

AD-A083 664

TENNESSE UNIV KNOXVILLE DEPT OF ENGINEERING SCIENCE --ETC F/O 12/1
RESEARCH ON NUMERICAL ALGORITHMS FOR THE THREE DIMENSIONAL NAVI--ETC(U)
SEP 79 A J BAKER AFOSR-79-0008

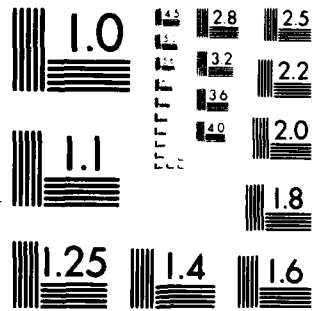
UNCLASSIFIED

AFDOL-TR-79-3101

NL

1-1-1
2/1/80

END
DATE
FILMED
6-80
DTIC



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

AFFDL-TR-79-3141

LEVEL

2
nu

ADA 083664

**RESEARCH ON NUMERICAL ALGORITHMS FOR THE
THREE-DIMENSIONAL NAVIER-STOKES EQUATIONS,
I. ACCURACY, CONVERGENCE & EFFICIENCY**

A. J. BAKER

*DEPARTMENT OF ENGINEERING SCIENCE AND MECHANICS
UNIVERSITY OF TENNESSEE
KNOXVILLE, TENNESSEE*

DECEMBER 1979

TECHNICAL REPORT AFFDL-TR-79-3141
Interim Report for period October 1978 - September 1979

DTIC
ELECTE
S APR 29 1980 D
A

Approved for public release; distribution unlimited.

AIR FORCE FLIGHT DYNAMICS LABORATORIES
AIR FORCE WRIGHT AERONAUTICS LABORATORIES
AIR FORCE SYSTEMS COMMAND
WRIGHT-PATTERSON AIR FORCE BASE, OHIO 45433

DDC FILE COPY


80 4 25 024

NOTICE

When Government drawings, specifications, or other data are used for any purpose other than in connection with a definitely related Government procurement operation, the United States Government thereby incurs no responsibility nor any obligation whatsoever; and the fact that the government may have formulated, furnished, or in any way supplied the said drawings, specifications, or other data, is not to be regarded by implication or otherwise as in any manner licensing the holder or any other person or corporation, or conveying any rights or permission to manufacture, use, or sell any patented invention that may in any way be related thereto.


This report has been reviewed by the Information Office (OI) and is releasable to the National Technical Information Service (NTIS). At NTIS, it will be available to the general public, including foreign nations.

This technical report has been reviewed and is approved for publication.


PROJECT ENGINEER


LOWELL KEEL, Maj, USAF
Chief, Aerodynamics & Airframe Branch

FOR THE COMMANDER


PETER J. BUTKEWICZ, Colonel, USAF
Chief, Aeromechanics Division

"If your address has changed, if you wish to be removed from our mailing list, or if the addressee is no longer employed by your organization please notify AEFDL/EXM, W-PAFB, OH 45433 to help us maintain a current mailing list".

Copies of this report should not be returned unless return is required by security considerations, contractual obligations, or notice on a specific document.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER AFDRL TR-79-3141	2. GOVT ACCESSION NO. AD-A083 664	3. RECIPIENT'S CATALOG NUMBER (9)
4. TITLE (and Subtitle) RESEARCH ON NUMERICAL ALGORITHMS FOR THE THREE DIMENSIONAL NAVIER-STOKES EQUATIONS. I. ACCURACY, CONVERGENCE & EFFICIENCY.		5. DATE OF REPORT & PERIOD COVERED Interim Technical Report 1 Oct. 1978 - 30 Sept. 1979
7. AUTHOR(s) A. J./Baker		8. CONTRACT OR GRANT NUMBER(s) AFOSR-79-0005 <i>new</i>
9. PERFORMING ORGANIZATION NAME AND ADDRESS Department of Engineering Science & Mechanics University of Tennessee, Knoxville, TN 37916		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS 2307N428 <i>DIN4</i>
11. CONTROLLING OFFICE NAME AND ADDRESS Air Force Flight Dynamics Laboratory Wright-Patterson AFB, Ohio 45433		12. REPORT DATE September 1979
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)		13. NUMBER OF PAGES 50
		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		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number) Navier-Stokes Numerical Solution Finite Element		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) The objective of this research is to develop a highly accurate and efficient numerical solution algorithm for the non-linear three-dimensional Navier-Stokes equations for aerodynamics applications. A candidate algorithm has been derived employing finite element interpolation theory, the non-linear extension of quasi-variational principles, and the concept of tensor product bases. The resultant solution statement is rendered <i>→ next page</i>		

40 704-495

20. Cont'd

cont → soluable using an implicit integration algorithm, and the replacement of the standard Jacobian of a Newton iteration algorithm with a tensor matrix product form. →

→ A linearized stability analysis indicates the basic algorithm is spatially fourth-order accurate in its most elementary embodiment. The control of an added dissipation mechanism can elevate this to sixth order, but numerical experimentation indicates the resultant artificial diffusion is unacceptably large. By the same measure, this additional accuracy is intrinsic to the quadratic element embodiment of the algorithm with freedom from artificial diffusion. The results of several numerical experiments for single- and multi-dimensional convection-dominated test problems confirms the basic viability of this algorithm and its tensor product formulation. The latter is of paramount importance in rendering the algorithm nominally as efficient as familiar lower-order accurate finite difference formulations.

↖

FOREWORD

This report describes work completed in the Department of Engineering Science and Mechanics, University of Tennessee, Knoxville, TN, on the topic of research on numerical solution algorithms for the three-dimensional, Navier-Stokes equations. Sponsorship was provided by the United States Air Force under USAF Grant Number AFOSR-79-0005.

The work reported herein is part of a multi-year effort and was performed during the period 1 October 1978 to 30 September 1979. The principal investigator is Dr. A. J. Baker, Associate Professor, and the contract technical monitor was Dr. Charles E. Jobe, AFFDL.

The author wishes to acknowledge the valued interaction with colleagues, especially Dr. D. W. Pepper and Dr. M. O. Soliman. Considerable assistance was provided by Mr. T. M. Chaudry in the generation of numerical solutions.

Sponsor	
File #	<input checked="" type="checkbox"/>
DDO TMS	<input type="checkbox"/>
Unpublished	<input type="checkbox"/>
Classification	<input type="checkbox"/>
By	
Classification	
Availability Codes	
Dist	Avail and/or Special
A	

TABLE OF CONTENTS

SECTION	PAGE
I. INTRODUCTION	1
II. THEORETICAL ANALYSIS	5
1. Overview.	5
2. Navier-Stokes Equations System.	5
3. The Dissipative Weighted-Residuals Algorithm.	7
4. Accuracy and Convergence, Stability	15
III. DISCUSSION OF RESULTS.	20
1. Overview.	20
2. Accuracy and Error Control.	20
3. Matrix Iterative Solution Efficiency.	35
4. Resolution of Gradients, Shocks	38
5. Transformation to Arbitrary Domains	40
IV. CONCLUSIONS AND RECOMMENDATIONS.	48
REFERENCES	49

LIST OF ILLUSTRATIONS

FIGURE		PAGE
1	Computed Substantial Derivative Operator for Continuity Equation, Courant No. = 0.1	3
2	Advection of Cosine Hill in a Solid-Body Rotation Velocity Field, C = 0.1 (Reference 17)	22
3	Advection of Cosine Hill in Solid Body Rotation, Initial Condition, Exact Final Solution	23
4	Advection of Cosine Hill in Solid Body Rotation, Linear Tensor Product Algorithm.	24
5	Advection of Cosine Hill in Solid Body Rotation, Quadratic Tensor Product Algorithm.	29
6	Two-Dimensional Square Wave Convection, Linear Tensor Product Algorithm, C = 0.125.	33
7	Two-Dimensional Square Wave Convection, C = 0.125	34
8	Initial and Pressure-Perturbed Velocity Distributions for One-Dimensional Supersonic Shocked Duct Flow.	39
9	Computed Duct Velocity Distributions Using Linear Element Algorithm	41
10	Effect of Dissipation Level v^d on Shocked Duct Flow Solution, $k = 1$, $v^i = 0$, $n\Delta t = 20$	42
11	Effect of Dissipation Level v^i on Shocked Duct Flow Solution, $k = 1$, $v^d = 0.0645$, $n\Delta t = 20$	43
12	Plane-Quadrilateral Two-Dimensional Finite Element for $[N_{1+}]$	46
13	Generalized Quadrilateral Two-Dimensional Finite Element for $[N_{2+}]$	46

LIST OF TABLES

TABLE		PAGE
1	SUMMARY OF COSINE HILL SENSITIVITY STUDY LINEAR TENSOR PRODUCT ALGORITHM, $C = 0.5$	27
2	SUMMARY OF COSINE HILL SENSITIVITY STUDY QUADRATIC TENSOR PRODUCT ALGORITHM, $C = 0.5$	31
3	MATRIX ITERATION EFFICIENCY SUMMARY	37

LIST OF SYMBOLS

a	coefficient
C	initial-value finite element matrix; Courant Number
d	parameter
e	specific energy
f	function of known argument
g	derivative column matrix; amplification factor
F	homogeneous discretized equivalent equation system
h	integration time-step
i	$\sqrt{-1}$
j	index
J	Jacobian
k	degree of finite element cardinal basis; damping coefficient
\mathcal{L}	differential operator
L	differential operator
M	number of finite elements spanning R
n	unit normal vector; dimension of space
N	finite element cardinal basis
p	pressure; iteration index
q	generalized dependent variable
Q	generalized discretized dependent variable
R	spatial domain of differential operator
S	finite element assembly operator
t	time
u_j	velocity vector
U	convection finite element matrix

V	scale velocity
\vec{x}	Cartesian coordinate system
α	parameter
β	parameter
γ	ratio of specific heats; parameter
Γ	parameter
∂	partial derivative operator
∂R	boundary of solution domain R
δ	Kronecker delta; parameter
δQ	iteration vector
Δ	measure; increment
θ	heat flux vector; integration parameter
λ	multiplier; wavelength
μ	dissipation level
ν	multiplier
ρ	density
σ	stress tensor; integration parameter
Σ	summation operator
τ	integration parameter
ω	wave number
Ω	solution domain

Superscripts:

m	m^{th} derivative
p	iteration index
T	matrix transpose
$\hat{}$	unit vector
$\dot{}$	ordinary derivative
*	solution approximation

Subscripts:

e	element reference
i,j,	tensor indices
j	time step index
k	degree of polynomial
o	initial condition
t	time derivative

Notation:

{ }	column matrix
[]	square matrix
\cup	union
\cap	intersection
\in	belongs to
\otimes	tensor product

SECTION I

INTRODUCTION

Even though finite difference fluid mechanics techniques have matured significantly in recent years, the "computational wind tunnel" remains a distant vision. Physically complete flow field predictions have been generally limited, by computer speed and cost (at least), to relatively small regions on two-dimensional space. However, with the current rapid evolution of digital hardware technology, the near-term prospect exists for emergence of a large-scale computational capability for aerodynamic flowfield prediction. Candidate fourth generation computers include at least the STAR, and CYBER, and the CRAY, and perhaps a dedicated Navier-Stokes solver (Proceedings of the NASA Workshop on Future Computer Requirements for Computational Aerodynamics, ref.1).

Numerical algorithms for solution of simplified forms of the three-dimensional Navier-Stokes equations have historically been based upon finite difference theory, (ref. 2). MacCormack's explicit time-split method, MacCormack and Paullay (ref. 3), emerged as an efficient second-order accurate algorithm for solution of the inviscid Euler equations, and combined with shock-capturing has enjoyed world-wide use. Alternatively, two-dimensional viscous flow computations have generally employed alternating-direction, implicit, second-order accurate finite difference algorithms for time (or space) evolution, and successive overrelaxation for boundary value problems and/or direct iteration at steady-state. The Crank-Nicolson algorithm is the representative example procedure (ref. 2). An implicit algorithm circumvents the severe stability constraint associated with explicit integration procedures; for example, Beam and Warming have developed implicit forms for the Euler equations (ref. 4) and the compressible Navier-Stokes equations (ref. 5).

In recent years, the engineering community has become cognizant of functional analysis and interpolation theory as an alternative theoretical foundation for development of numerical algorithms for the Navier-Stokes equations. The finite element method is the engineering embodiment of these tools of the mathematician. The apparent fundamental difference between finite difference and finite element theory is that the former deals with difference algebra while the latter employs vector field theory and

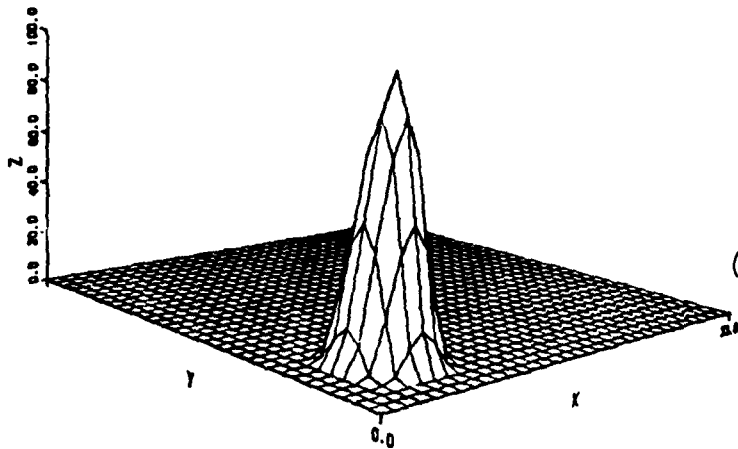
calculus. However, as must occur, both concepts belong to interpolation theory, Prenter (ref. 6). A strong common theoretical foundation is thus emerging, which is recognized and employed herein to assist in derivation of optimally-accurate, and efficient algorithms for the highly non-linear Navier-Stokes equations. For example, use of a linear finite element algorithm applied to the diffusion term in the Navier-Stokes equations on a uniform grid yields identically the second-order accurate finite difference form, i.e., the central difference operator. This does not necessarily extend to initial-value and/or non-linear differential operators however. For example, finite element theory renders the linear finite element discrete embodiment of the substantial time-derivative,

$$\frac{\partial}{\partial t} + \vec{u} \cdot \nabla$$

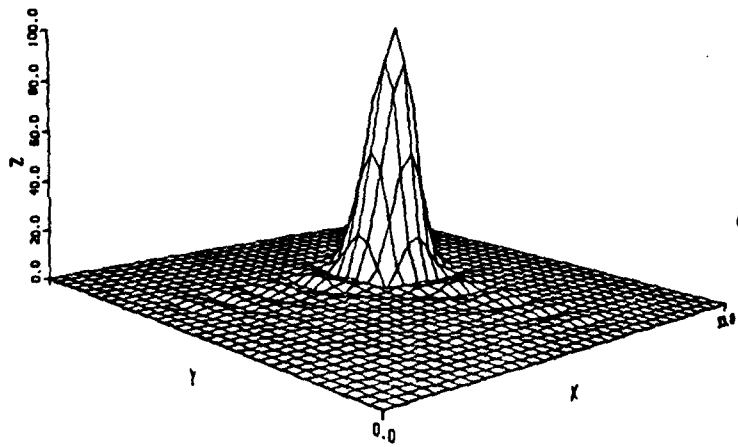
a fourth-order accurate representation on a uniform grid. Figure 1a-b illustrates the accuracy obtainable for implicit solution of the (hyperbolic) continuity equation using linear finite element theory. An elementary rearrangement produces the Crank-Nicolson finite difference form, which is only a second-order accurate algorithm. The computed results shown in Fig. 1c illustrate the significant loss in accuracy attendant with this algorithm for this discretization.

This elementary comparison simply documents that the finite element embodiment of linear interpolation theory can produce optimally-accurate finite difference equivalents of differential operators, representative of those dominating the three-dimensional Navier-Stokes equations. Of perhaps greater importance, finite element theory also yields a thoroughly standardized procedure for generation of potentially higher-order accurate operators using quadratic, cubic and/or higher-dimensional interpolation theory. Confirmation for linear hyperbolic and parabolic equations is presented in reference 7.

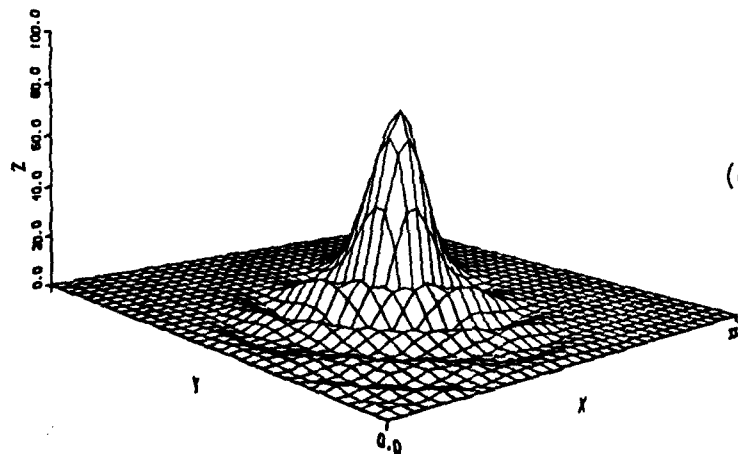
The current objective, for which results of the Phase I effort are reported herein, is to assess extension of this highly promising theoretical procedure to solution of the highly non-linear, three-dimensional Navier-Stokes equations for aerodynamic configurations. The primary requirement of Phase I was to assess factors controlling accuracy, convergence and efficiency of the



(a) Initial Distribution



(b) Linear Finite Element
150 Time Steps



(c) Second-Order Accurate
Finite Difference
150 Time Steps

Figure 1. Computed Substantial Derivative Operator for
Continuity Equation, Courant No. = 0.1

predictions produced by the developed finite element theory for partial differential equations modeling the important non-linear convection operator in the Navier-Stokes equations. The specific approach has been to delete all physical viscosity terms, which yields the Euler simplification to Navier-Stokes, to determine algorithm performance in the absence of the smoothing introduced by viscosity. This corresponds as well to the large Reynolds number limit of Navier-Stokes, wherein the field of aerodynamics generally resides. Both linear and quadratic finite element tensor product formulation have been evaluated and proven viable, as presented herein.

SECTION II

THEORETICAL ANALYSIS

1. Overview

The multi-year goal is to derive, implement into a computer program, and evaluate accurate and economical numerical algorithms for solution of the three-dimensional Navier-Stokes equations for turbulent aerodynamic flows. This section introduces the topic with a concise statement of the appropriate differential equation system requiring solution. The uniformity pervading this system then yields a concise statement of the dissipative-Weighted Residuals finite element algorithm. A theoretical analysis of a linearized model equation is discussed that isolates and estimates acceptable levels for key parameters imbedded within the basic algorithm that control accuracy and convergence.

2. Navier-Stokes Equation System

The set of partial differential equations governing the transient three-dimensional flows of interest is the familiar and very non-linear Navier-Stokes system. All equation systems studied are derived directly from the Navier-Stokes equations. The non-dimensional conservation form (in Cartesian tensor notation with summation implied over repeated subscripts) for a compressible, viscous, heat-conducting fluid is

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

$$L(\rho u_i) = \frac{\partial (\rho u_i)}{\partial t} + \frac{\partial}{\partial x_j} [u_j \rho u_i - \tau_{ij}] = 0 \quad (2)$$

$$L(\rho H) = \frac{\partial (\rho H)}{\partial t} + \frac{\partial}{\partial x_j} [u_j \rho H - u_j (\tau_{ij} + p \delta_{ij}) - q_j] - \frac{\partial p}{\partial t} = 0 \quad (3)$$

In equations 1 - 3, ρ is density, ρu_i is the momentum vector, p is pressure, and the Stokes stress tensor τ_{ij} , heat flux vector q_j , and stagnation enthalpy H are defined as

$$\tau_{ij} = -\rho\delta_{ij} + \frac{\mu}{\text{Re}} \left[\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right] - \frac{\mu}{3\text{Re}} \frac{\partial u_k}{\partial x_k} \delta_{ij} \quad (4)$$

$$q_j \equiv -k \frac{\partial T}{\partial x_j} \quad (5)$$

$$H \equiv h + \frac{1}{2} u_i u_i \quad (6)$$

Herein, μ is the dynamic viscosity, k is thermal conductivity, T is temperature, h is static enthalpy, δ_{ij} is the Kronecker delta, and Re is the Reynolds number. An equation of state closes the system.

Equations 1 - 6 are valid for both laminar and turbulent flows. For the latter, however, their solution becomes tractable in a practical sense only after time-averaging. While more definitive resolutions are the subject of current research, conventional mass-weighted time-averaging will probably serve the aerodynamic requirements. Therefore, the Reynolds decomposition yields (ref. 9)

$$u_j(\vec{x}, t) \equiv \bar{u}_j(\vec{x}) + u_j'(\vec{x}, t) \quad (7)$$

The mass-weighted time-average velocity is defined as

$$\bar{u}_j \equiv \overline{\rho u_j} / \bar{\rho} \quad (8)$$

and

$$\overline{\rho u_i'^2} \equiv \lim_{T \rightarrow \infty} \int_t^{t+T} (\rho u_i - \bar{\rho} \bar{u}_i) dt \equiv 0 \quad (9)$$

This operation yields the important relation

$$\overline{\rho u_i u_j} = \bar{\rho} \bar{u}_i \bar{u}_j + \overline{\rho u_i' u_j'} \quad (10)$$

Equations 7 - 8 are also employed to define the time-averaged and fluctuating components of the scalar fields h , H , and T . Hence, for example,

$$\bar{H} = \bar{h} + \frac{1}{2} \bar{u}_i \bar{u}_i + \frac{1}{2} \overline{\rho u_i' u_i'} / \bar{\rho} \quad (11)$$

Substitution of the defining equations 7 - 8 into 1 - 6, time averaging and collecting terms yields the time-averaged Navier-Stokes equation system

$$L(\bar{\rho}) = \frac{\partial \bar{\rho}}{\partial t} + \frac{\partial}{\partial x_j} [\bar{u}_j \bar{\rho}] = 0 \quad (12)$$

$$L(\bar{\rho} \bar{u}_i) = \frac{\partial (\bar{\rho} \bar{u}_i)}{\partial t} + \frac{\partial}{\partial x_j} [\bar{u}_j \bar{\rho} \bar{u}_i + \bar{\rho} \delta_{ij} - \bar{\sigma}_{ij} + \overline{\rho u_i' u_j'}] = 0 \quad (13)$$

$$L(\bar{\rho} \bar{H}) = \frac{\partial (\bar{\rho} \bar{H})}{\partial t} + \frac{\partial}{\partial x_j} [\bar{u}_j \bar{\rho} \bar{H} + \overline{\rho H' u_j'} - \bar{u}_i \bar{\sigma}_{ij} - \overline{u_i' \sigma_{ij}'} + \bar{q}_j] - \frac{\partial \bar{\rho}}{\partial t} = 0 \quad (14)$$

where

$$\bar{\sigma}_{ij} \equiv \bar{\tau}_{ij} + \bar{\rho} \delta_{ij} \quad (15)$$

is constituted solely of viscous contribution to the state of stress. The Reynolds stress tensor $\overline{\rho u_i' u_j'}$ represents six additional unknowns that have been introduced into equations 12 - 14. Various techniques are available to accomplish closure, references 8 - 10. The primary emphasis in the current assessment corresponds to the identical vanishing of both stress tensors.

3. The Dissipative-Weighted Residuals Algorithm

The time-averaged Navier-Stokes equations are established. This system encompasses the Euler equations for inviscid flows as a subset, as well as the various simplified forms including parabolic Navier-Stokes, boundary layer, and incompressible flows including the streamfunction-vorticity form. Independent of the particular sub-class, each of the partial differential equations of the various coupled systems is of the form

$$L(q_j) = \frac{\partial q_j}{\partial \tau} + \frac{\partial}{\partial x_i} [u_i q_j + \sigma_{ij} + f \delta_{ij}] = 0 \quad (16)$$

In equation 16, q_j is a generalized dependent variable (which could be a vector, e.g., ρu_j), u_i is the convection velocity, σ_{ij} is a stress tensor (if present), and f is any non-homogeneity. The n-dimensional problem is

defined on the Euclidean space R^n , spanned by the \vec{x} coordinate system with scalar components x_i , $1 \leq i \leq n$, and τ is a generalized initial-value coordinate. The solution domain Ω is defined as the product of R^n and τ , for all elements of \vec{x} belonging to R^n and all elements of τ belonging to the open interval measured from τ_0 , i.e.,

$$\Omega \equiv R^n \times \tau = \{(\vec{x}, \tau) : \vec{x} \in R^n \text{ and } \tau \in [\tau_0, \tau)\}$$

The boundary $\partial\Omega$ of the solution domain is the product of the boundary ∂R of R^n , spanned by \vec{x} , and τ , i.e., $\partial\Omega \equiv \partial R \times \tau$. Thereupon, a differential constraint may be applied of the form

$$L(q_j) = a_1 q_j + a_2 \frac{\partial}{\partial x_i} q_j \hat{n}_i + a_3 = 0 \quad (17)$$

In equation 17, the a_i are specified coefficients and \hat{n}_i is the outwards pointing unit normal vector. Finally, an initial distribution of q_j on $\Omega_0 \equiv R^n \times \tau_0$

$$q(\vec{x}, \tau_0) \equiv q_0(\vec{x}) \quad (18)$$

The basic concept of a (finite element) numerical solution algorithm is to subdivide Ω into non-overlapping sub-domains Ω_e , such that their union forms Ω , i.e.,

$$\Omega \equiv U \Omega_e \equiv U R_e^n \times \tau \quad (19)$$

and to enforce approximate satisfaction of equations (16) - (18) on $U \Omega_e$. The finite element approach is to assume every member of the set q_j interpolated on Ω_e as

$$q_e(\vec{x}, \tau) \equiv \{N_k(\vec{x})\}^T \{Q(\tau)\}_e \quad (20)$$

In equation (20), the elements of the row matrix $\{N_k(\vec{x})\}^T$ are typically assumed polynomials on \vec{x} , complete to degree k , and the elements of $\{Q(\tau)\}_e$ are the (unknown) expansion coefficients that are required determined by numerical procedures. The resultant solution q^* , which is the numerical approximation to

$q(\vec{x}, \tau)$, is then the summation of the individual q_e over the total number of elements (M) spanning Ω as

$$q(\vec{x}, \tau) \approx q^*(\vec{x}, \tau) \equiv \sum_{e=1}^M q_e(\vec{x}, \tau) \quad (21)$$

The fundamental requirement in formulation of a numerical solution algorithm is to render the "distance" between q^* and q a minimum, i.e., to make the error in the approximate solution q^* as small as possible. Since the approximate solution q^* lies in the domain spanned by the interpolation basis $\{N_k\}$, then the error can be minimized by requiring it to be orthogonal to this space. The measurable error is $L(q^*) \neq 0$, as results from substitution of equation 21 into 16. From variational boundary value theory (ref. 11), this error is minimized (in an appropriate norm) by requiring it to be orthogonal to $\{N_k\}$. Quite simply, then,

$$\int_{R^n} \{N_k(\vec{x})\} L(q^*) \equiv \{0\} \quad (22)$$

is the desired statement. (It is of considerable interest that, with extension of the "weights" $\{N_k\}$ to include other specified parameters, equation 22 can be a theoretical basis for derivation of most finite difference algorithms, reference 12. However, selection of weights other than $\{N_k\}$ normally does not enhance solution accuracy, as discussed herein, and in reference 7.) An addition to equation 22 is desired, as becomes apparent from the theoretical analysis in the next section, since equation 22 can yield a neutrally stable algorithm with no inherent dissipative mechanisms for damping generated numerical error. The desired form can be achieved by requiring the gradient of $L(q^*)$ to also be orthogonal to $\{N_k\}$, i.e.,

$$\vec{\beta} \cdot \int_{R^n} \{N_k\} \nabla L(q^*) \equiv \{0\} \quad (23)$$

where $\vec{\beta}$ is an n-dimensional vector to be determined. Finally, for consistency, the error on $\partial\Omega$ must also be minimized; hence,

$$\lambda \int_{\partial R} \{N_k\} \delta(q^*) \equiv \{0\} \quad (24)$$

where λ is a scalar to be determined.

The dissipative Weighted-Residuals algorithmic statement is the linear combination of equations 22 - 24. Since the $\{N_k(\vec{x})\}$ are defined non-zero only locally on each subdomain (finite element) Ω_e , the global integrations indicated are replaced by integration on R_e^n , and their individual contributions appropriately summed ("assembled") to yield the algorithmic statement

$$S_e \left[\int_{R_e^n} \{N_k\} L(q_e) - \vec{\beta} \cdot \int_{R_e^n} \{N_k\} \nabla L(q_e) + \lambda \int_{\partial R_e \cup \partial R^n} \{N_k\} \ell(q_e) \right] \equiv \{0\} \quad (25)$$

In equation 25, S_e is the defined assembly operator which amounts to row-wise element summations within the defined element matrices. Similarly, $L(q^*)$ etc. has become replaced by an element of its definition $L(q_e)$, see equation 21.

Since the matrix elements of $\{N_k\}$ and the various errors $L(q_e)$, $\nabla L(q_e)$ and $\ell(q_e)$ all possess known functional dependence on \vec{x} , the indicated integrals are readily evaluated. The sole unknown dependence is on the initial value coordinate τ ; hence, equation 25 is a system of ordinary differential equations which express the τ -derivative of $q^*(\vec{x}, \tau)$ in terms of its current distribution of nodal values $\{Q(\tau)\}$. Therefore, equation 25 takes the form of the matrix ordinary differential equation system.

$$S_e \left[[C]_e \{Q\}'_e + [U]_e \{Q\}_e + [K]_e \{Q\}_e + \{f\}_e \right] \equiv \{0\} \quad (26)$$

The terms in equation 26 correspond one-to-one with those in equation 16, i.e., the first accounts for evolution, the second convection, the third the stress field, and the fourth contains all non-homogeneities. The $[U]_e$ and $[K]_e$ matrices in equation 26 are general non-linear functions of $\{Q_e\}$; only the dominant non-linearity is explicitly expressed.

These theoretical proceedings have thus transformed a single non-linear partial differential equation into a much larger system of non-linear ordinary differential equations. The requirement is to integrate this system. For efficiency, a single-step implicit algorithm is preferred, and all are expressed within the expression

$$\{Q\}_{j+1} \equiv \{Q\}_j + h \left[\theta \{Q\}_{j+1} + (1 - \theta) \{Q\}_j \right] \quad (27)$$

where $0 \leq \theta \leq 1$ controls implicitness, stability and accuracy, and j is the integration step index and h is the step size. Equation 26 provides the (implicit) expression for the $\{Q\}$ column matrices. While not acceptable in practice, equations 26 - 27 can be combined using a formal inverse operation, yielding

$$\begin{aligned} \{Q\}_{j+1} = \{Q\}_j - h\theta [C]_{j+1}^{-1} & \left\{ \left[[U]_{j+1} + [K]_{j+1} \right] \{Q\}_{j+1} + \{f\}_{j+1} \right\} \\ & - h(1 - \theta) [C]_j^{-1} \left\{ \left[[U]_j + [K]_j \right] \{Q\}_j + \{f\}_j \right\} \end{aligned} \quad (28)$$

Combining like terms and letting I signify the identity matrix, equation 28 can be rewritten as

$$\begin{aligned} & \left[I + h\theta [C]_{j+1}^{-1} \left([U]_{j+1} + [K]_{j+1} \right) \right] \{Q\}_{j+1} \\ & = \left[I - h(1 - \theta) [C]_j^{-1} \left([U]_j + [K]_j \right) \right] \{Q\}_j \\ & - h \left\{ \theta [C]_{j+1}^{-1} \{f\}_{j+1} + (1 - \theta) [C]_j^{-1} \{f\}_j \right\} \end{aligned} \quad (29)$$

which is a non-linear algebraic equation system for $\{Q\}_{j+1}$.

Therefore, the final algorithmic requirement is to establish a suitable equation solving technique for equation 29. One approach, preferred by some in the finite difference community, is to linearize the terms in the left matrix of equation 29, and achieve a direct solution. The alternative approach is to employ a matrix iteration procedure with full non-linearity retained. To accomplish the former, the inverse operation $[C]_{j+1}^{-1}$ and indicated matrix multiplications in equation 30 must be completed. Only if $[C]$ is a diagonal matrix is the inverse trivial, and the matrix multiplied $[U]_k$ and $[K]_k$ matrices for $k = j, j+1$, become modified in an elementary manner, say $[\bar{U}]_k$ and $[\bar{K}]_k$. (Of course, this diagonalizing operation destroys the order-of-accuracy intrinsic to the basic finite element algorithm and is to be avoided.) Formally, however, the solution algorithm for equation 29 can be made non-iterative, assuming a Taylor series expansion for the modified matrix terms as, e.g.

$$\begin{aligned} [\bar{U}]_{j+1} &= [\bar{U}]_j + \frac{\partial[\bar{U}]}{\partial\{Q\}} \left(\{Q\}_{j+1} - \{Q\}_j \right) + \dots \\ &\approx [\bar{U}]_j \end{aligned} \quad (30)$$

The second expression is obtained assuming the dominant non-linearity has already been extracted in equation 26. Then, assuming $\theta = 1/2$ for simplicity, equation 30 becomes expressible as

$$\begin{aligned} \left[I + \frac{h}{2} \left([\bar{U}]_j + [\bar{K}]_j \right) \right] \{Q\}_{j+1} \\ = \left[I - \frac{h}{2} \left([\bar{U}]_j + [\bar{K}]_j \right) \right] \{Q\}_j - \frac{h}{2} \left(\{\bar{f}\}_{j+1} + \{\bar{f}\}_j \right) \end{aligned} \quad (31)$$

which is a linear algebraic equation system for $\{Q\}_{j+1}$ and directly solvable.

Based upon work reported herein, the steps necessary to achieve equation (31) are undesirable in terms of accuracy. In addition, formation of $[C]_k$ and $[C]_k^{-1}$ are to be avoided at all cost. While $[C]_{j+1}^{-1}$ can be cleared from the left of equation 29 by a premultiplication by $[C]_{j+1}$, whereupon $[C]_{j+1}[I] = [C]_{j+1}$, difficulty appears on the right with terms in $[C]_{j+1}[C]_j^{-1} \neq [I]$. This is removable however, by limiting $\theta \equiv 1/2$ which produces the trapezoidal rule. Formally, using equation 26 assembled and evaluated at the mid-interval, equation 28 then becomes reexpressed for $\theta = 1/2$ as

$$\begin{aligned} \{Q\}_{j+1} &= \{Q\}_j - \frac{h}{2} [C]_{j+\frac{1}{2}}^{-1} \left\{ \left([U]_{j+\frac{1}{2}} + [K]_{j+\frac{1}{2}} \right) \{Q\}_{j+\frac{1}{2}} + \{f\}_{j+\frac{1}{2}} \right\} \\ &= \{Q\}_j - \frac{h}{2} [C]_{j+\frac{1}{2}}^{-1} \left\{ \left([U]_{j+1} + [K]_{j+1} \right) \{Q\}_{j+1} + \{f\}_{j+1} \right. \\ &\quad \left. + \left([U]_j + [K]_j \right) \{Q\}_j + \{f\}_j \right\} \dots \quad (32) \end{aligned}$$

No difficulty is then encountered in clearing the inverse, and equation 32 takes the form

$$\begin{aligned} \left[[C]_{j+\frac{1}{2}} + \frac{h}{2} \left([U]_{j+1} + [K]_{j+1} \right) \right] \{Q\}_{j+1} \\ = \left[[C]_{j+\frac{1}{2}} - \frac{h}{2} \left([U]_j + [K]_j \right) \right] \{Q\}_j - \frac{h}{2} \left(\{f\}_{j+1} + \{f\}_j \right) \end{aligned} \quad (33)$$

By assumption,

$$[C]_{j+\frac{1}{2}} \equiv \frac{1}{2} \left([C]_{j+1} + [C]_j \right) \quad (34)$$

and the Taylor series expansion yields

$$[C]_{j+1} = [C]_j + \frac{\partial [C]}{\partial \{Q\}} \left(\{Q\}_{j+1} - \{Q\}_j \right) + \dots \quad (35)$$

The second term vanishes identically for the Euler equations in conservative form, and the linearized direct solution form for equation 33 is

$$\begin{aligned} & \left[[C]_j + \frac{h}{2} \left([U]_j + [K]_j \right) \right] \{Q\}_{j+1} \\ & = \left[[C]_j - \frac{h}{2} \left([U]_j + [K]_j \right) \right] \{Q\}_j - \frac{h}{2} \left(\{f\}_{j+1} + \{f\}_j \right) \end{aligned} \quad (36)$$

Equation 36 retains the accuracy of the basic algorithm and is unconditionally stable according to a linearized analysis.

The desirable feature of equation 36 is that it is linear, hence solvable in a non-iterative manner using any direct procedure, e.g., Gaussian elimination, Cholesky decomposition, etc. The price extracted for this feature is that, while the $[C]$, $[U]$, and $[K]$ matrices are banded, they become block-banded for solution of systems of partial differential equations. Of course, direct solutions are rarely used for the sake of computer cost and storage, but are replaced by partial factorization of the matrix (ref. 4). At the expense of extraneous terms being added to the differential equation, this operation produces block tridiagonal matrices which can be efficiently solved. Hence, the order of accuracy of the basic algorithm, equation 29, is retained, while there may be a specific degradation of accuracy level at any given solution point.

The alternative approach is to matrix solve equation 33 as non-linear. Letting $\{F\} \equiv \{0\}$ be the homogeneous form of equation 33, the Newton iteration algorithm is

$$\left[J \left(Q_{j+1}^p \right) \right] \{ \delta Q \}_{j+1}^{p+1} = - \left\{ F \left(Q_{j+1}^p \right) \right\} \quad (37)$$

The dependent variable in equation 37 is the iteration vector, defined by the relation

$$Q_{j+1}^{p+1} \equiv Q_{j+1}^p + \delta Q_{j+1}^{p+1} \quad (38)$$

where p is the iteration index. The right side of equation 37 is equation 33 evaluated with the pth iterate, i.e.,

$$\{F\}_{j+1}^p = S_e \left[[C]_e \left(\{Q\}_{j+1}^p - \{Q\}_j \right) + \frac{h}{2} \left(\{g_e\}_{j+1}^p + \{g_e\}_j \right) \right] \quad (39)$$

where

$$\{g_e\}_\ell^p \equiv ([U]_e + [K]_e) \{Q\}_\ell^p + \{f\}_\ell \quad (40)$$

The vanishing of {F} to within definition of computed zero renders equation 37 homogeneous, hence convergence of the iteration. By definition, the Jacobian is the derivative of equation 39 with respect to $\{Q\}_{j+1}^p$. Hence,

$$[J] = S_e \left[[C]_e + \frac{h}{2} ([U]_e + [K]_e) + \frac{h}{2} \frac{\partial ([U]_e + [K]_e)}{\partial \{Q\}_e} \{Q\}_e \right] \quad (41)$$

where the final term accounts for contributions stemming from implicit non-linearity. The rank of [J] equals the order of $\{\delta Q\}$; Dirichlet boundary constraints are applied within the evaluation of {F}.

The ostensible key feature of the iterative solution algorithm, equations 37 - 41 is accuracy and elimination (potentially) of the block-banded Jacobian matrix structure for multiple variable systems. Several other potentially attractive attributes emerge on closer analysis. First, as discussed for the non-iterative algorithm, three-dimensional solution requirements demand a spatial factorization that permits replacement of the large, sparse-matrix Jacobian operations with elementary banded-matrix procedures. This can be accomplished in the present instance by replacing the multi-dimensional interpolation basis $\{N_k(\vec{x})\}$, equation 20, with the tensor product basis (ref. 13)

$$\{N_k(\vec{x})\} \Rightarrow \{N_k(x_\alpha)\} \otimes \{N_k(x_\beta)\} \otimes \{N_k(x_\gamma)\} \quad (42)$$

where \otimes signifies the tensor product. The Greek letter subscripts are not summation indices but refer instead to scalar components of the resolution of \vec{x} on to the coordinate system spanning R^n . Using equation 42, the original finite element algorithm statement, equation 26, becomes replaced by the tensor matrix product form. For example, the first matrix becomes of the form

$$[C]_e \Rightarrow [C_1]_e \otimes [C_2]_e \otimes [C_3]_e \quad (43)$$

Hence, the Jacobian can be evaluated as the tensor matrix product,

$$[J] \Rightarrow [J_1] \otimes [J_2] \otimes [J_3] \quad (44)$$

and each of the $[J_\alpha]$ exhibit the desired narrow band matrix structure. Specifically, for $k = 1$ in equation 42, $[J_\alpha]$ is tridiagonal, while for $k = 2$ it is pentadiagonal with a cyclic profile.

This factorization yields a highly efficient solution algorithm upon application of the concept of fractional steps (ref. 14). Specifically, evaluation of the homogeneous solution statement $\{F\}$, equation 39, is replaced by the tensor product form, and the Newton algorithm becomes

$$\begin{aligned} [J_3(Q)_{j+1}^p] \otimes [J_2(\bar{Q})_{j+1}^p] \otimes [J_1(\bar{Q})_{j+1}^p] \{\delta Q\}_{j+1}^{p+1} = \\ - \{F_1(\bar{Q})_{j+1}^p\} \otimes \{F_2(\bar{Q})_{j+1}^p\} \otimes [F_3(Q)_{j+1}^p] \end{aligned} \quad (45)$$

Here, \bar{Q} and $\bar{\bar{Q}}$ represent intermediate iterates, and δQ is interpreted as the iteration vector for each respective iterate. If required for accuracy, evaluation of the right-side of equation (45) may be replaced with conventional multi-dimensional operations.

4. Accuracy and Convergence, Stability

A von Neumann stability analysis (ref. 2) can quantify the formal order of accuracy of the developed algorithm for a linearized one-dimensional

equation, hence predict an appropriate value for $\vec{\beta}$ equation 25. Therefore, consider the inviscid x_1 momentum equation 2 with constant advection velocity U_0 .

$$L(u) = \frac{\partial u}{\partial t} + U_0 \frac{\partial u}{\partial x} = 0 \quad (46)$$

The analytical solution for equation 46 is the Fourier expansion

$$u(x,t) = V \exp \left[i \omega (x - U_0 t) \right] \quad (47)$$

where $i \equiv \sqrt{-1}$, $\omega = 2\pi/\lambda$ is the wave number where λ is wavelength, and V is the initial velocity distribution.

A rearrangement of equation 25 yields the desired form for analysis. Using a Green-Gauss theorem, the second term in equation 25 can be transformed to

$$\vec{\beta} \cdot \int_{R_e^n} \{N_k\} \nabla L(q_e) = -\nabla \cdot \left[\vec{\beta} \int_{R_e^n} \{N_k\} L(q_e) \right] + \vec{\beta} \cdot \oint_{\partial R_e} \{N_k\} L(q_e) \hat{n} \quad (48)$$

Since the tensor product algorithm form is to be utilized, from reference 15 let $\vec{\beta} \equiv v_\alpha \Delta_e^\alpha \hat{i}_\alpha$, where v_α is a scalar to be determined and Δ_e^α is the measure of the discretization of the one-dimensional domain R_e^α spanned by x_α with unit vector \hat{i}_α . Under this assumption, the second term in equation 48 vanishes identically. Setting $\lambda \equiv 0$ for convenience for this analysis, equation 25 can be written in the compact tensor product form

$$S_e \left[\int_{R_e^\alpha} \left[\left(1 + \Delta_e^\alpha \frac{\partial}{\partial x_\alpha} v_\alpha \right) \{N_k(x_\alpha)\} L(q_e) \right] \right] \equiv \{0\} \quad (49)$$

It is an elementary exercise to evaluate equation 49 for any differential equation. Recalling equation 26 as the required final form, the initial-value and convection matrices are defined as

$$[C_\alpha]_e \equiv \int_{R_e^\alpha} \left[\left(1 + \Delta_e^\alpha \frac{\partial}{\partial x_\alpha} v_\alpha \right) \{N_k(x_\alpha)\} \right] \{N_k(x_\alpha)\}^T dx_\alpha \quad (50)$$

$$[U_\alpha]_e \equiv \int_{R_e^\alpha} \left[1 + \Delta_e^\alpha \frac{\partial}{\partial x_\alpha} v_\alpha \right] \{N_k\} \frac{\partial}{\partial x_\alpha} \left[\{U_\alpha\}_e^T \{N_k\} \{N_k\}^T \right] dx_\alpha \quad (51)$$

For equation 46, and limiting the interpolation basis to linear, i.e., $k \equiv 1$ in equation 42, equation 50 yields

$$[C_\alpha]_e = \frac{\Delta_e^\alpha}{6} \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} + \frac{v \Delta_e^\alpha}{2} \begin{bmatrix} -1 & -1 \\ 1 & 1 \end{bmatrix} \quad (52)$$

for v assumed a constant. For a uniform discretization, the assembly of this term in equation 26 over the two elements sharing node j yields the finite difference recursion relation

$$S_e \left[[C_\alpha]_e \{Q\}_e \right] = \frac{\Delta^\alpha}{6} \left[(1+3v) Q_{j-1} + 4Q_j + (1-3v) Q_{j+1} \right] \quad (53)$$

where Δ^α is the measure of the uniform discretization of R^α . Similarly, the convection operator in equation (46) produces (ref. 7)

$$[U_\alpha]_e = \frac{1}{6} \{U_0^\alpha\}_e^T \begin{bmatrix} \begin{Bmatrix} 2 \\ 1 \end{Bmatrix} & \begin{Bmatrix} 2 \\ 1 \end{Bmatrix} \\ -\begin{Bmatrix} 1 \\ 2 \end{Bmatrix} & -\begin{Bmatrix} 1 \\ 2 \end{Bmatrix} \end{bmatrix} + \frac{v}{2} \{U_0^\alpha\}_e^T \begin{bmatrix} \begin{Bmatrix} 1 \\ 1 \end{Bmatrix} & -\begin{Bmatrix} 1 \\ 1 \end{Bmatrix} \\ -\begin{Bmatrix} 1 \\ 1 \end{Bmatrix} & \begin{Bmatrix} 1 \\ 1 \end{Bmatrix} \end{bmatrix} \quad (54)$$

For a constant advection velocity, $\{U_0^\alpha\}_e = U_0 \{1\}$; the assembly yields the finite difference recursion relation

$$\begin{aligned} S_e \left[[U_\alpha]_e \{Q\}_e \right] &= \frac{U_0}{2} \left[-(1+2v) Q_{j-1} + 4v Q_j + (1-2v) Q_{j+1} \right] \\ &= \frac{U_0}{2} \left[-Q_{j-1} + Q_{j+1} \right] + U_0 v \left[-Q_{j-1} + 2Q_j - Q_{j+1} \right] \quad (55) \end{aligned}$$

In equation 55, the first term is the familiar second order accurate central difference convection operation. The second term is equally familiar as the second-order accurate diffusion operator with $U_0 v \Delta^\alpha$ as the "viscosity" level.

The occurrence of these two difference expressions is to be expected, as confirmation of the assertion that linear finite element methodology yields optimal order-accuracy difference relations for the same degree interpolation. The full impact of the algorithmic development becomes apparent upon completion of the von Newman stability analysis. For the complete algorithm applied to equation 46, formal order-of-accuracy is assessed using the semi-discrete Fourier expansion

$$u^*(\Delta x, t) \equiv \sum_e u_e = V \exp \left[i \omega(j\Delta x - \lambda t) \right] \quad (56)$$

where Δx is the measure of the uniform discretization of R^1 and $\lambda \equiv \beta + i\delta$, where β and δ are real numbers. Direct substitution of equation 56 into equations 53 and 55, and expanding the resultant expressions for β and δ in a Taylor series yields (ref. 15)

$$\beta = U_0 \left[1 + \left(-\frac{1}{180} + \frac{\nu^2}{12} \right) d^4 + O(d^6) \right] \quad (57)$$

$$\delta = U_0 \left[-\frac{\nu d^3}{12} + O(d^5) \right] \quad (58)$$

Here, $d \equiv \omega\Delta x$ and $O(\)$ indicates the order of the truncated term. The phase accuracy of the discrete solution u^* agrees with U_0 to sixth order in Δx by requiring $\nu^{-1} \equiv \sqrt{15}$. For $\nu > 0$, correspondingly $\delta < 0$, and an artificial damping is introduced of the form,

$$\exp \left[-\omega^4 k t + O(\Delta x^5) \right] \quad (59)$$

where $k \equiv U_0 (\Delta x)^3 / 12\sqrt{15}$ is the damping coefficient. The damping is highly selective due to the ω^4 factor.

Setting $\nu \equiv 0$ eliminates δ , and the derived algorithm is a fourth-order accurate representation of the differential equation 46. This high order accuracy, obtained with use of the simplest linear interpolation, results directly from the finite element-derived initial-value matrix $[C_\alpha]_e$. In distinction, finite difference practice replaces equation 53 with $\Delta^\alpha Q_j$, which immediately degrades the algorithm to second order accuracy. In

addition, the conventional finite difference approach to introduce dissipation is to add an "artificial viscosity" term $\mu \partial^2 u / \partial x^2$ to equation 46. The resulting dissipation term directly comparable to equation 59 is

$$\exp\left[-\omega^2 \mu t + O(\Delta x^2)\right] \quad (60)$$

The resultant artificial damping is less selective due to the wave number exponent of 2 and the appearance of the term of order Δx^2 .

Using a fully discrete Fourier analysis, see reference 16, the amplification factor for the dissipative finite element algorithm for $\theta \equiv \frac{1}{2}$ is

$$g = \frac{1 + \frac{1}{2} \cos(\omega \Delta x) - 3Cv \sin^2(\frac{1}{2}\omega \Delta x) - i\frac{3}{4}(C+2v)\sin(\omega \Delta x)}{1 + \frac{1}{2} \cos(\omega \Delta x) + 3Cv \sin^2(\frac{1}{2}\omega \Delta x) + i\frac{3}{4}(C-2v)\sin(\omega \Delta x)} \quad (61)$$

where $C \equiv U_0 \Delta t / \Delta x$ is the Courant Number. The numerator and denominator are complex conjugates for $v \equiv 0$, hence the basic (non-dissipative) algorithm is neutrally stable for all Δx and Δt . Therefore, error induced by a solution will propagate undamped and unmagnified throughout the solution domain.

Selecting $v > 0$ destroys the neutral stability, and damping of both induced error and the physical solution will result.

SECTION III

DISCUSSION OF RESULTS

1. Overview

The convergence and stability analyses are strictly limited to the elementary differential equation studied. In particular, the practical instance of a constant imposed velocity is negligible. Nevertheless, they do help interpret the predictions of numerical solutions regarding indicated error sources and their modification. Extension of the elementary analysis to include use of the quadratic basis within the algorithm is complicated by the fact that a simple finite difference nodal recursion relation does not result. Therefore, the analysis of carefully executed numerical tests appears the appropriate method to refine and extend the scope of these predictions. For this purpose, linear and non-linear differential equations, that model the convection operator intrinsic to the Navier-Stokes equations, have been selected for study. Primary emphasis is an assessment of factors affecting solution accuracy and economy. Key results of the study are discussed in this section.

2. Accuracy and Error Control

The developed finite element solution algorithm requires evaluation with respect to parameters to be selected for error control. Specifically, these include

- a) discretization/integration step (Courant Number), C
- b) finite element interpolation degree, k
- c) phase error control, ν

Recall the definition of Courant Number, $C \equiv U\Delta t/\Delta_e$, which is available of course in any numerical algorithm. The last two are more uniquely expressed within the structure of the developed algorithm.

A demanding test problem corresponds to the two dimensional form of equation 46,

$$L(u) = \frac{\partial u}{\partial t} + \vec{U}_0 \cdot \nabla u = 0 \quad (62)$$

where $\vec{U}_0 = r\dot{\omega}\hat{\theta}$ corresponds to a solid body rotation with angular velocity $\dot{\omega}$, and the initial distribution for u is the "cosine hill." The initial distribution would appear identical to Figure 1a), moved to the 9 o'clock position, and the exact solution is diffusion- and dispersion-free convection clockwise around the solution domain. From reference 16, Figures 2a) - c) illustrate a typical linear element solution at the quarter, three-quarter and full-rotation time step, for the non-dissipative form of the algorithm. The peak value remains within 1% of its original level of 100, although the lagging phase error retards its relative displacement. The ripple structure in the background plane is pure dispersion error, which propagates generally undissipated throughout the solution, in agreement with the basic neutral stability. The results shown in Figure 2d) were obtained using a digital filtering technique that effectively dissipates the background error structure. However, the concentration peak was also diffused well below its initial-value. Hence, these solutions are quite good in comparison to standard finite difference results, yet exhibit significant error upon a closer examination.

A comprehensive numerical evaluation of this problem, for variation of parameters a) -c), can firmly quantize algorithm performance and has been completed. For comparison, Figure 3 shows the appropriate portion of the actual initial distribution of u , which also corresponds to the exact final solution. Figure 3b) shows the distribution computed for $k = 1$, $\nu = 0$ and $C = 0.25$ (at the center of the solution distribution; since U_0 varies linearly with r , $0 < C < 0.4$ on the field). As noted, the peak lags by about $\frac{1}{2}\Delta_e$ of occurring at the correct node point. The background ripple structure, observed in Figure 2, occurs as nominal rows of positive and negative concentrations in Figure 3b). This phase error induced extremum level is -10 and the nominal wavelength is $4\Delta_e$. As a consequence, the symmetries of the correct solution are not apparent. However, if two additional time steps were taken, the symmetries would nominally reappear and a peak value of 99 would occur at the circled location. Hence, the solution visually appears a close approximation to the correct solution, Figure 2c).

Increasing the integration time step (or, equivalently, decreasing the mesh measure) increases the Courant number which adds truncation error to the numerical solution. Comparing Figure 4b), obtained using $C = 0.5$, $k = 1$

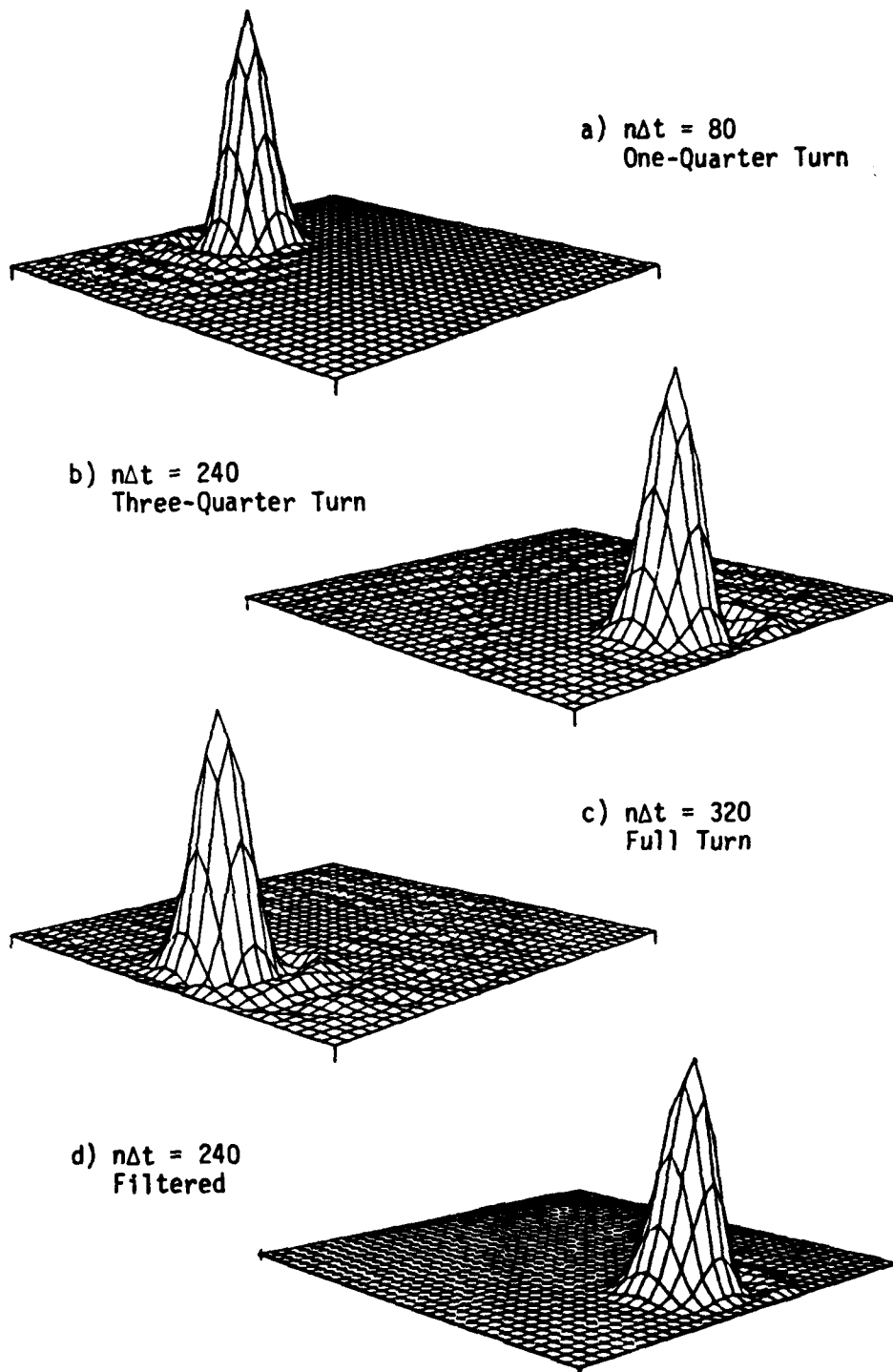


Figure 2. Advection of Cosine Hill in a Solid-Body Rotation Velocity Field, $C = 0.1$ (Reference 16)

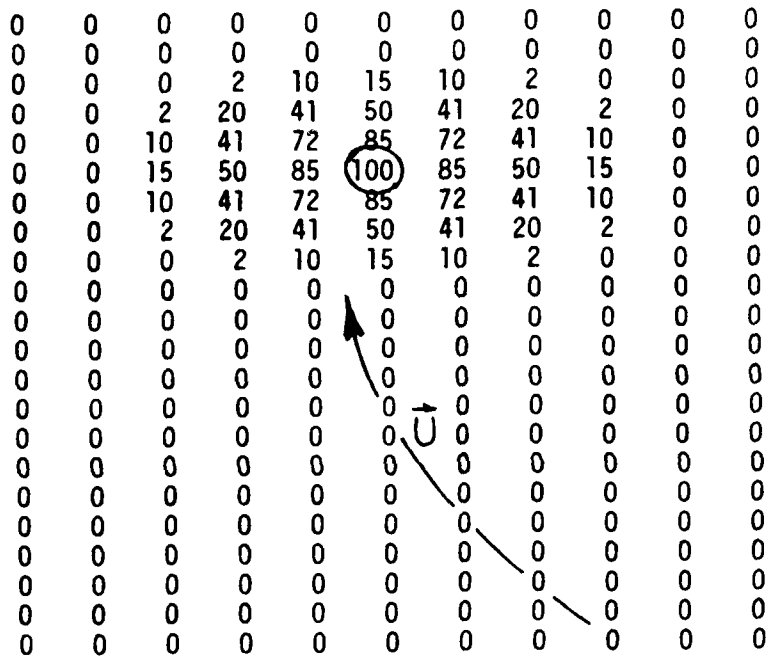


Figure 3. Advection of Cosine Hill in Solid Body Rotation, Initial Condition, Exact Final Solution

0	0	0	0	0	0	0	0	0	0	0
0	0	0	2	3	5	3	1	0	0	0
0	0	1	7	12	16	14	7	2	0	0
0	0	6	19	32	38	32	19	6	0	0
0	1	12	32	56	68	58	34	12	1	0
-1	2	15	44	77	93	80	46	15	1	0
-1	0	12	43	78	95	80	44	12	-1	0
-1	-2	4	26	54	69	56	26	2	-3	0
0	-4	-3	5	21	28	19	3	-5	-5	0
1	-2	-6	-7	-4	-3	-6	-9	-6	-1	2
2	1	-2	-6	-8	-9	-10	-7	0	3	1
1	3	2	0	0	-1	-1	0	4	3	0
0	2	2	3	2	4	4	3	2	0	0
0	0	0	1	1	1	2	1	0	-2	-1
0	-1	-1	-1	0	-1	-1	-1	-1	-1	0
0	0	-2	-2	9	0	-2	-1	0	0	1
0	0	0	-1	0	1	0	0	0	1	0
0	0	0	0	0	0	1	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	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0

a) $C = 0.25, v = 0$

0	0	0	0	2	2	1	0	0	0	0
0	0	1	2	4	5	5	2	1	0	0
0	0	3	8	13	15	13	8	3	0	0
0	2	7	15	26	33	28	18	6	1	0
0	2	11	28	46	57	51	31	12	1	0
0	3	15	39	66	82	72	44	16	1	0
-1	2	14	41	74	93	81	47	13	0	-1
-2	0	9	32	64	81	69	34	4	-4	-1
-2	-4	0	14	36	47	35	9	-6	-7	0
0	-5	-7	-4	4	9	0	-10	-13	-4	2
1	-2	-7	-11	-11	-12	-16	-17	-9	1	4
2	0	-2	-7	-10	-12	-11	-8	0	7	4
1	3	3	0	-1	-1	0	2	6	6	0
0	2	4	4	3	4	6	6	4	0	-2
-1	0	1	2	2	2	2	2	0	-2	-3
0	0	0	0	0	0	-1	-2	-2	-2	0
0	0	0	-1	-2	-1	-1	-2	-2	0	0
0	0	0	0	-2	-1	0	0	0	0	1
0	0	0	0	0	-2	0	2	1	0	0
0	0	0	0	0	-1	-1	0	1	0	-1
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0

b) $C = 0.5, v = 0$

Figure 4. Advection of Cosine Hill in Solid Body Rotation, Linear Tensor Product Algorithm

and $\nu = 0$, to Figure 4a) shows the noticeable degradation. The computed peak lags by one full mesh interval, and the associated dispersion error has reduced its level to 93. However, the solution symmetry about the indicated peak is quite good, (note dashed contour), and it would appear acceptable visually. The trailing dispersion error wake extremum is increased to -17, and the wavelength is nominally centered on $5\Delta_e$. For comparison, Figure 4c) shows the final solution as predicted using the Crank-Nicolson finite difference algorithm at $C = 0.25$. The result is much poorer than the linear element tensor product solution at twice the Courant number, i.e., half the computer CPU. The peak is only 47 and the extremum dispersion error level is -24. This popular algorithm is obtained by the elementary change of equation 52 to

$$\begin{bmatrix} C \\ \alpha \end{bmatrix}_{FD} \equiv \frac{\Delta_e^\alpha}{6} \begin{bmatrix} 3 & 0 \\ 0 & 3 \end{bmatrix} \quad (63)$$

As seen, the consequence is introduction of unacceptable error levels at modest Courant Number.

A prime requirement is to evaluate the efficiency of ν , the introduced dissipation parameter, to alter (improve) the error levels in these solutions. Elementary one-dimensional studies, references 16 - 17, have confirmed that the optimal linear analysis value of $\nu = 1/\sqrt{15} = 0.258$ is entirely too large. The loop test case is much more demanding, and confirmed the viability of separate and distinct value of ν , for use in the initial-value and convection matrices, see equations 52 and 53. Terming them correspondingly ν^i and ν^d , Table 1 summarizes the results of the cosine hill tests for the linear tensor product algorithm. All data are for the final station. For comparison, case 1 is the summary for the solution shown in Figure 4b, in terms of computed solution peak, displacement (in Δ_e) of the peak from its correct location, and fore/aft levels, hence symmetry about the peak. The dispersion wake error is quantized on extremum level and nominal wavelength of the dominant trailing wave. Using $\nu^d = 0.06$ alone, case 2, which amounts solely to an artificial diffusion, the wake error is decreased by two while the solution peak is decreased by 15% to 78. Introducing $\nu^i < \nu^d$

1	1	4	2	4	4	4	3	2	1	1
1	3	4	4	7	5	6	5	3	3	1
2	3	5	8	9	9	9	8	5	4	1
4	5	9	12	13	14	14	12	8	5	2
5	7	11	15	19	20	18	16	12	7	3
6	8	14	20	24	27	26	23	15	8	3
6	9	17	25	31	36	35	28	20	10	3
4	10	18	27	37	42	42	35	24	11	0
1	8	17	27	39	46	47	42	28	10	-2
-1	5	12	24	37	47	50	44	29	10	-6
-5	0	5	16	29	40	47	42	28	7	-11
-8	-4	-4	4	15	27	35	34	24	5	-13
-9	-8	-11	-7	0	9	17	20	15	1	-14
-8	-10	-15	-17	-14	-8	-2	2	1	-4	-12
-4	-9	-15	-20	-21	-21	-19	-14	-12	-12	-12
0	-5	-9	-15	-20	-23	-24	-23	-21	-17	-12
2	0	-2	-6	-9	-13	-17	-17	-17	-14	-9
2	2	2	1	1	0	0	-3	-4	-3	-1
0	2	3	4	8	10	12	13	13	11	11
0	0	1	3	6	12	16	18	16	14	12
0	0	0	0	1	5	11	17	19	16	7
0	0	0	0	0	0	0	0	0	0	0

c) $C = 0.25, \nu = 0$, Crank-Nicolson [C]

0	0	0	0	1	1	1	0	0	0	0
0	0	1	2	4	5	4	2	0	0	0
0	0	3	7	12	14	13	7	2	0	0
0	1	6	15	26	32	28	17	6	0	0
0	2	11	28	47	58	52	32	11	1	0
0	2	15	39	68	85	75	45	15	1	0
-1	1	14	43	78	98	85	48	13	-1	-1
-2	-1	8	34	67	86	71	34	3	-5	-1
-2	-5	-1	14	37	49	36	9	-8	-8	0
-1	-6	-9	-5	4	8	0	-13	-15	-5	2
1	-3	-8	-13	-13	-14	-18	-19	-10	1	4
2	1	-2	-8	-12	-13	-13	-9	0	7	4
1	4	3	0	-1	-1	0	3	7	7	1
0	3	4	5	4	5	7	7	5	1	-2
0	0	1	3	3	3	3	2	0	-3	-4
0	-1	0	0	0	0	-1	-2	-3	-3	-1
0	0	-1	-1	-2	-2	-1	-2	-2	0	0
0	0	0	0	-2	-1	0	0	0	1	1
0	0	0	0	0	-1	0	2	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	0	0	0	0	0	0	0	0

d) $C = 0.5, \nu^i = 0.012, \nu^d = 0.072$

Figure 4. Advection of Cosine Hill in Solid Body Rotation, Linear Tensor Product Algorithm (Concluded)

TABLE 1
SUMMARY OF COSINE HILL SENSITIVITY STUDY
LINEAR TENSOR PRODUCT ALGORITHM, C = 0.5

Case No.	Dissipation		Solution			Wake Dispersion Error	
	v^i	v^d	Peak	Displacement*	Symmetry	Peak	Wavelength*
1	.0	.0	93	-1	81/82	-17	4
2	.0	.06	78	-1	71/66	-9	5
3	.012	.06	102	-1	87/89	-21	6
4	.012	.072	98	-1	85/86	-19	6
5	.015	.06	110	-1	92/98	-26	6
6	.012	.09	93	-1	81/80	-16	6
7	.03	.06	117	-1	61/161	-74	**

*Units are Δ_e , the uniform discretization measure.

**No discernable dominant wavelength.

over a range, case Numbers 3 - 7 can remarkably increase the computed solution peak and affect solution symmetry as well as dispersion error. Case 4 corresponds to the "numerically optimized" combination, and the complete final solution station is shown in Figure 4d. Comparing to Figure 4b, and to the correct solution in Figure 3, the ν -modified solution is a uniform improvement except for the modest increase in wake error level. Importantly, the peak of 98 and the surrounding level near 85 are both substantial improvements. The ability of a small proportion of ν^i to totally alter the solution, and the pure diffusive character of ν^d , should be of considerable use and certainly requires a definitive study.

The same test sequence has been completed for the quadratic tensor product algorithm. Figure 5a is the base solution for $C = 0.25$ and $\nu = 0$. Comparing to Figure 3, it is an excellent approximation to the correct solution. The peak occurs exactly at the correct node, and the surrounding solution symmetry and levels are excellent. Furthermore, the extremum wake error level is only -2. On all assessment bases, the improvement over the linear tensor product results, see Figure 4a, and for the Crank-Nicolson results Figure 4c, is impressive. Equally impressive is the observation that only 1% additional computer storage was required and the increase in CPU was only 16% (due mainly to a pentadiagonal rather than tridiagonal Jacobian). Doubling the Courant number produced the results shown in Figure 5b), which are a nominal improvement over the linear tensor product results at $C = 0.25$, Figure 4a. Hence, comparable accuracy is attained at a 40% decrease in CPU. Finally, Figure 5c displays the results obtained using the quadratic diagonalized initial-value matrix equivalent of Crank-Nicolson. The results are uniformly poorer on all bases of comparison.

Table 2 summarizes the results of the ν sensitivity study for the quadratic tensor product algorithm. Since the non-dissipative results maintained a larger peak value, the algorithm can withstand somewhat larger levels of ν^d in comparison to the linear element form. Case 5 results were the "best" improvement and the entire solution is shown in Figure 5d). In comparison to the linear element case, the improvement is not quite as substantial (since the error requiring improvement was smaller), but nevertheless uniformly positive.

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

a) $C = 0.25, \nu = 0$

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

b) $C = 0.5, \nu = 0$

Figure 5. Advection of Cosine Hill in Solid Body Rotation, Quadratic Tensor Product Algorithm

0	0	0	1	1	2	0	0	0	0	0	0
0	0	1	1	4	7	6	4	0	0	0	0
0	0	4	9	14	16	14	10	4	0	0	-1
0	0	7	15	24	29	26	18	8	0	0	0
0	2	13	27	44	53	46	31	13	2	0	0
0	3	17	37	62	77	66	43	17	2	0	0
-1	1	16	39	67	85	74	46	16	-2	-1	0
-2	0	13	36	65	86	71	38	9	-8	-3	2
-3	-6	9	15	40	56	45	19	-4	-14	-3	4
-2	-7	-8	-5	10	21	11	-3	-16	-15	0	7
0	-4	-10	-14	-10	-5	-9	-15	-16	-8	3	7
2	-1	-7	-13	-18	-21	-18	-17	-10	1	9	6
4	4	2	-2	-7	-9	-7	-4	0	8	8	4
1	4	7	9	4	2	6	8	9	10	3	-1
0	1	3	8	6	5	8	7	6	3	-3	-6
-1	-1	0	3	2	1	3	1	0	-1	-5	-5
0	-1	-2	-1	-2	-3	-2	-4	-5	-3	-3	0
0	-1	0	-2	-4	-7	-4	-3	-4	-1	0	4
0	0	0	0	-2	-3	0	1	2	2	0	2
0	0	1	0	-1	-4	0	1	5	6	2	1
0	0	0	0	0	-2	-1	-1	1	2	0	-3
0	0	0	0	0	0	0	0	0	0	0	0

c) $C = 0.5, \nu = 0$, Finite Difference [C]

0	0	0	0	1	1	1	0	0	0	0	0
0	0	1	2	3	4	3	2	0	0	0	0
0	0	2	5	11	13	11	6	1	0	0	0
0	1	5	13	25	33	28	16	3	-1	0	0
0	1	9	25	51	66	58	34	9	0	0	0
0	1	13	39	76	99	85	49	14	0	0	0
0	0	12	43	81	101	82	44	12	0	0	0
-1	-2	7	33	63	74	54	25	4	-1	0	0
-1	-3	0	12	29	31	17	3	-2	-2	0	0
0	-3	-6	-4	0	-2	-6	-7	-5	-1	1	0
0	0	-5	-8	-8	-10	-10	-5	-2	0	0	0
0	2	0	-4	-5	-6	-4	0	2	1	0	0
0	1	2	0	0	0	1	2	2	0	0	0
0	0	1	2	2	3	3	2	0	-1	-1	0
0	0	0	0	0	0	0	0	-1	-1	0	1
0	0	-1	-1	-1	-1	-1	-1	0	0	1	0
0	0	0	0	0	0	0	0	0	1	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	0	0	0	0	0	0	0	0	0	0

d) $C = 0.5, \nu^i = 0.012, \nu^d = 0.09$

Figure 5. Advection of Cosine Hill in Solid Body Rotation, Quadratic Tensor Product Algorithm (Concluded)

TABLE 2
 SUMMARY OF COSINE HILL SENSITIVITY STUDY
 QUADRATIC TENSOR PRODUCT ALGORITHM, C = 0.5

Case No.	Dissipation		Solution			Wake Dispersion Error	
	v^i	v^d	Peak	Displacement*	Symmetry**	Peak	Wavelength*
1	.0	.0	99	-.2	-	- 9	5
2	.0	.06	69	~ 0	-	- 2	~0
3	.012	.06	128		-	-20	5
4	.012	.072	116		-	-14	5
5	.012	.09	101	-.2	-	-10	5
6	.015	.114	100	-.2	-	-10	5

*Units of Δ_e , the uniform discretization measure.

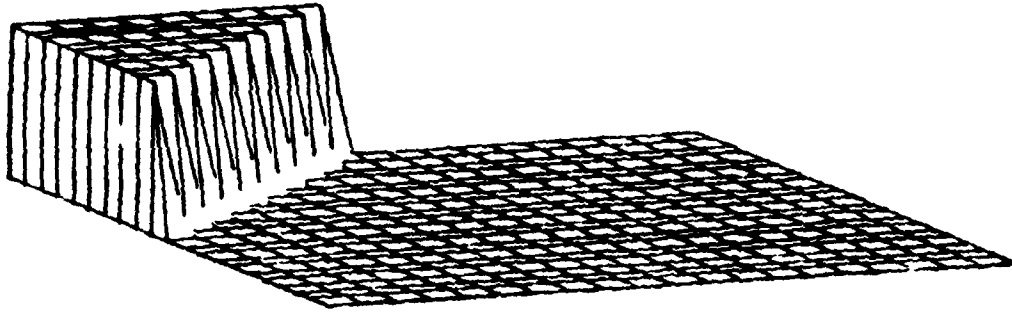
**Not directly evaluable from available data.

Comparison sensitivity studies have been essentially completed for the non-linear form of the model equation 62, i.e.,

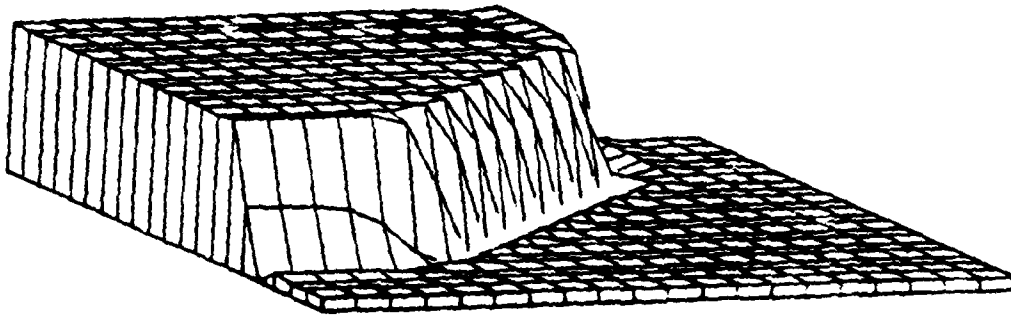
$$L(u) = \frac{\partial u}{\partial t} + \vec{U} \cdot \nabla u = 0 \quad (64)$$

where $\vec{U} = u\hat{i} + v\hat{j}$ and v satisfies an identical equation. The test case is the two-dimensional square wave propagating parallel to the principal diagonal of the solution domain. Figure 6a shows the initial condition, while Figure 6b shows the final solution obtained for $k = 1$, $v = v^i = v^d = 0.182$ and $C = 0.125$. This solution is a close approximation to the exact solution, wherein the dominant wave front remains interpolated across only one element, the trailing plateau remains at unity with no overshoot, and the solution gradient perpendicular to the left boundary vanishes. Increasing $v^d > 0.182$ levels the indicated overshoots but also diffuses the wave front across three or more Δ_e . The action of a separately evaluated $v^i \neq v^d$ in correcting this trend needs to be evaluated. Figure 6c is the comparison finite difference solution, which is essentially identical to that in reference 4, page 96. It is obtained by use of equation 63 instead of equation 52, and replacement of the v -modified term in equation 54 with an artificial viscosity term. The influence of the degraded selectivity of the diffusion operator, compare equations 59 and 60, is quite apparent. Again, the interpolation theory-derived algorithm appears superior to the equivalent complexity finite difference algorithm.

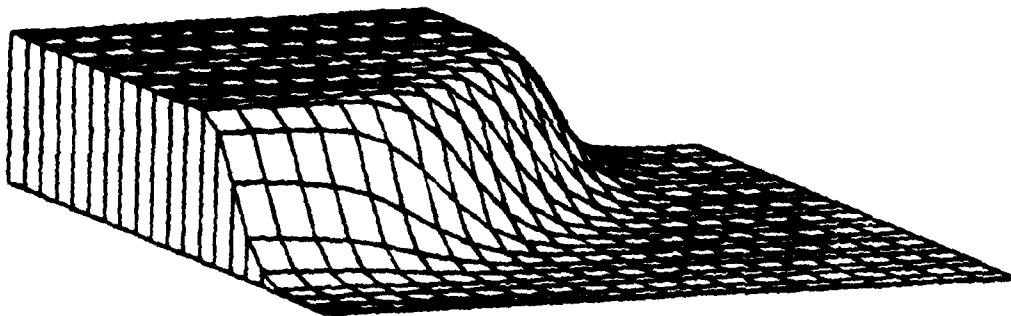
The action of $v > 0$ within the dissipative Weighted-Residuals theory is crucial. Specifically, Figure 7a shows the solution generated by the non-dissipative $k = 1$ algorithm, which has become completely nonsensical. Hence, it is imperative that the basic neutral stability be modified by $v > 0$ for a non-linear problem involving steep field gradients, e.g., shocks. The same holds for the quadratic embodiment of the algorithm. Figure 7b shows the final solution obtained for $k = 2$, $v^d = 0.182$, $v^i = 0$ and $C = 0.125$. Solution fidelity is comparable to that of the $k = 1$ solution, with a slight improvement in steepness of the dominant wave front. A sensitivity study for $v^i > 0$ should also be completed.



a) Initial Condition, $t = 0$

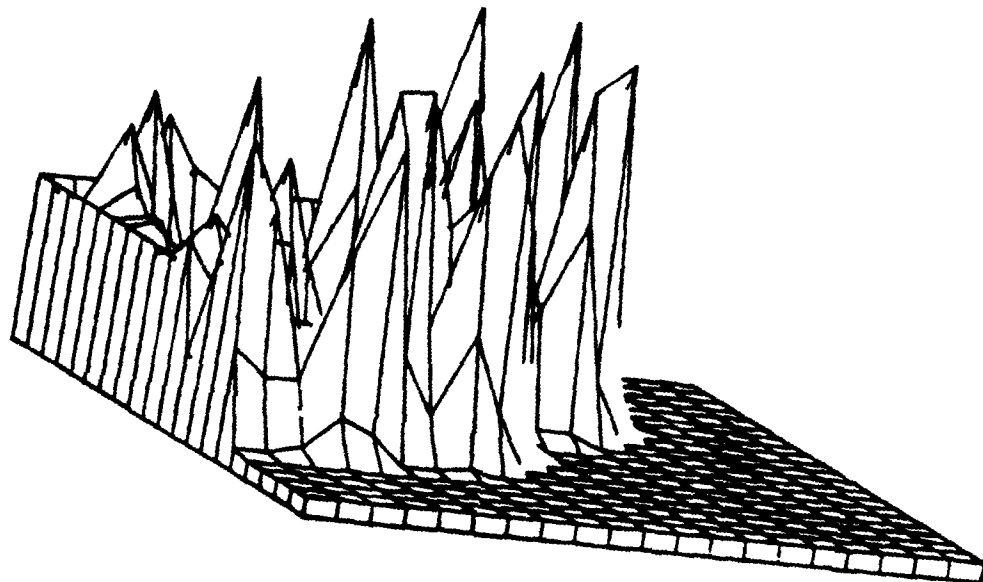


b) $n\Delta t = 120$, $v^i = v^d = 0.182$

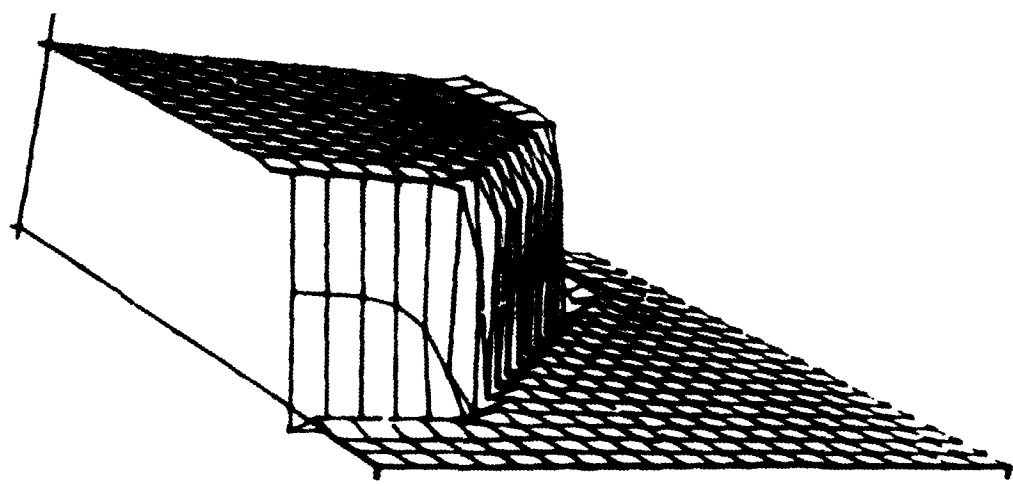


c) $n\Delta t = 120$, Finite Difference Form

Figure 6. Two-Dimensional Square Wave Convection,
Linear Tensor Product Algorithm, $C = 0.125$



a) $k = 1, n\Delta t = 120, \nu = 0$



b) $k = 2, n\Delta t = 120, \nu^i = 0, \nu^d = 0.182$

Figure 7. Two-Dimensional Square Wave Convection, $C = 0.125$

3. Matrix Iterative Solution Efficiency

The solutions discussed were generated employing the Newton iteration algorithm, equations 37 - 41, as embodied within the tensor product formulation, equation 45. This matrix solution algorithm can be rendered non-iterative by the elementary modifications to {F}, equation 39, of changing the sign of $h/2$ and removing all terms involving matrix products on $\{Q\}_{j+1}^P$. The Jacobian is unaltered. Most of the linear cosine-hill convection problems were solved using the non-iterative form. With a convergence requirement of 10^{-3} , the iterative algorithm required two iterations to produce solutions essentially identical to the direct solution. Hence, a CPU savings of 50% can be expected for a single scalar equation. The introduction of multiple dependent variables would introduce a block-banded structure into [J] that would moderate this basic advantage. The iterative form would employ multiple right-side back-substitutions into the LU decomposition of the basic tri- or pentadiagonal Jacobian [J]. At some point a trade-off might occur, and the test problem must correspond to solution of multiple equations.

In anticipation, the fully iterative non-linear algorithm has been evaluated for the turbulent boundary layer equation using both mixing length and turbulence kinetic energy-dissipation function turbulence closures. The fundamental question was whether an economical prediction technique, based upon finite differences, could yield a reduced number of iterations for this rather non-linear equation system. One procedure was the non-uniform-step extension of the finite difference method used by Soliman (ref. 17). A second order accurate estimate of q_{j+1}^1 is obtained using the leap-frog finite difference equation for non-uniform grid

$$\begin{aligned} q_{j+1}^1 = & q_j [1 - 2(1 + \eta) + (1 + \eta)^2] + q_j^1 [1 + \eta] \Delta_{j+1} \\ & + q_{j-1} [2(1 + \eta) - (1 + \eta)^2] \end{aligned} \quad (65)$$

where $\eta \equiv \Delta_{j+1}/\Delta_j$ is the non-uniformity of the integration step. A second-order accurate difference formula for $q_j^1 = q_j^1(q_j, q_{j-1}, q_{j-2})$ was employed.

Hence, this approach constitutes a second-order accurate leap-frog predictor, followed by the second-order accurate trapezoidal integration corrector. The intent is to accurately predict q_{j+1}^1 such that fewer corrections are needed for convergence. No equation solution is required but computer CPU cost is incurred from the finite difference evaluations.

The second predictor technique is conceptually opposite, in that no attempt is made to predict q_{j+1}^1 but instead a maximum evaluation of δq_{j+1}^1 is sought. Assuming the second and fourth terms in equation 39 are stored separately, and further that $\{f\}_{j+1} \approx \{f\}_j$, then specifying that $q_{j+1}^1 \equiv q_j$ eliminates the first term in equation 39 and the estimate of $\{F\}_{j+1}^1$ is constructed directly on q_j . This method is termed δQ -predict, requires a matrix solution of equation 37 but eliminates the element DO loop to form $\{F\}_{j+1}^1$, equation 39. The comparison method is straightforward application of the Newton algorithm. As with δQ -predict, $q_{j+1}^1 \equiv q_j$ but a computer program pass is made to correctly evaluate $\{g\}_{j+1}^1$, hence $\{F(Q_{j+1}^1)\}$.

The results are summarized in Table 3 and categorized with respect to predictor type and input solution parameters. Decreasing the convergence requirement to 10^{-4} , Tests 1-2, reduced the number of $\{F\}$ evaluations and matrix solves by about 40% and did not measurably affect accuracy, as measured in the shape factor norm. The accuracy of both predictor-type procedures was identical to the Newton solution as measured in both the energy and shape factor norms, Test 2. Surprisingly, the Q -predict using the finite difference formula did not reduce the number of F evaluations, i.e., $\{Q\}_{j+1}^1$ could not be predicted with sufficient accuracy. As a consequence, its solution CPU cost was maximum. The δQ -predict required fewer solution steps and $\{F\}$ evaluations, but more equation solutions ($207 + 80 = 287$ vs 272) than the Newton algorithm for a net 13% savings. Increasing the maximum τ -step by a factor of six does not change the overall solution accuracy, compare tests 2-3, nor the overall comparison among the solution methods. Hence, the Q -predict method can be discarded. For test 4, the maximum τ -step was reached earlier in the solution, and the increase in truncation error has decreased the solution energy by about a percent. Hence, the mathematical measure of the solution is degraded, but the engineering measure of

TABLE 3

MATRIX ITERATION EFFICIENCY SUMMARY

Test No.	Solution Parameters				Solution Efficiency			Solution Accuracy	
	Predictor Type	Iteration Convergence Required	Maximum τ -Step (cm)	No. of Solution Steps	No. of {F} Evaluations	Computer CPU	Solution Energy Final	Shape Factor Final	
1.	None Q	1.E(-5)	2.5	110	436	25.9	--	1.378	
				110	435	27.5	--	1.378	
2.	None Q δQ	1.E(-4)	2.5	110	272	15.4	121.28	1.378	
				100	270	17.1	121.29	1.378	
				80	207	13.7	121.29	1.377	
3.	None Q δQ	1.E(-4)	15.0	87	218	1.23	121.25	1.377	
				78	222	1.50	121.22	1.377	
				57	178	1.17	121.24	1.377	
4.	None δQ Mod.	1.E(-4)	15.0	37	129	.86	120.69	1.375	
				30	111	.91	120.68	1.374	

shape factor norm is negligibly different. Here, the δQ -predict was optimized to allow maximal early steps which reduced the total solution steps by seven to 30. However, the increased number of matrix solutions ($30 + 111 = 141$) reduced solution efficiency improvement to only 5%. Enhancing the no-predict procedure could reduce this modest superiority to zero; hence the δQ -predict method is only marginally attractive. Therefore, neither prediction method holds particular promise for economizing an iterative algorithm, and the storage and logic required for their execution can be deleted.

4. Resolution of Gradients, Shocks

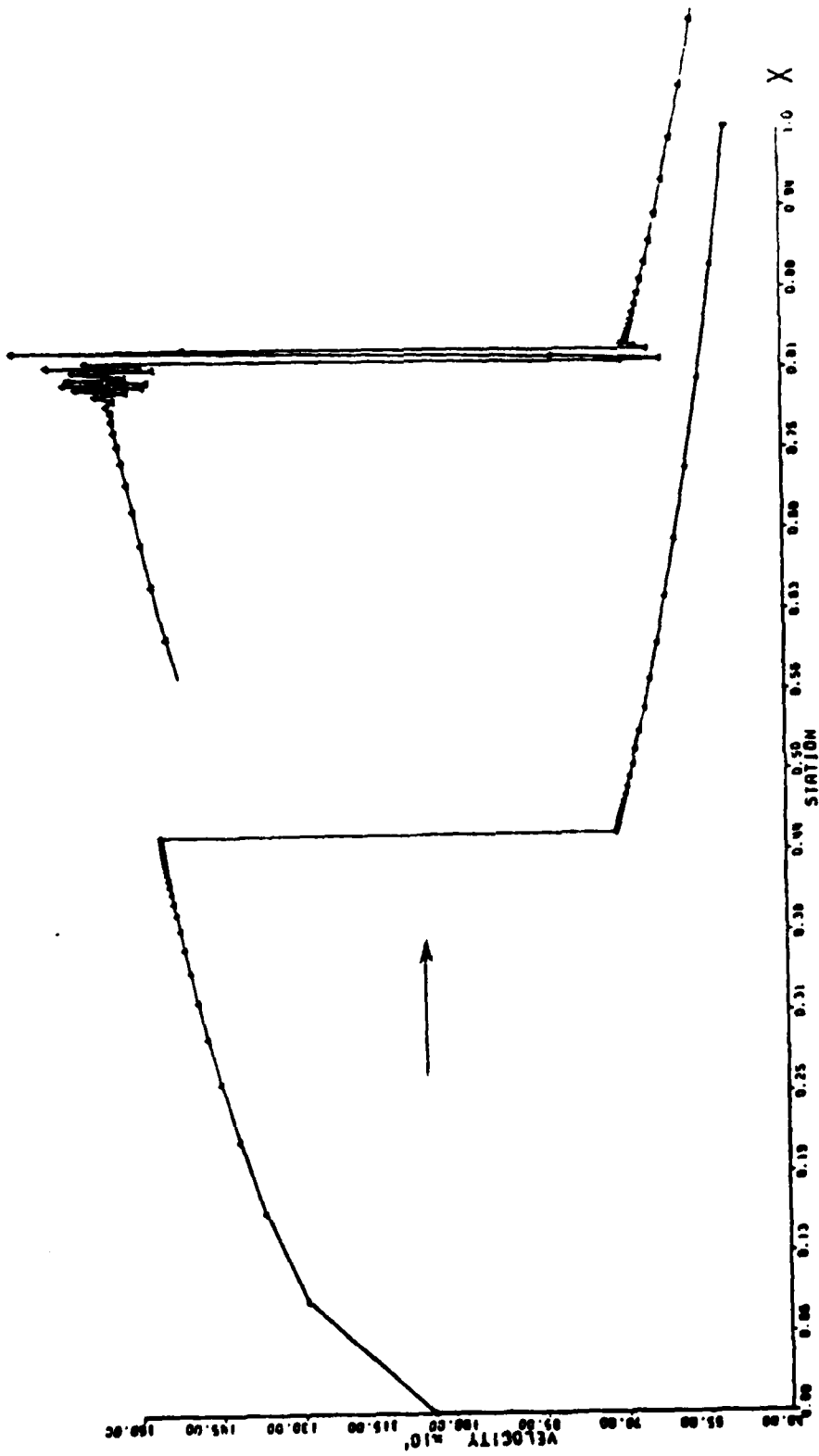
An important requirement is the capturing of imbedded flowfield shocks. The performance criteria for an algorithm include shock steepness, strength, and location. By and large, the addition of artificial diffusion to an Euler algorithm permits prediction of interpolated approximations of a shock. Use of conservative or non-conservative forms of an algorithm affects shock strength.

The results shown in Figures 6 - 7, for the two-dimensional Burger's equation, indicate differences in travelling wave front steepness for two forms of the tensor product algorithm in comparison to a finite difference algorithm. Additional insight can be obtained from examination of a solution of the Euler equations for one-dimensional shocked duct flow. The axial momentum equation in non-conservative form is

$$L(\rho u) = \rho \frac{\partial u}{\partial t} + \rho u \frac{\partial u}{\partial x} + \frac{\partial p}{\partial x} + \rho u^2 \frac{\partial \ln A}{\partial x} = 0 \quad (66)$$

where $A(x)$ is the duct cross-sectional area distribution. Figure 8a shows the initial distribution of $u(x)$ with a shock located in a region of fine discretization; each symbol corresponds to a node point. Figure 8b shows the u distribution resulting from application of a perturbation to $p(x)$ that moves the shock a modest distance upstream.

The influence of the levels of the dissipation parameters ν^i and ν^d , on the solution of $u(x,t)$ in the immediate vicinity of the perturbation have been studied. Particular interest focuses on the dispersion error amplitude



b) Perturbed Distribution

a) Initial Distribution

Figure 8. Initial and Pressure-Perturbed Velocity Distributions for One-Dimensional Supersonic Shocked Duct Flow

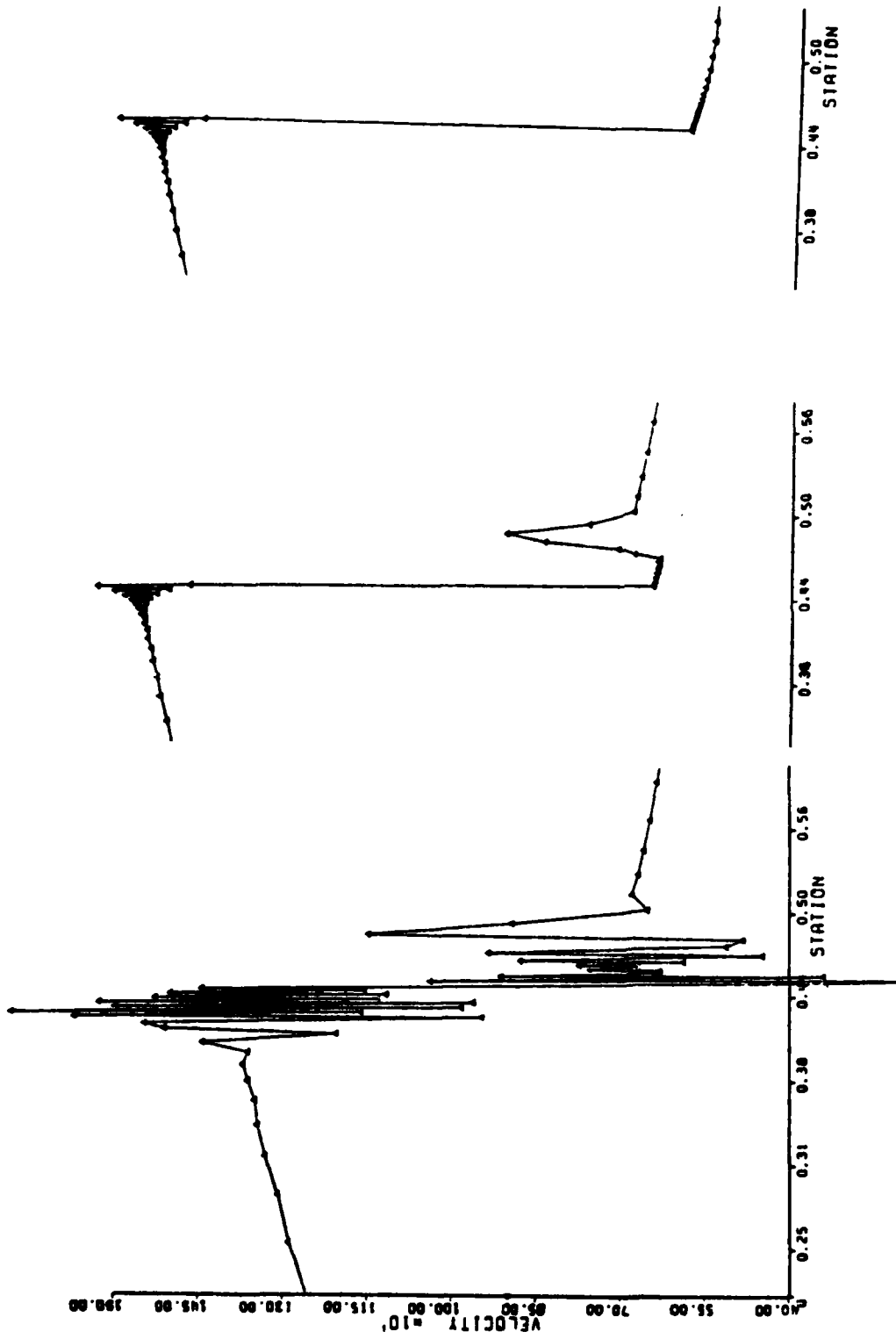
on the supersonic side of the shock, and on the decay of the downstream traveling wave. Figure 9a shows the preponderance of error generated by the non-dissipative form of the algorithm after twenty time steps. In contrast, for $v^i = v^d = 0.0645$, Figures 9b - 9c show the linear element solutions at 20 and 200 time steps. The dispersion error waveform on the supersonic side remains essentially stationary throughout, and the amplitude of the travelling wave decays rapidly.

The final shock strength is overpredicted by about 10%, and depends primarily on the level of v^d . Figure 10 illustrates the influence of various v^d for $v^i \equiv 0$ for the linear element algorithm at $n\Delta t = 20$. In comparison to Figure 10a, which is the correct solution, and the dashed curves, decreasing v^d in the range $0.2582 > v^d \geq 0.0645$ improves shock strength accuracy, while admitting a larger dispersion error envelope on the supersonic side and a diminished dissipation of the downstream travelling wave. Introduction of $v^i > 0$ has very little effect on shock strength or dispersion error envelope, but does introduce destructive interference into the travelling wave. Figure 11 illustrates the results obtained for the linear element algorithm for $v^d = 0.0645$, $0.0322 \leq v^i \leq 0.193$. Comparing Figure 11d with 10d, it appears that a further decrease in v^d to improve shock strength would be permissible.

5. Transformation to Arbitrary Domains

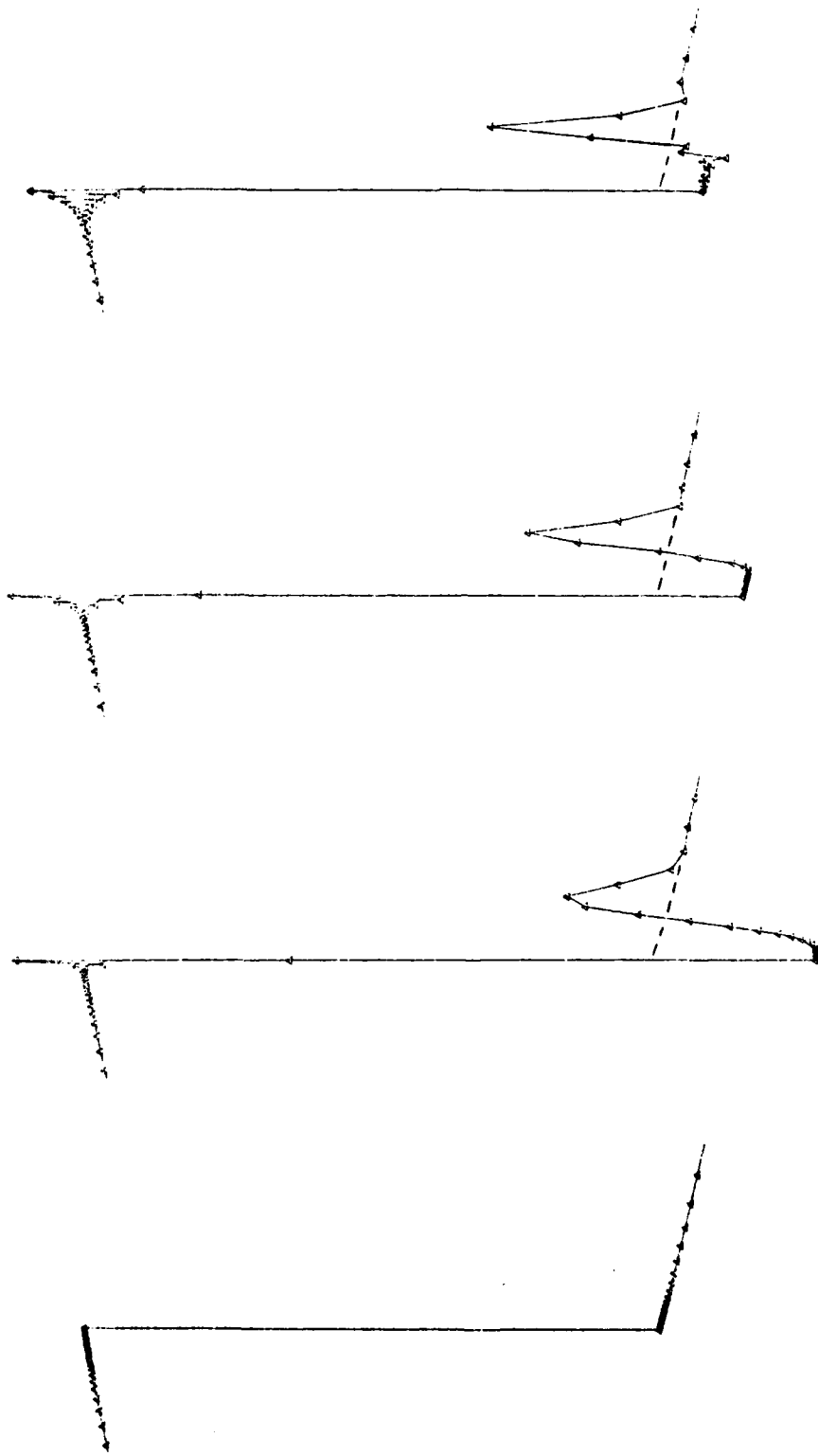
For the tensor product algorithm to be viable, it must be operable for general geometries. Geometric versatility was the original key feature of the finite element concept, but use of coordinate transformation to map non-regular shaped domains onto the unit square will supplant this. Thompson and co-workers, references 18 - 20, pioneered the concept of solution of non-self-adjoint Poisson equations to generate the mappings. An alternative procedure, using an analytical transformation on a local basis, appears most useful for the tensor product algorithm, while enjoying direct extension to three-dimensional space.

Figure 12 illustrates the concept for a straight-sided domain on two-dimensional space. The origin of the "natural" n_i coordinate system is defined at the centroid. The coordinate curves of n_i remain straight lines, but the system is not orthogonal in the \bar{x}_i plane. However, the n_i system



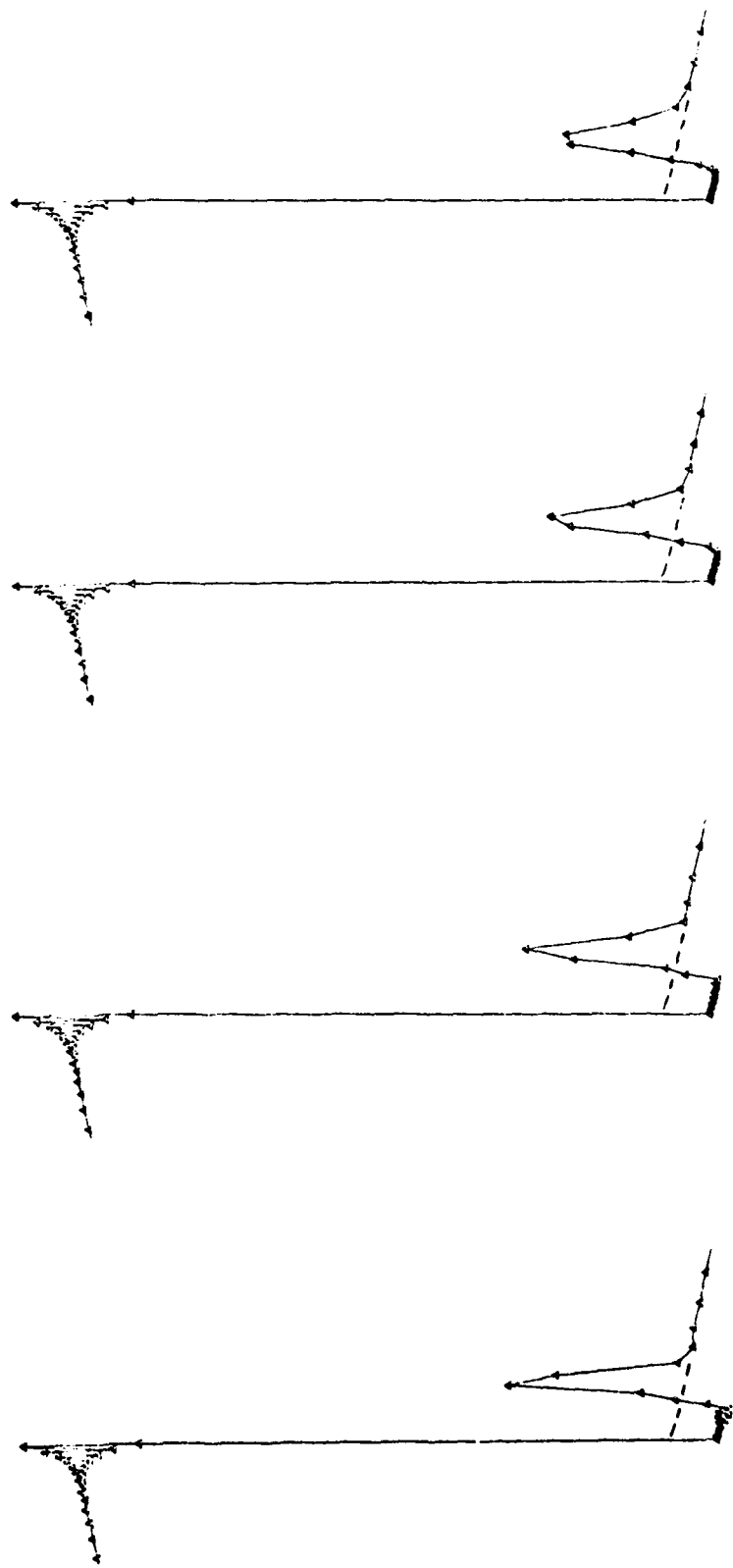
a) $n\Delta t = 20, v^i = v^d = 0$ b) $n\Delta t = 20, v^i = v^d = 0.0645$ c) $n\Delta t = 200, v^i = v^d = 0.0645$

Figure 9. Computed Duct Velocity Distributions Using Linear Element Algorithm



a) Analytical b) $v^d = 0.258$ c) $v^d = 0.129$ d) $v^d = 0.0645$

Figure 10. Effect of Dissipation Level v^d on Shocked Duct Flow Solution, $k = 1$, $v^i = 0$, $n\Delta t = 20$.



a) $v^i = 0.0322$ b) $v^i = 0.0645$ c) $v^i = 0.129$ d) $v^i = 0.193$

Figure 11. Effect of Dissipation Level v^i on Shocked Duct Flow Solution, $k = 1$, $v^d = 0.0645$, $n\Delta t = 20$

is defined as orthogonal and normalized in the transform plane. The tool for accomplishing this, on a local basis, is contained within the standard finite element isoparametric coordinate transformation $\{N_{1+}(\eta_i)\}$, see references 12 and 21, where

$$\{N_{1+}(\eta_i)\} = \frac{1}{4} \begin{Bmatrix} (1 - \eta_1)(1 - \eta_2) \\ (1 + \eta_1)(1 - \eta_2) \\ (1 + \eta_1)(1 + \eta_2) \\ (1 - \eta_1)(1 + \eta_2) \end{Bmatrix} \quad (67)$$

Employing the classical concept of finite element interpolation theory, the global \bar{x}_i coordinate system is interpolated on R_e^2 in terms of the global node coordinates $\{XI\}_e$ of the finite element domain as

$$\begin{aligned} \bar{x}_1 &= \{N_{1+}(\eta_i)\}^T \{X1\}_e \\ \bar{x}_2 &= \{N_{1+}(\eta_i)\}^T \{X2\}_e \end{aligned} \quad (68)$$

The tensor product algorithm requires establishment of the vector gradient, equations 1 - 3. Hence, using the chain rule

$$\begin{aligned} \frac{\partial}{\partial x_j} q_e(x_i) &= \frac{\partial}{\partial x_j} \{N_{1+}(\eta_i)\}^T \{Q\}_e \\ &= \left[\frac{\partial}{\partial \eta_i} \{N_{1+}(\eta_i)\}^T \frac{\partial \eta_i}{\partial x_j} \hat{e}_j \right] \{Q\}_e \end{aligned} \quad (69)$$

where \hat{e}_j are unit vectors of the x_j coordinate system. The elements of $\{N_{1+}(\eta_i)\}$ are readily differentiated by η_i , but the backwards transformation $\eta_i = \eta_i(x_j)$ is not available for formation of $\partial \eta_i / \partial x_j$. However, the forward transformation yields elements of the Jacobian $[J]$ as

$$[J] \equiv \begin{bmatrix} \frac{\partial x_1}{\partial \eta_1} & \frac{\partial x_2}{\partial \eta_1} \\ \frac{\partial x_1}{\partial \eta_2} & \frac{\partial x_2}{\partial \eta_2} \end{bmatrix} \quad (70)$$

The elements of $[J]$ are easy to form and are linear polynomials in η_i , ie.

$$[J] = \begin{bmatrix} a(1 - \eta_2) + b(1 + \eta_2) & , & c(1 - \eta_2) + d(1 + \eta_2) \\ \alpha(1 - \eta_1) + \beta(1 + \eta_1) & , & \gamma(1 - \eta_1) + \delta(1 + \eta_1) \end{bmatrix} \quad (71)$$

where a-d and α - δ are specified constants that depend solely on the global node coordinates $\{XJ\}_e$.

Equation 69 requires the inverse operation, ie.

$$[J]^{-1} = \begin{bmatrix} \frac{\partial \eta_1}{\partial x_1} & \frac{\partial \eta_2}{\partial x_1} \\ \frac{\partial \eta_1}{\partial x_2} & \frac{\partial \eta_2}{\partial x_2} \end{bmatrix} \quad (72)$$

From equation 71, and the definition of a matrix inverse,

$$[J]^{-1} = \frac{1}{\det [J]} \begin{bmatrix} \gamma(1 - \eta_1) + \delta(1 + \eta_1) & , & -\alpha(1 - \eta_1) - \beta(1 + \eta_1) \\ -c(1 - \eta_2) - d(1 + \eta_2) & , & a(1 - \eta_2) + b(1 + \eta_2) \end{bmatrix} \quad (73)$$

where $\det [J]$ is the determinant of the Jacobian, equation 71. Hence, since a-d and α - δ are known constants, equation 69 is formally evaluated as

$$\frac{\partial}{\partial x_j} q_e(x_i) = \hat{\epsilon}_j [J]^{-1} \frac{\partial}{\partial \eta_i} \{N_{1+}(\eta)\}^T \{Q\}_e \quad (74)$$

The finite element algorithm requires integral operations, see equation 25. The differential element becomes

$$\begin{aligned} d\tau = dx_1 dx_2 &= \frac{\partial x_1}{\partial \eta_j} \frac{\partial x_2}{\partial \eta_k} d\eta_j d\eta_k \\ &= \det [J] d\eta_1 d\eta_2 \end{aligned} \quad (75)$$

For the single vector operation within the tensor product algorithm, $\det [J]$ in equations 75 and 73 will cancel, leaving $[J]^{-1}$ in the form of polynomials. The unit vector $\hat{\epsilon}_j$ is contracted with the convection velocity; the inner product is an invariant and should be evaluated in the local η_i coordinate system.

No fundamental change is involved in going to curved-sided or three-dimensional element domains. The extension in concept is illustrated in Figure 13 for a two-dimensional general domain the x_j coordinate system is interpolated quadratically as

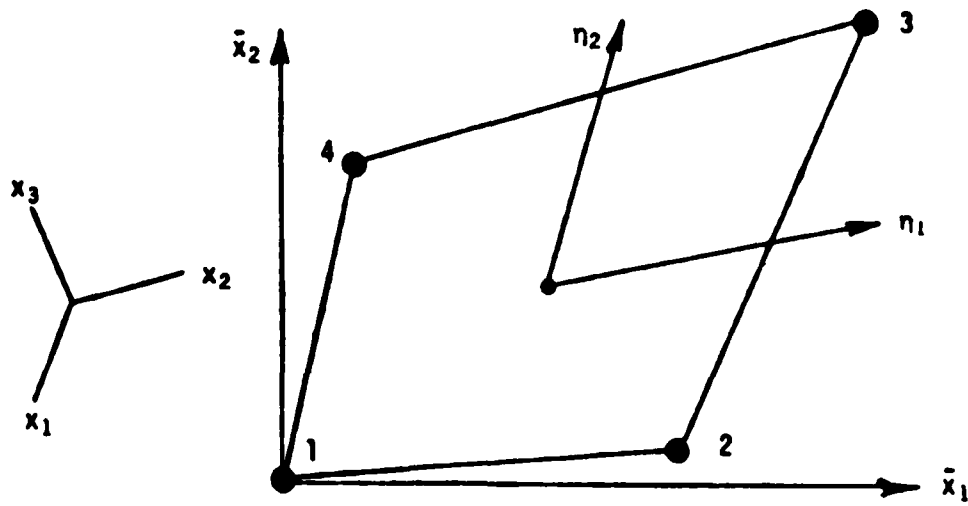


Figure 12. Plane-Quadrilateral Two-Dimensional Finite Element for $[N_1]$.

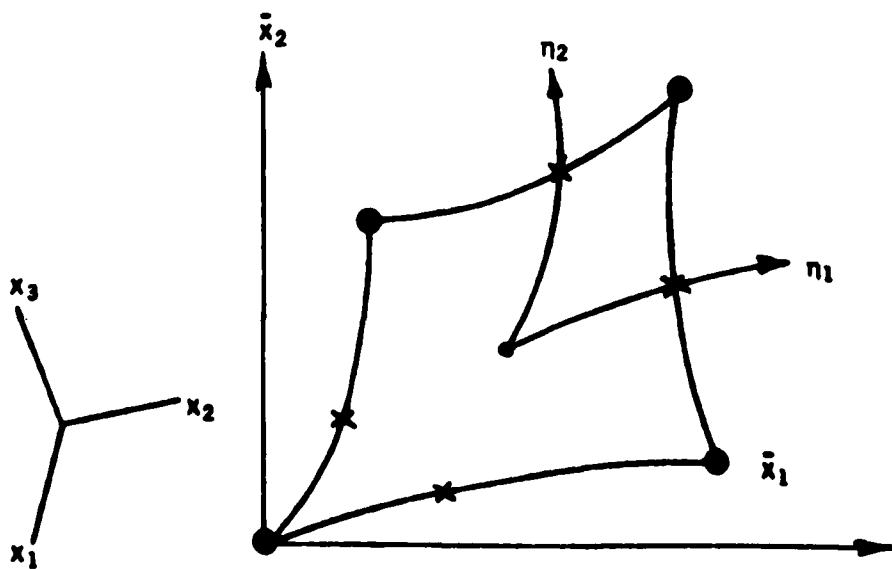


Figure 13. Generalized Quadrilateral Two-Dimensional Finite Element for $[N_2]$.

$$\begin{aligned}\bar{x}_1 &= \{N_{2+}(\eta_i)\}^T \{X1\}_e \\ \bar{x}_2 &= \{N_{2+}(\eta_i)\}^T \{X2\}_e\end{aligned}\quad (76)$$

where $\{X1\}$ contains eight elements equal to the respective global nodal coordinates. The matrix elements of the isoparametric quadrilateral basis are expressed in the η_i coordinate system as (ref. 21).

$$\{N_{2+}(\eta_i)\} = \frac{1}{4} \begin{pmatrix} (1 - \eta_1)(1 - \eta_2)(-\eta_1 - \eta_2 - 1) \\ (1 + \eta_1)(1 - \eta_2)(\eta_1 - \eta_2 - 1) \\ (1 + \eta_1)(1 + \eta_2)(\eta_1 + \eta_2 - 1) \\ (1 - \eta_1)(1 + \eta_2)(-\eta_1 + \eta_2 - 1) \\ 2(1 - \eta_1^2)(1 - \eta_2) \\ 2(1 + \eta_1)(1 - \eta_2^2) \\ 2(1 - \eta_1^2)(1 + \eta_2) \\ 2(1 - \eta_1)(1 + \eta_2^2) \end{pmatrix}\quad (77)$$

Then

$$\frac{\partial}{\partial x_j} q_e(x_i) = \hat{\epsilon}_j [J]^{-1} \frac{\partial}{\partial \eta_i} \{N_{2+}(\eta)\}^T \{Q\}_e\quad (78)$$

and the number of matrix elements in $\{Q\}_e$ now equals eight. Numerical integration procedures will undoubtedly be required to evaluate equation 25 in the tensor product form using equation 78.

SECTION IV

CONCLUSIONS AND RECOMMENDATIONS

The objective of this research project was to develop a highly accurate and efficient numerical solution algorithm for the non-linear three-dimensional Navier-Stokes equations in aerodynamics applications. A candidate algorithm has been derived employing finite element interpolation theory, the non-linear extension of quasi-variational principles, and the concept of tensor product bases. The resultant solution statement is rendered soluable using an implicit integration algorithm, and the replacement of the standard Jacobian of a Newton iteration algorithm with a tensor matrix product form.

A linearized stability analysis indicates the basic algorithm is spatially fourth-order accurate in its most elementary embodiment. The control of an added dissipation mechanism can elevate this to sixth order, but numerical experimentation indicates the resultant artificial diffusion is unacceptably large. By the same measure, this additional accuracy is intrinsic to the quadratic element embodiment of the algorithm with freedom from artificial diffusion. The results of several numerical experiments for single- and multi-dimensional convection-dominated test problems confirms the basic viability of the developed algorithm and its tensor product formulation. The latter is of paramount importance in rendering the algorithm nominally as efficient as familiar lower-order accurate finite difference formulations.

These results of the first year's research on the topic are highly encouraging. Specific recommendations include:

1. A computer program test bed should be constructed for solution of the multi-dimensional, multi-dependent variable Navier-Stokes equations. This program should contain both the linear and quadratic element embodiments for solution comparisons.
2. Evaluation of the transformation of arbitrarily shaped solution domains to a regular grid, using the isoparametric mapping, should be accomplished in this program.
3. The developed numerical procedures should be evaluated for solution of several Navier-Stokes problems for which accurate comparison results are available.

REFERENCES

1. "Future Computer Requirements for Computational Aerodynamics," NASA CP 2032, 1978.
2. Roache, P. J., Computational Fluid Mechanics, Hermosa Publishers, USA, 1972.
3. MacCormack, R. W., and Paullay, A. J., "Computational Efficiency Achieved by Time-Splitting of Finite Difference Operators," AIAA Paper No. 72-154, 1972.
4. Beam, R. M., and Warming, R. F., "An Implicit Finite-Difference Algorithm for Hyperbolic Systems in Conservation-Law Form," J. Comp. Phys. Vol. 22, No. 1, 1977.
5. Beam, R. M. and Warming, R. F., "An Implicit Factored Scheme for the Compressible Navier-Stokes Equations," AIAA J., V. 16, pp. 393-402, 1978.
6. Prenter, P. M., Splines and Variational Methods, John Wiley, New York, 1975.
7. Baker, A. J., and Soliman, M. O., "Utility of a Finite Element Solution Algorithm for Initial-Value Problems," J. Comp. Phys., V. 32, No. 3, pp. 289-324, 1979.
8. Cebeci, T., and Smith, A. M. O., Analysis of Turbulent Boundary Layers, Academic Press, New York, 1974.
9. Launder, B. E., Reece, G. J., and Rodi, W., "Progress in the Development of a Reynolds-Stress Turbulence Closure," J. Flu. Mech., Vol. 68, Pt. 3, pp. 537-566, 1975.
10. Lumley, J. L., "Toward a Turbulent Constitutive Relation," J. Flu. Mech., Vol. 41, Pt. 2, pp. 413-434, 1970.
11. Oden, J. T. and Reddy, J. N., An Introduction to the Mathematical Theory of Finite Elements, Wiley-Interscience, NY, 1976.
12. Baker, A. J., Finite Element Computational Fluid Mechanics, text manuscript, 1979.
13. Halmos, P. R., Finite Dimensional Vector Spaces, Van Nostrand, NY, 1958.
14. Yanenko, N. N., The Method of Fractional Steps, Springer-Verlag, NY, 1976.
15. Raymond, W. H. and Garder, A., "Selective Damping in a Galerkin Method for Solving Wave Problems with Variable Grids," Mon. Weather Rev., Vol. 104, pp. 1583-1590, 1976.

16. Baker, A. J., Soliman, M. O. and Pepper, D. W., "A Time-Split Finite Element Algorithm For Environmental Release Prediction," in Finite Elements In Water Resources, Pentech Press, London, pp. 4.53 - 4.65, 1978.
17. Soliman, M. O., "Accuracy and Convergence of a Finite Element Algorithm for Computational Fluid Dynamics," Ph.D. Dissertation, Univ. of Tennessee, Report ESM 78-1, 1978.
18. Thames, F. C., Thompson, J. F., and Mastin, C. W., "Numerical Solution of the Navier-Stokes Equations for Arbitrary Two-Dimensional Airfoils," NASA SP-347, 1975.
19. Thompson, J. F., Thames, F. C., and Mastin, C. W., "Boundary-Fitted Curvilinear Coordinate Systems for Solution of Partial Differential Equations on Fields Containing Any Number of Arbitrary Two-Dimensional Bodies," NASA CR-2729, 1976.
20. Thames, F. C., Thompson, J. F., Mastin, C. W., 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.
21. Zienkiewicz, O. C., The Finite Element Method, McGraw-Hill, London, 1977.

D
30