levels in the upper plane, Fig. 18c) compared to the lower plane. The dispersion error produces the evidenced trailing wakes with peaks of -7%. The solution alterations produced by introducing $\nu > 0$ are summarized in Figure 19, which are printouts of central-plane distributions for $\nu = 0.006$ and 0.012. The dispersion error peaks are modestly reduced, with the corresponding decrease in peak level.

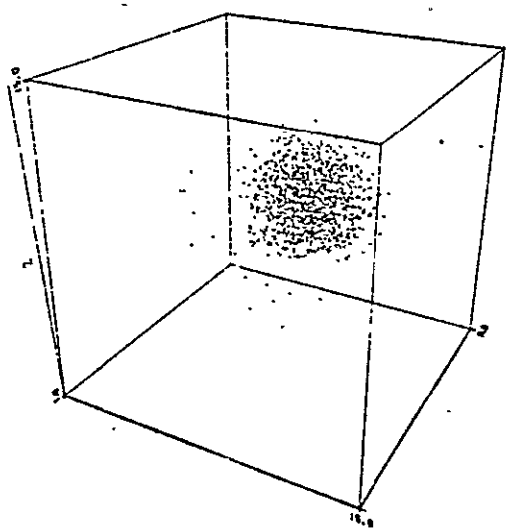
The comparison results for the quadratic algorithm at C = 2/3 are shown in Figure 20 for the non-dissipative form. The peak value is enhanced, as occurred for the two-dimensional solution, and overall symmetries are considerably improved over the linear element results. The dispersion error peaks in the trailing wake are also modestly higher in comparison. In Figure 21a), setting $\nu = 0.06$ diffuses the peak to 96%, and reduces the wake error by nominally half while retaining the symmetry preservation of the $\nu = 0$ solution.



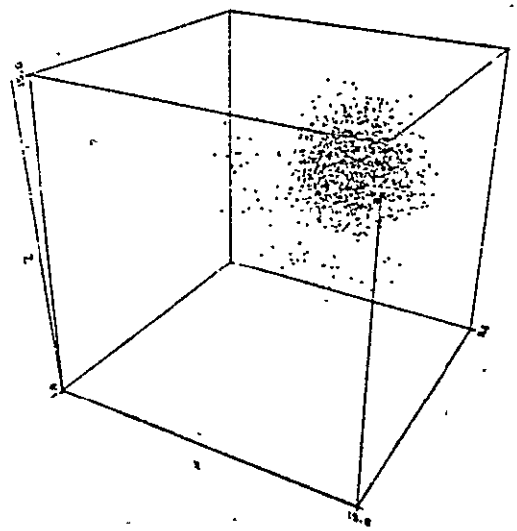
a) $n\Delta t = 0$



b) $n\Delta t = 50$

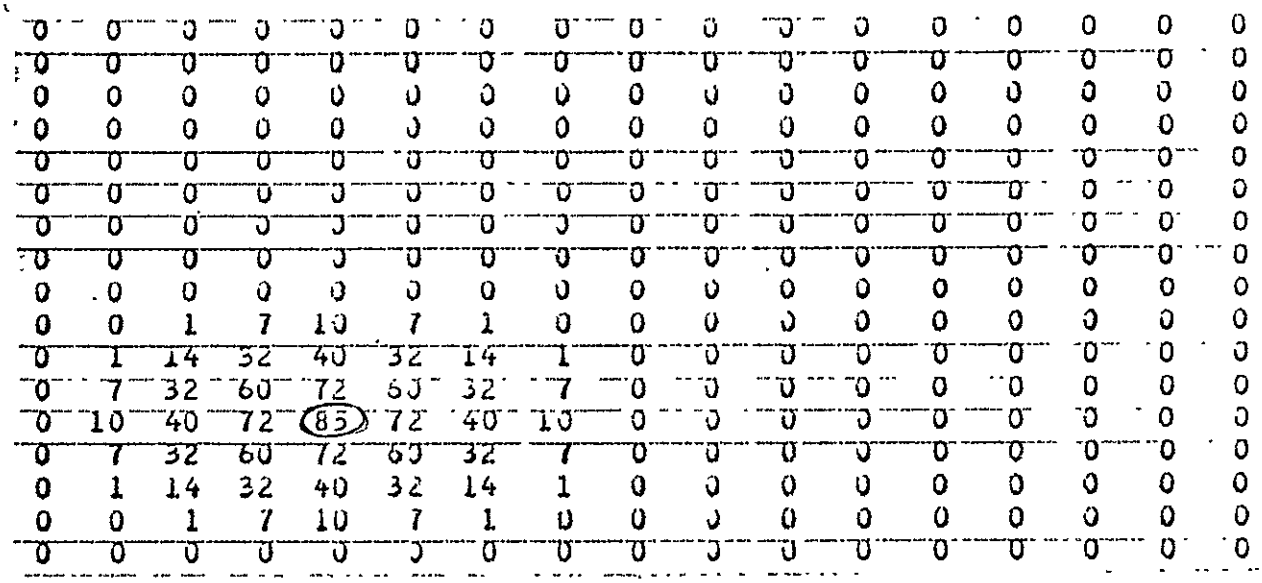


c) $n\Delta t = 100$

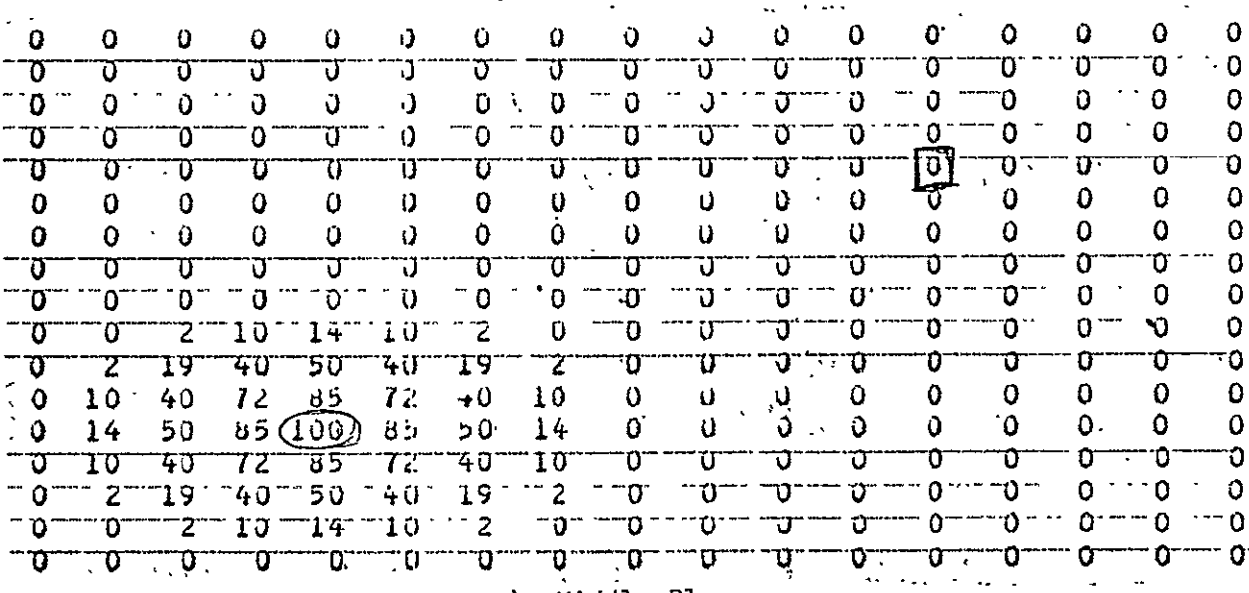


d) $n\Delta t = 150$

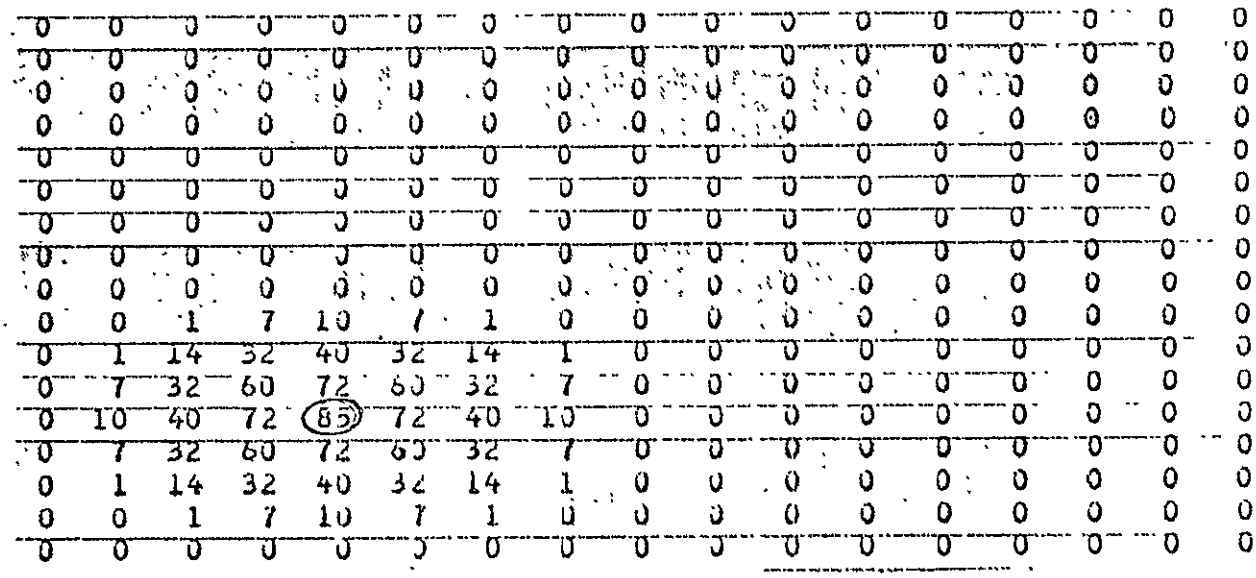
Fig. 16. Solution of Three-Dimensional Advection of Cosine-Sphere, Linear Tensor Product Algorithm, $k = 1$, $\nu = 0$.



a) Lower Plane



b) Middle Plane



c) Upper Plane

Fig. 17. Three-Dimensional Planar Initial Distributions at Center

0	0	0	0	0	0	0	0	-1	0	7	15	15	10	4	1	0
0	0	0	0	0	0	0	-1	-2	-2	18	33	35	25	11	4	0
0	0	0	0	-1	1	1	-4	-4	9	37	65	70	51	25	10	0
0	0	1	0	-1	2	0	-6	-3	16	50	84	(91)	70	35	15	0
0	0	0	0	-1	2	0	-6	-2	16	47	77	(84)	65	33	15	0
0	0	0	0	0	1	0	-2	-3	6	27	47	50	37	18	7	0
0	0	0	0	0	0	1	0	-3	-2	6	16	16	9	2	0	0
0	0	0	0	0	0	0	1	-1	-3	-3	-2	-3	-4	-2	-1	0
0	0	0	0	0	0	0	1	1	0	-2	-6	-6	-4	-1	0	0
0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0
0	0	0	0	0	0	0	0	0	0	1	2	2	1	0	0	0
0	0	0	0	0	0	0	0	0	0	0	-1	-1	-1	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

a) Lower Plane

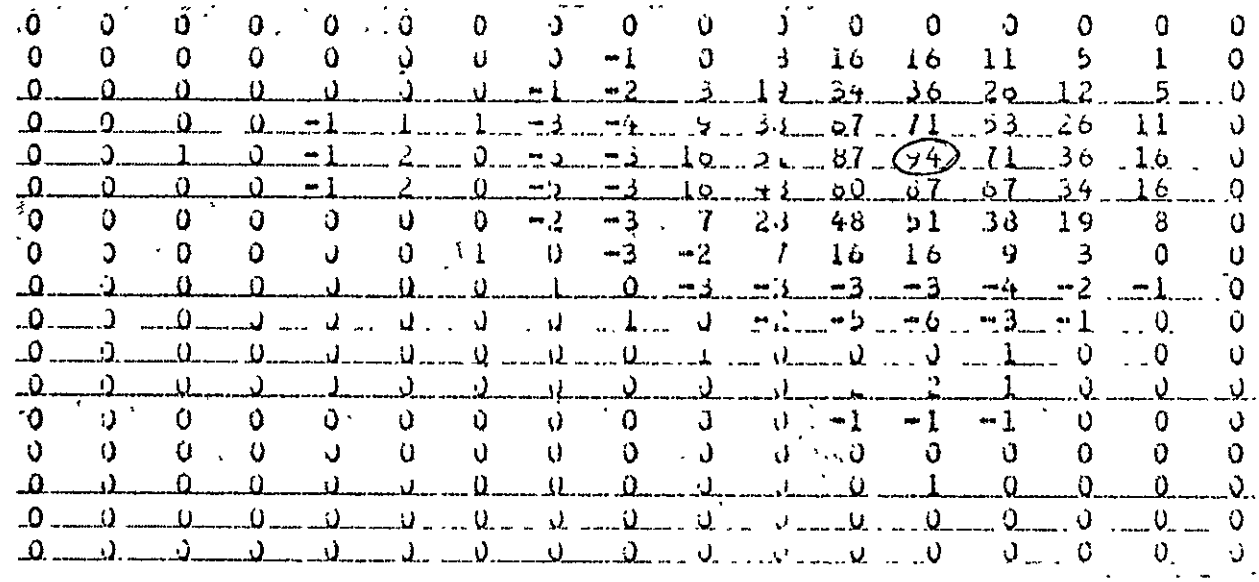
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	-1	0	8	15	16	10	4	1	0
0	0	0	0	0	0	0	-1	-3	2	19	35	37	26	12	4	0
0	0	1	0	-1	1	1	-4	-5	8	39	70	74	54	26	10	0
0	0	1	0	-2	3	0	-7	-4	14	53	91	(99)	74	37	16	0
0	0	1	0	-1	2	0	-6	-3	16	50	84	91	70	35	15	0
0	0	0	0	0	1	0	-2	-3	6	28	50	53	39	19	8	0
0	0	0	0	0	0	1	0	-3	-3	6	16	16	8	2	0	0
0	0	0	0	0	0	0	1	0	-3	-3	-3	-4	-5	-3	-1	0
0	0	0	0	0	0	0	0	1	0	-2	-6	-7	-4	-1	0	0
0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0
0	0	0	0	0	0	0	0	0	0	1	2	3	1	0	0	0
0	0	0	0	0	0	0	0	0	0	0	-1	-2	-1	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b) Middle Plane

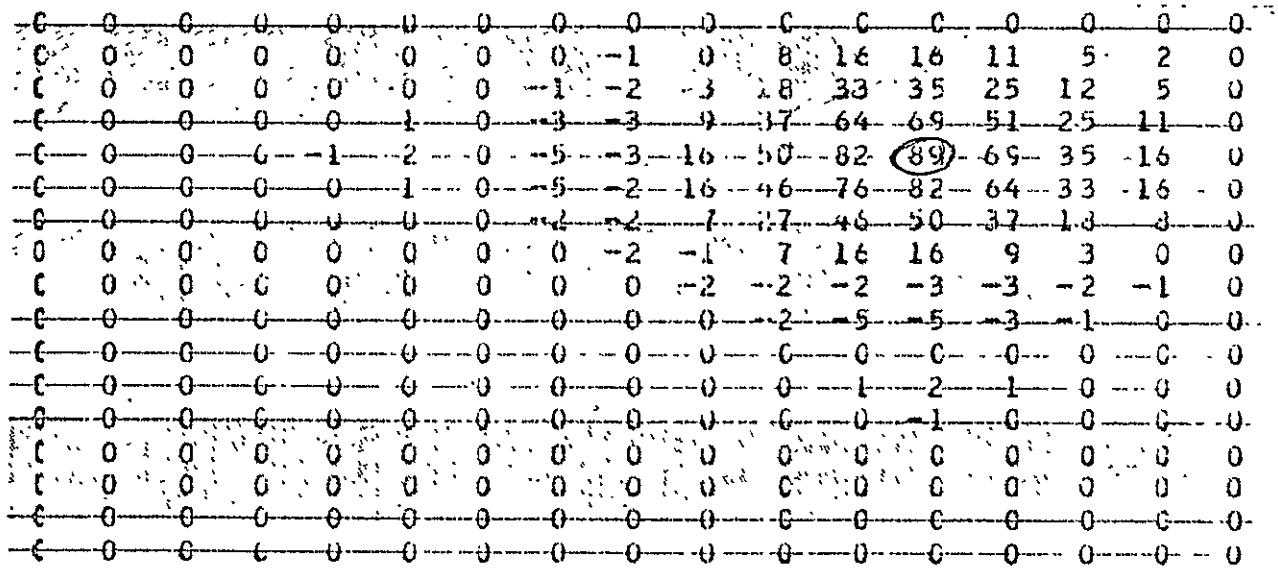
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	-1	0	5	10	10	7	3	1	0
0	0	0	0	0	0	0	0	-2	0	13	25	26	17	8	3	0
0	0	0	0	-1	0	1	-2	-4	4	28	51	54	38	17	7	0
0	0	1	0	-1	1	1	-4	-5	8	39	70	(74)	54	26	10	0
0	0	0	0	-1	1	1	-4	-4	9	37	65	70	51	25	10	0
0	0	0	0	0	0	0	-1	-3	3	20	37	39	28	13	5	0
0	0	0	0	0	0	0	0	-2	-2	3	9	8	4	0	0	0
0	0	0	0	0	0	0	0	0	-2	-3	-4	-5	-4	-2	-1	0
0	0	0	0	0	0	0	0	0	0	-1	-4	-4	-2	0	0	0
0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0
0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	-1	-1	-1	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

c) Upper Plane

Fig. 18. Computed Centroidal Plane Distributions for Three-Dimensional Solutions, k = 1, $\nu \equiv 0$, C = 2/3



a) Middle Plane, $\nu = 0.006$



b) Middle Plane, $\nu = 0.012$

Fig. 19. Computed Centroidal Plane Distributions for Three-Dimensional Solutions, k = 1, C = 2/3.

0	0	0	0	0	0	0	0	-1	0	5	12	14	9	3	1	0
0	0	0	0	0	0	1	0	-3	2	15	29	34	23	9	3	0
0	0	0	0	0	0	2	-2	-6	10	35	61	71	52	23	9	0
0	0	0	0	-1	0	3	-4	-7	17	49	79	(92)	71	34	14	0
0	0	0	0	0	0	2	-3	-5	14	42	69	79	61	29	12	0
0	0	0	0	0	0	2	-1	-4	5	23	42	49	35	15	5	0
0	0	0	0	0	0	1	0	-2	0	5	14	17	10	2	0	0
0	0	0	0	0	0	0	1	1	-2	-4	-5	-7	-6	-3	-1	0
0	0	0	0	0	0	0	0	1	0	-1	-3	-4	-2	0	0	0
0	0	0	0	0	0	0	0	0	1	2	2	3	2	1	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	-1	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

a) Lower Plane

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	-1	0	6	14	16	10	3	1	0
0	0	0	0	0	0	1	0	-3	3	17	34	39	27	10	3	0
0	0	0	1	0	0	3	-3	-8	12	41	71	84	61	27	10	0
0	0	-1	1	-1	0	4	-5	-9	21	57	92	(108)	84	39	16	0
0	0	0	0	-1	0	3	-4	-7	17	49	79	92	71	34	14	0
0	0	0	0	0	0	2	-2	-5	6	27	49	57	41	17	6	0
0	0	0	0	0	0	1	0	-3	0	6	17	21	12	3	0	0
0	0	0	0	0	0	0	1	1	-3	-5	-7	-9	-8	-3	-1	0
0	0	0	0	0	0	0	0	1	0	-2	-4	-5	-3	0	0	0
0	0	0	0	0	0	0	0	0	1	2	3	4	3	1	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	-1	-1	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	-1	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b) Middle Plane

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	4	9	10	6	2	0	0
0	0	0	0	0	0	1	0	-2	1	11	23	27	18	6	2	0
0	0	0	0	0	0	2	-1	-6	6	28	52	61	43	18	6	0
0	0	0	1	0	0	3	-3	-8	12	41	71	(84)	61	27	10	0
0	0	0	0	0	0	2	-2	-6	10	35	61	71	52	23	9	0
0	0	0	0	0	0	1	-1	-4	3	18	35	41	28	11	4	0
0	0	0	0	0	0	0	0	-1	-1	3	10	12	6	1	0	0
0	0	0	0	0	0	0	0	1	-1	-4	-6	-8	-6	-2	0	0
0	0	0	0	0	0	0	0	0	0	-1	-2	-3	-1	0	0	0
0	0	0	0	0	0	0	0	0	0	1	2	3	2	1	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

c) Upper Plane

Fig. 20. Computed Centroidal Plane Distributions for Three-Dimensional Solutions, k = 2, ν = 0, C = 2/3.

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	-1	1	7	14	16	11	4	1	0
0	0	0	0	0	0	1	0	-2	4	17	32	37	26	11	4	0
0	0	0	0	0	0	2	-2	-5	12	38	66	76	57	26	11	0
0	0	0	0	0	0	2	-4	-5	19	52	84	90	76	37	16	0
0	0	0	0	0	0	1	-3	-4	16	45	73	84	66	32	14	0
0	0	0	0	0	0	1	-1	-4	7	26	45	52	38	17	7	0
0	0	0	0	0	0	0	0	-2	0	7	16	19	12	4	1	0
0	0	0	0	0	0	0	0	0	-2	-4	-4	-5	-5	-2	-1	0
0	0	0	0	0	0	0	0	0	0	-1	-3	-4	-2	0	0	0
0	0	0	0	0	0	0	0	0	0	1	1	2	2	1	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

a) Middle Plane, ν = 0.06

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	1	7	14	16	11	5	2	0
0	0	0	0	0	0	0	0	-2	5	17	31	35	26	12	5	0
0	0	0	0	0	0	1	-2	-3	12	36	61	69	53	26	11	0
0	0	0	0	0	0	1	-3	-3	19	48	77	87	69	35	16	0
0	0	0	0	0	0	1	-2	-2	16	42	68	77	61	31	14	0
0	0	0	0	0	0	0	-1	-2	8	25	42	48	36	17	7	0
0	0	0	0	0	0	0	0	-1	1	8	16	19	12	5	1	0
0	0	0	0	0	0	0	0	0	-1	-2	-2	-3	-3	-2	0	0
0	0	0	0	0	0	0	0	0	0	-1	-2	-3	-2	0	0	0
0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b) Middle Plane, ν = 0.12

Fig. 21. Computed Centroidal Plane Distributions for Three-Dimensional Solutions, k = 2, C = 2/3.

Increasing $\nu = 0.012$ diffuses the peak further to 87% while further improving the wake error, Figure 21.b). All the three-dimensional results further confirm that acceptable levels of dissipation ν are appreciably below the linear analysis optimum value.

The computer requirements for the three-dimensional solutions encompass those of all the lower dimensional cases. A total of 120,000 words of core were sufficient to execute both the linear and quadratic algorithm forms on 17^3 mesh. The Jacobian LU decomposition, for the quadratic requires about twice the storage of the linear, but the Jacobian core constitutes less than 1% of total storage using the tensor product algorithm. The CPU to execute one sweep of the quadratic algorithm form is approximately 15 - 20% larger than to execute the linear element sweep. The linear element three-dimensional test case requires less than 1 minute of CPU on an IBM 360/195 computer for execution.

SUMMARY AND CONCLUSIONS

An accurate and efficient tensor product basis finite element solution algorithm is established for application to convection dominated flow field predictions. The intrinsic fourth-order accuracy is enhanced using a dissipative formulation to modify phase error-induced oscillations and instabilities. Embedding the formulation within the method of fractional steps yields a core-efficient procedure for implicit integration of the resultant ordinary differential equation systems.

The results of numerical experiments, for linear and non-linear model partial differential equations, firmly quantize the performance and accuracy of the developed algorithm. In particular, practically acceptable levels of

numerical diffusion demand the dissipation parameter be selected much smaller than an "optimum" linear analysis evaluation. The overall performance of the quadratic tensor product basis algorithm is modestly superior to the linear basis form, although the latter functions quite well for an exceedingly simple formulation. Minor modifications to the derived matrix structures for the linear algorithm yields familiar finite difference forms, the performance of which appears inferior based upon the results presented. The developed finite element algorithm should find wide application in computational fluid dynamics.

REFERENCES

1. Thames, F. C., Thompson, J. F., Mastin, W. C., and Walker, R. L., "Numerical Solutions for Viscous and Potential Flow About Arbitrary Two-Dimensional Bodies Using Body Fitted Coordinate Systems," J. Comp. Phys., Vol. 24, No. 3, pp. 245-273, 1977.

2. Mastin, W. C. and Thompson, J. F., "Three-Dimensional Body-Fitted Coordinate Systems for Numerical Solutions of the Navier-Stokes Equations," AIAA Paper No. 78-1147, 1978.

3. Strang, G. and Fix, G. J., An Analysis of the Finite Element Method, Prentice-Hall, New Jersey, 1973.

4. Baker, A. J. and Soliman, M. O., "Utility of a Finite Element Solution Algorithm for Initial-Value Problems," J. Comp. Phys., to appear.

5. Soliman, M. O. and Baker, A. J., "Accuracy and Convergence of a Finite Element Algorithm for Laminar Boundary Layer Flow," Computers and Fluids, to appear.

6. Soliman, M. O. and Baker, A. J., "A High Order Accurate Numerical Solution Algorithm for Turbulent Boundary Layer Flow," AIAA Paper 79-0001, 1979.

7. Prenter, P. M., Splines and Variational Methods, John Wiley, New York, 1975.

8. Dendy, J. E., "Two Methods of Galerkin Type Achieving Optimum L^2 Rates of Convergence for First Order Hyperbolics," SIAM. J. Num. Anal., V. II, No. 3, p. 637-653, 1974.

9. Halmos, P. R., Finite-Dimensional Vector Spaces, Van Nostrand, N.Y., 1958.

10. Yanenko, N. N., The Method of Fractional Steps, Springer-Verlag, New York, 1971.

11. Roache, P. J., Computational Fluid Dynamics, Hermosa Publ., Albuquerque, USA, 1972.

12. Raymond, W. H. and Garder, A., "Selective Damping in a Galerkin Method for Solving Wave Problems With Variable Grids," Monthly Weather Review, Vol. 104, pp. 1583-1590, 1976.

13. Beam, R. M. and Warming, R. F., "An Implicit Finite Difference Algorithm for Hyperbolic Systems in Conservation Law Form" J. Comp. Phys., Vol. 22, p. 87-110, 1976.

14. Pepper, D. W. & Baker, A. J., "A Simple One-Dimensional Finite Element Algorithm with Multi-Dimensional Capability," Num. Heat Transfer, V. 2, pp. 81-95, 1979.

15. Baker, A. J., Soliman, M. O., and Pepper, D. W., "A Time-Split Finite Element Algorithm for Environmental Release Prediction," Finite Element In Water Resources, Pentech Press, London, pp. 453-465, 1978.

16. Pepper, D. W. and Kern, C. D., "Modeling The Dispersion of Atmospheric Pollution Using Cubic Splines and Chapeau Functions," Atmos. Environ., Vol. , 1979.