UC Berkeley
SEMM Reports Series

Title
Unconditionally Stable Element-by-Element Algorithms for Dynamic Problems

Permalink
https://escholarship.org/uc/item/4wb6j6xd

Authors
Ortiz, Miguel
Pinsky, Peter
Taylor, Robert

Publication Date
1982

eScholarship.org Powered by the California Digital Library
University of California

REPORT NO.
UCB/SESM-82/01

STRUCTURAL ENGINEERING AND
STRUCTURAL MECHANICS

UNCONDITIONALLY STABLE ELEMENT-BY-ELEMENT ALGORITHMS FOR DYNAMIC PROBLEMS

by

MIGUEL ORTIZ

PETER M. PINSKY

and

ROBERT L. TAYLOR

JANUARY 1982

DEPARTMENT OF CIVIL ENGINEERING
UNIVERSITY OF CALIFORNIA
BERKELEY, CALIFORNIA

UNCONDITIONALLY STABLE ELEMENT-BY-ELEMENT ALGORITHMS FOR DYNAMIC PROBLEMS

Miguel Ortiz, Peter M. Pinsky and Robert L. Taylor

Division of Structural Engineering and Structural Mechanics
Department of Civil Engineering
University of California, Berkeley

ABSTRACT

A collection of results is presented regarding the consistency, stability and accuracy of operator split methods and product formula algorithms for general nonlinear equations of evolution. These results are then applied to the structural dynamics problem, exploiting the element-by-element additive decomposition of the discrete dynamic equations resulting from a finite element discretization. Unconditionally stable algorithms are obtained that involve only element coefficient matrices. The storage requirements and operation counts are comparable to those of explicit methods. The method places no restriction on the topology of the finite element mesh.

January 25, 1982

UNCONDITIONALLY STABLE ELEMENT-BY-ELEMENT ALGORITHMS FOR DYNAMIC PROBLEMS

Miguel Ortiz, Peter M. Pinsky and Robert L. Taylor

Division of Structural Engineering and Structural Mechanics

Department of Civil Engineering

University of California, Berkeley

1. Introduction

Explicit algorithms in structural dynamics have the desirable property that they eliminate both the need of storing a large coefficient matrix and the equation-solving computational effort. However, explicit algorithms are conditionally stable. This stability condition is particularly stringent in stiff systems, thus severely limiting the usefulness of the method.

Alternatively, implicit algorithms such as the trapezoidal rule are second order accurate and unconditionally stable, the choice of time step thus being solely restricted by accuracy considerations. The main drawback of these methods is the fact that they give rise to large systems of simultaneous equations. This situation is particularly burdensome within the context of nonlinear analysis where the coefficient matrix has to be formed and triangularized many times during the integration process.

In the past, a number of attempts have been directed towards developing algorithms that decrease the storage and equation-solving requirements that are common to implicit schemes while retaining the unconditional stability property. A promising possibility is furnished by the so called operator split methods and product formula algorithms. These approximation techniques have been known to mathematicians for several decades, and have been successfully applied to both theoretical and numerical problems in semigroup theory and in various areas of

mathematical physics [1,2,3]. The suggestion that these methods can be used in computational mechanics as well seems to have first appeared in the Russian literature [4,5,6]. Recently, these techniques have been successfully applied to the study of the heat conduction problem [7]. Also, the partitioned analysis method for coupled systems [8,9,10] can be viewed as an example of application of these ideas.

In the present paper, a collection of results is first presented regarding operator split methods and product formula algorithms for general nonlinear equations of evolution. These results illustrate the point that product formulas can be advantageously applied to any set of equations of evolution where the evolutionary operator has an additive decomposition (operator split) into several, hopefully simpler, component operators. The basic idea underlying product formulas is that of treating each one of the component operators independently. In a typical integration process, one applies an algorithm to the solution vector that is consistent with the first component operator, the result of which is then operated upon with an algorithm which is consistent with the second component operator, and so on. It is shown herein that the global or product algorithm so obtained is consistent with the complete evolution operator. Furthermore, it is shown that if the individual algorithms are unconditionally stable so is the resulting global algorithm. Finally, a double pass technique is discussed that allows the construction of second order accurate product algorithms from second order accurate individual algorithms. Some of these results have also been discussed in [7] for the particular case of linear equations of evolution and the trapezoidal rule.

The rest of the paper is devoted to a specific application of these techniques, namely an element-by-element split approach to finite element dynamic problems. The basic idea in the method is to exploit the element-by-element additive decomposition of the discrete dynamic equations resulting from a finite element discretization. This additive decomposition appears naturally when velocities and stresses are taken as unknowns. Applying the general product formula techniques to this specific case, unconditionally stable algorithms are obtained that involve only element coefficient matrices. The resulting data base requirements are identical to

typical explicit methods and operation counts are considerably smaller than the conventional implicit methods. Second order accuracy is obtained when second order accurate element algorithms such as the trapezoidal rule are used. Finally, the method places no topological restrictions on the finite element mesh.

Numerical examples are presented in Section 5 that illustrate the characteristics of the method.

2. General Product Algorithms Based on Operator Splits of the Equations of Evolution

Often in the numerical treatment of engineering problems one is led to consider equations of evolution of the following general form

$$\mathbf{A}\dot{\mathbf{x}} + \mathbf{B}(\mathbf{x}) = \mathbf{f} \ ; \quad \mathbf{x}(0) = \mathbf{x}_o \tag{1}$$

a particular case of which is the unforced equation

$$\mathbf{A}\dot{\mathbf{x}} + \mathbf{B}(\mathbf{x}) = 0 \ ; \quad \mathbf{x}(0) = \mathbf{x}_o \tag{2}$$

In the above, $\mathbf{x}$ is an s-dimensional vector, $\mathbf{A}$ is a positive definite symmetric matrix and $\mathbf{B}$ is a nonlinear function from R^s into R^s.

It is useful for the subsequent discussion to endow R^s with the following "energy" inner product

$$<\mathbf{x},\mathbf{y}> = \mathbf{x}^T\mathbf{A}\mathbf{y} \tag{3}$$

for every $\mathbf{x},\mathbf{y} \in R^s$, with its associated norm

$$||\mathbf{x}||^2 = <\mathbf{x},\mathbf{x}>. \tag{4}$$

In this context, an unconditionally stable algorithm for equation (2) is a one-parameter family of (nonlinear) functions $\mathbf{F}(h) : R^s \rightarrow R^s$, $h > 0$, satisfying

1) Consistency:

$$\lim_{h\rightarrow 0^+} \mathbf{A}\frac{\mathbf{F}(h)\mathbf{x} - \mathbf{x}}{h} = \mathbf{B}(\mathbf{x}) \quad \textit{for every} \quad \mathbf{x} \in R^s \tag{5}$$

2) Unconditional stability:

$$||\mathbf{F}(h)\,\mathbf{x} - \mathbf{F}(h)\,\mathbf{y}|| \leqslant ||\mathbf{x} - \mathbf{y}|| \quad \textit{for every} \quad \mathbf{x},\mathbf{y} \in R^s, \quad h>0. \tag{6}$$

In the linear case, the mapping $\mathbf{F}(h)$ is linear in $\mathbf{x}$ and the stability condition (6) reduces to the simpler form

$$||\mathbf{F}(h)\,\mathbf{x}|| \leqslant ||\mathbf{x}|| \quad \textit{for every} \quad \mathbf{x} \in R^s \tag{7}$$

This in turn implies that the norm $||\mathbf{F}(h)|| \leqslant 1$. Recalling the well-known inequality relating the spectral radius $\rho(\mathbf{F}(h)) \equiv \inf_n ||\mathbf{F}(h)^n||^{1/n}$ to the norm of $\mathbf{F}(h)$

$$\rho(\mathbf{F}(h)) \leqslant ||\mathbf{F}(h)|| \tag{8}$$

it may be seen that the stability condition (6) is in general more stringent than the usual concept of stability that requires $\rho(\mathbf{F}(h)) \leqslant 1$. If $\mathbf{F}(h)$ is a stable algorithm for (2) in the sense of (5) and (6) then convergence is guaranteed under mild conditions on $\mathbf{B}$ [12].

In a variety of problems in mechanics the evolutionary operator $\mathbf{B}$ and the forcing term $\mathbf{f}$ admit an additive decomposition

$$\mathbf{B} = \sum_{i=1}^{N} \mathbf{B}_i \quad ; \quad \mathbf{f} = \sum_{i=1}^{N} \mathbf{f}_i. \tag{9}$$

We are concerned here with the problem of constructing a class of computationally efficient algorithms that exploit the additive form of $\mathbf{B}$ and $\mathbf{f}$. For instance, in Section 4 a natural decomposition resulting from a finite element spatial discretization is studied in detail.

Let $\mathbf{F}_i(h)$, $i = 1,2,\cdots,N$ denote stable algorithms consistent with $\mathbf{A}$ and $\mathbf{B}_i$. The corresponding global product algorithm then takes the form

$$\mathbf{F}(h) = \mathbf{F}_N(h)\,\mathbf{F}_{N-1}(h) \cdots \mathbf{F}_1(h) \equiv \prod_{i=1}^{N} \mathbf{F}_i(h) \tag{10}$$

In other words, the algorithm $\mathbf{F}(h)$ amounts to applying the individual algorithms $\mathbf{F}_i(h)$ consecutively to the solution vector, taking the result from each one of these applications as the initial conditions for the next algorithm. The global algorithm is complete for a given time step when all the individual algorithms have been applied.

We next show that the global product algorithm so defined is in fact consistent with $\mathbf{A}$ and $\mathbf{B}$, and that it is unconditionally stable if the individual algorithms $\mathbf{F}_i(h)$ are.

Proposition 1. The product algorithm $\mathbf{F}(h)$ defined in (10) is consistent with $\mathbf{A}$ and $\mathbf{B}$.

Proof. For the sake of generality, the forced case is considered in this proof. The fact that the individual operators $\mathbf{F}_i(t)$ are consistent with $\mathbf{A}$ and $\mathbf{B}_i$ implies

$$\mathbf{F}_i(h)\,\mathbf{x} = \mathbf{x} + h\,\mathbf{A}^{-1}(-\mathbf{B}_i(\mathbf{x}) + \mathbf{f}_i) + O(h^2), \qquad i=1,2,\cdots,N$$

Taking the product of $\mathbf{F}_1(h)$ and $\mathbf{F}_2(h)$ and retaining terms up to second order we obtain

$$\begin{aligned}
&\mathbf{F}_2(h)\,\mathbf{F}_1(h)\,\mathbf{x} \equiv \mathbf{F}_2(h)(\mathbf{F}_1(h)\,\mathbf{x}) = \\
&\mathbf{F}_1(h)\,\mathbf{x} - h\,\mathbf{A}^{-1}\mathbf{B}_2(\mathbf{F}_1(h)\,\mathbf{x}) + h\,\mathbf{A}^{-1}\mathbf{f}_2 + O(h^2) = \\
&\mathbf{x} - h\,\mathbf{A}^{-1}(\mathbf{B}_1 + \mathbf{B}_2)(\mathbf{x}) + h\,\mathbf{A}^{-1}(\mathbf{f}_1 + \mathbf{f}_2) + O(h^2)
\end{aligned}$$

Proceeding by induction it is readily shown that

$$\begin{aligned}
&\mathbf{F}(h)\,\mathbf{x} = \left(\prod_{i=1}^{N}\mathbf{F}_i(h)\right)\mathbf{x} = \\
&\mathbf{x} - h\,\mathbf{A}^{-1}\left(\sum_{i=1}^{N}\mathbf{B}_i\right)(\mathbf{x}) + h\,\mathbf{A}^{-1}\left(\sum_{i=1}^{N}\mathbf{f}_i\right) + O(h^2) = \\
&\mathbf{x} - h\,\mathbf{A}^{-1}\mathbf{B}(\mathbf{x}) + h\,\mathbf{A}^{-1}\mathbf{f} + O(h^2)
\end{aligned}$$

which proves that $\mathbf{F}(h)$ is consistent with $\mathbf{A}$ and $\mathbf{B}$.

Proposition 2. The global product algorithm $\mathbf{F}(h)$ is unconditionally stable if all the individual algorithms $\mathbf{F}_i(h)$ are.

Proof. It follows from the definition (6) of unconditional stability that for every $\mathbf{x},\mathbf{y} \in R^s$ and $h>0$

$$\begin{aligned}
&||\mathbf{F}(h)\,\mathbf{x} - \mathbf{F}(h)\,\mathbf{y}|| = \\
&\left|\left|\left(\prod_{i=1}^{N}\mathbf{F}_i(h)\right)\mathbf{x} - \left(\prod_{i=1}^{N}\mathbf{F}_i(h)\right)\mathbf{y}\right|\right| = \\
&\left|\left|\mathbf{F}_N(h)\left(\prod_{i=1}^{N-1}\mathbf{F}_i(h)\right)\mathbf{x} - \mathbf{F}_N(h)\left(\prod_{i=1}^{N-1}\mathbf{F}_i(h)\right)\mathbf{y}\right|\right| \leqslant \\
&\left|\left|\left(\prod_{i=1}^{N-1}\mathbf{F}_i(h)\right)\mathbf{x} - \left(\prod_{i=1}^{N-1}\mathbf{F}_i(h)\right)\mathbf{y}\right|\right|
\end{aligned}$$

Proceeding by induction one finds

$$||\mathbf{F}(h)\,\mathbf{x} - \mathbf{F}(h)\,\mathbf{y}|| \leqslant ||\mathbf{x} - \mathbf{y}||$$

which proves the unconditional stability of the global product algorithm.

This last result can be stated by saying that norm stability of the individual algorithms, in the sense of (6), is preserved by the product formula (10). It is interesting to note, on the other hand, that no general statements can be made about the stability of the product formula

in the sense of the spectral ratio, given that, unlike the norm, this quantity is not well-behaved with respect to matrix multiplications.

3. Second Order Accuracy and the Double Pass Technique

In many practical situations, second order accuracy is very desirable. The question we address in this section is under what circumstances second order accurate algorithms $\mathbf{F}_i(h)$ result in global second order accuracy.

Let us first examine the "single pass" product algorithm (10). For $\mathbf{F}(h)$ to be second order accurate it has to agree with the exact solution $\mathbf{x}(t)$ up to second order terms in the Taylor expansion

$$\mathbf{x}(t) = \mathbf{x}_o + \dot{\mathbf{x}}_o\, t + \frac{1}{2}\ddot{\mathbf{x}}_o\, t^2 + \cdots \tag{11}$$

But solving for $\dot{\mathbf{x}}$ in (2) it follows that

$$\dot{\mathbf{x}}_o = -\mathbf{A}^{-1}\mathbf{B}(\mathbf{x}_o) \tag{12}$$

Also, differentiating (1) one gets

$$\mathbf{A}\ddot{\mathbf{x}} = -\frac{d}{dt}\mathbf{B}(\mathbf{x}) = -\mathbf{DB}(\mathbf{x})\,\dot{\mathbf{x}} \tag{13}$$

and therefore

$$\ddot{\mathbf{x}}_o = -\mathbf{A}^{-1}\mathbf{DB}(\mathbf{x}_o)\,\dot{\mathbf{x}}_o = \mathbf{A}^{-1}\mathbf{DB}(\mathbf{x}_o)\,\mathbf{A}^{-1}\mathbf{B}(\mathbf{x}_o)$$

where $\mathbf{DB}(\mathbf{x}_o)$ denotes the derivative of $\mathbf{B}$ at $\mathbf{x}_o$.

Therefore, a second order accurate algorithm has to satisfy:

$$\mathbf{F}(h)\,\mathbf{x} = \mathbf{x} - h\,\mathbf{A}^{-1}\mathbf{B}(\mathbf{x}) + \frac{h^2}{2}\mathbf{A}^{-1}\mathbf{DB}(\mathbf{x})\,\mathbf{A}^{-1}\mathbf{B}(\mathbf{x}) + O(h^3) \tag{14}$$

Assume now that all the individual algorithms $\mathbf{F}_i(h)$ in (10) are second order accurate, i.e.,

$$\mathbf{F}_i(h)\,\mathbf{x} = \mathbf{x} - h\,\mathbf{A}^{-1}\mathbf{B}_i(\mathbf{x}) + \frac{h^2}{2}\mathbf{A}^{-1}\mathbf{DB}_i(\mathbf{x})\,\mathbf{A}^{-1}\mathbf{B}_i(\mathbf{x}) + O(h^3)$$

Taking the first two terms in the product algorithm (10) and expanding up to third order terms it follows that

$$\begin{aligned}
&\mathbf{F}_2(h)\,\mathbf{F}_1(h)\,\mathbf{x} \equiv \mathbf{F}_2(h)(\mathbf{F}_1(h)\,\mathbf{x}) = \\
&\mathbf{F}_1(h)\,\mathbf{x} - h\,\mathbf{A}^{-1}\mathbf{B}_2(\mathbf{F}_1(h)\,\mathbf{x}) + \frac{h^2}{2}\mathbf{A}^{-1}\mathbf{DB}_2(\mathbf{F}_1(h)\,\mathbf{x})\,\mathbf{A}^{-1}\mathbf{B}_2(\mathbf{F}_1(h)\,\mathbf{x}) + O(h^3) = \\
&\mathbf{x} - h\,\mathbf{A}^{-1}(\mathbf{B}_1 + \mathbf{B}_2)(\mathbf{x}) + \frac{h^2}{2}[\mathbf{A}^{-1}\mathbf{DB}_1(\mathbf{x})\mathbf{A}^{-1}\mathbf{B}_1(\mathbf{x}) + \mathbf{A}^{-1}\mathbf{DB}_2(\mathbf{x})\mathbf{A}^{-1}\mathbf{B}_2(\mathbf{x})] + \\
&h^2\mathbf{A}^{-1}\mathbf{DB}_2(\mathbf{x})\mathbf{A}^{-1}\mathbf{B}_1(\mathbf{x}) + O(h^3)
\end{aligned} \tag{15}$$

Proceeding by induction, one arrives to the following expression:

$$\left(\prod_{i=1}^{N}\mathbf{F}_i(h)\right)\mathbf{x} = \mathbf{x} - h\,\mathbf{A}^{-1}(\sum_{i=1}^{N}\mathbf{B}_i)(\mathbf{x}) + \frac{h^2}{2}[\sum_{i=1}^{N}\mathbf{A}^{-1}\mathbf{D}\mathbf{B}_i(\mathbf{x})\mathbf{A}^{-1}\mathbf{B}_i(\mathbf{x})] + h^2[\sum_{i=1}^{N-1}\sum_{j=i+1}^{N}\mathbf{A}^{-1}\mathbf{D}\mathbf{B}_j(\mathbf{x})\,\mathbf{A}^{-1}\mathbf{B}_i(\mathbf{x})] + O(h^3) \tag{16}$$

Using the identity

$$\sum_{i=1}^{N}\mathbf{A}^{-1}\mathbf{D}\mathbf{B}_i(\mathbf{x})\,\mathbf{A}^{-1}\mathbf{B}_i(\mathbf{x}) = \mathbf{A}^{-1}[\sum_{i=1}^{N}\mathbf{D}\mathbf{B}_i(\mathbf{x})]\mathbf{A}^{-1}[\sum_{i=1}^{N}\mathbf{B}_i(\mathbf{x})] - \sum_{i=1}^{N-1}\sum_{j=i+1}^{N}\mathbf{A}^{-1}\mathbf{D}\mathbf{B}_i(\mathbf{x})\mathbf{A}^{-1}\mathbf{B}_j(\mathbf{x}) - \sum_{i=1}^{N-1}\sum_{j=i+1}^{N}\mathbf{A}^{-1}\mathbf{D}\mathbf{B}_j(\mathbf{x})\mathbf{A}^{-1}\mathbf{B}_i(\mathbf{x})$$

arising from the expansion of the product $\mathbf{A}^{-1}[\sum_{i=1}^{N}\mathbf{D}\mathbf{B}_i(\mathbf{x})]\;\mathbf{A}^{-1}[\sum_{i=1}^{N}\mathbf{B}_i(\mathbf{x})]$, eq. (16) becomes:

$$\left(\prod_{i=1}^{N}\mathbf{F}_i(h)\right)\mathbf{x} = \mathbf{x} - h\,\mathbf{A}^{-1}\mathbf{B}(\mathbf{x}) + \frac{h^2}{2}\mathbf{A}^{-1}\mathbf{D}\mathbf{B}(\mathbf{x})\mathbf{A}^{-1}\mathbf{B}(\mathbf{x}) + \frac{h^2}{2}\sum_{i=1}^{N-1}\sum_{j=i+1}^{N}[\mathbf{A}^{-1}\mathbf{D}\mathbf{B}_j(\mathbf{x})\,\mathbf{A}^{-1}\mathbf{B}_i(\mathbf{x}) - \mathbf{A}^{-1}\mathbf{D}\mathbf{B}_i(\mathbf{x})\,\mathbf{A}^{-1}\mathbf{B}_j(\mathbf{x})] + O(h^3) \tag{17}$$

Comparing this result to eq. (14), we see that the single pass product algorithm is not second order accurate, as a result of the last term appearing in (17).

A technique that can be used to circumvent this problem is that of a "double pass" product algorithm:

$$\mathbf{F}(h)\,\mathbf{x} = \prod_{i=N}^{1}\mathbf{F}_i(h/2)\prod_{i=1}^{N}\mathbf{F}_i(h/2)\,\mathbf{x} \tag{18}$$

In a double pass product algorithm one first applies the individual algorithms from 1 to N with half the time step and then in reverse order, also with half the time step. If we now expand (18) as for the single pass case, one readily finds that the spurious terms drop out due to the fact that the contributions from the two passes cancel each other and one gets

$$\mathbf{F}(h)\,\mathbf{x} \equiv \prod_{i=N}^{1}\mathbf{F}_i(h/2)\prod_{i=1}^{N}\mathbf{F}_i(h/2)\,\mathbf{x} = \mathbf{x} - h\,\mathbf{A}^{-1}\mathbf{B}(\mathbf{x}) + \frac{h^2}{2}\mathbf{A}^{-1}\mathbf{D}\mathbf{B}(\mathbf{x})\,\mathbf{A}^{-1}\mathbf{B}(\mathbf{x}) + O(h^3) \tag{19}$$

which proves the second order accuracy of the double pass product algorithm.

4. Element-by-Element Product Algorithms for Dynamic Problems

The nature of the method is illustrated by means of an example, namely the dynamic problem for a linear elastic solid occupying a region Ω in R^n. In this case, the governing equations read:

$$\begin{aligned} \rho\,\dot{\mathbf{v}} &= \nabla\cdot\boldsymbol{\sigma} + \rho\,\mathbf{b} \\ \dot{\boldsymbol{\sigma}} &= \mathbf{D}:\nabla\mathbf{v} \end{aligned} \tag{20}$$

with boundary conditions

$$\begin{aligned} \mathbf{v} &= \bar{\mathbf{v}} \quad on\ \partial\Omega_u \\ \boldsymbol{\sigma}\cdot\mathbf{n} &= \bar{\mathbf{t}} \quad on\ \partial\Omega_\sigma \end{aligned} \tag{21}$$

where $\partial\Omega_u \bigcup \partial\Omega_\sigma = \partial\Omega$ and $\partial\Omega_u \bigcap \partial\Omega_\sigma = \varnothing$, and initial conditions

$$\begin{aligned} \mathbf{v}(0) &= \mathbf{v}_o \\ \boldsymbol{\sigma}(0) &= \boldsymbol{\sigma}_o \end{aligned} \tag{22}$$

In eq. (20), $\mathbf{v}$ and $\boldsymbol{\sigma}$ denote the velocity and stress fields over Ω, $\mathbf{b}$ the body forces, ρ the density and $\mathbf{D}$ the elastic modulus tensor, with the usual symmetries. The symbol $:$ implies the contraction $\mathbf{D}:\nabla\mathbf{v} \equiv D_{ijkl}\,v_{k,l}$. In eq. (21), $\bar{\mathbf{v}}$ and $\bar{\mathbf{t}}$ denote the prescribed velocities and tractions over the kinematic and traction boundaries $\partial\Omega_u$ and $\partial\Omega_\sigma$ respectively.

Eqs. (20) to (22) define an initial boundary value problem for the unknowns $\begin{Bmatrix}\mathbf{v}\\ \boldsymbol{\sigma}\end{Bmatrix}$. The equation $\dot{\mathbf{u}} = \mathbf{v}$ can be used to obtain the displacements from the velocities. It should be emphasized, however, that the choice of velocities and stresses as unknown variables is crucial for the success of the method.

In practice, a spatial discretization technique such as the finite element method is commonly used to obtain approximate solutions. The derivation of the discretized finite element equations of motion in terms of nodal displacements is a standard exercise and can be found elsewhere in the literature [11]. However, as will become apparent below, the application of the product formula techniques to this specific problem requires phrasing the equations motion as a set of first order ordinary differential equations in terms of velocities and stresses. For completeness, a brief account of the derivation of this form of the discretized equations of motion is given next.

A weak form of the linear momentum balance eq. is given by

$$\int_{\Omega} [\rho\,(\dot{\mathbf{v}} - \mathbf{b})\cdot\boldsymbol{\eta} + \boldsymbol{\sigma}:\nabla\boldsymbol{\eta}]\,d\Omega = \int_{\partial\Omega_\sigma} \bar{\mathbf{t}}\cdot\boldsymbol{\eta}\,d\Gamma \tag{23}$$

for all weighting functions $\boldsymbol{\eta}$ which satisfy the homogeneous boundary conditions on $\partial\Omega_u$. Introducing a set of global finite element interpolation functions ϕ_a, $a = 1,2,\cdots,N$, the interpolated velocity and weighting functions take the form

$$\begin{aligned} \mathbf{v} &= \mathbf{v}^a \phi_a \\ \boldsymbol{\eta} &= \boldsymbol{\eta}^a \phi_a \end{aligned} \tag{24}$$

where the summation convention is implied and $\mathbf{v}^a$ and $\boldsymbol{\eta}^a$ denote the nodal values of $\mathbf{v}$ and $\boldsymbol{\eta}$, respectively. The global interpolation functions ϕ_a are given by the expression

$$\phi_a = \sum_{e=1}^{Nel} \phi_a^e \tag{25}$$

where the index e ranges over the elements and ϕ_a^e denotes the element interpolation functions. Substituting (24) and (25) into (23) one obtains the following spatially discretized weak form of the linear momentum balance equations

$$M_{ab}\,\dot{\mathbf{v}}^b + \sum_{e=1}^{Nel} \int_{\Omega^e} \boldsymbol{\sigma}\cdot\mathbf{B}_a^e\,d\Omega = \sum_{e=1}^{Nel} \mathbf{f}_a^e = \mathbf{f}_a \tag{26}$$

where $\mathbf{B}_a^e = \nabla\phi_a^e$ and

$$\begin{aligned} M_{ab} &= \sum_{e=1}^{Nel} \int_{\Omega^e} \rho\,\phi_a^e \phi_b^e\,d\Omega \\ \mathbf{f}_a^e &= \int_{\Omega^e} \rho\,\mathbf{b}\,\phi_a^e\,d\Omega + \int_{\partial\Omega_\sigma^e} \bar{\mathbf{t}}\phi_a^e\,d\Gamma \end{aligned} \tag{27}$$

where $\mathbf{f}_a^e$ is the contribution to the global force vector associated with element e.

The integral in eq. (26) can be evaluated using numerical quadrature, leading to

$$M_{ab}\,\dot{\mathbf{v}}^b + \sum_{e=1}^{Nel} \sum_{k=1}^{M} w^{e,k}\,\boldsymbol{\sigma}^{e,k}\cdot\mathbf{B}_a^{e,k} = \sum_{e=1}^{Nel} \mathbf{f}_a^e = \mathbf{f}_a \tag{28}$$

where k ranges over the integration points, $w^{e,k}$ denotes the integration weights associated with element e and $\boldsymbol{\sigma}^{e,k}$ and $\mathbf{B}_a^{e,k}$ denote the values of $\boldsymbol{\sigma}$ and $\mathbf{B}_a^e$ at the integration points, respectively. A complete set of discretized equations of evolution is finally obtained by sampling the constitutive equations (20.b) at the integration points * and making use of the finite element

* An alternative is to use a mixed form where interpolations for stresses are introduced in each element and

interpolation (24.a) to obtain

$$\dot{\boldsymbol{\sigma}}^{e,k} = \mathbf{D}^{e,k} : \mathbf{B}_a^{e,k} \mathbf{v}^a \tag{29}$$

The unknowns of the discretized problem (28), (29) can be conveniently arranged in an array

$$\mathbf{x} = \begin{Bmatrix} \mathbf{v}^1 \\ \cdot \\ \cdot \\ \cdot \\ \mathbf{v}^N \\ \boldsymbol{\sigma}^{1,1} \\ \cdot \\ \cdot \\ \cdot \\ \boldsymbol{\sigma}^{Nel,M} \end{Bmatrix} \tag{30}$$

Denoting

$$\mathbf{A} = \begin{bmatrix} \mathbf{M} & 0 \\ 0 & \mathbf{I} \end{bmatrix} \; ; \quad \mathbf{B}\,\mathbf{x} = \begin{Bmatrix} -\sum_{e=1}^{Nel} \sum_{k=1}^{M} w^{e,k} \boldsymbol{\sigma}^{e,k} \cdot \mathbf{B}_a^{e,k} \\ \mathbf{D}^{e,k} : \mathbf{B}_a^{e,k} \mathbf{v}^a \end{Bmatrix} \; ; \quad \bar{\mathbf{f}} = \begin{Bmatrix} \mathbf{f}_a \\ 0 \end{Bmatrix} \tag{31}$$

The evolution eqs. (28), (29) can be more compactly expressed as

$$\mathbf{A}\,\dot{\mathbf{x}} = \mathbf{B}\,\mathbf{x} + \bar{\mathbf{f}} \tag{32}$$

The evolution of the system determined by (32) takes place in R^s, $s = N + M \times Nel$. For the study of the problem at hand it proves convenient to introduce in R^s the following "energy inner product"

$$\langle \mathbf{x}_1, \mathbf{x}_2 \rangle \equiv M_{ab} \mathbf{v}_1^a \cdot \mathbf{v}_2^b + \sum_{e=1}^{Nel} \sum_{k=1}^{M} w^{e,k} \boldsymbol{\sigma}_1^{e,k} : \mathbf{C}^{e,k} : \boldsymbol{\sigma}_2^{e,k} \tag{33}$$

where $\mathbf{C}^{e,k} = [\mathbf{D}^{e,k}]^{-1}$. The quantity

$$\frac{1}{2} ||\mathbf{x}||^2 \equiv \frac{1}{2} \langle \mathbf{x}, \mathbf{x} \rangle \tag{34}$$

is in fact the total energy of the system, which motivates the term "energy inner product" used above. In the unforced case, the rate of energy change of the system is given by

$$\frac{d}{dt} \left(\frac{1}{2} ||\mathbf{x}||^2 \right) = \langle \dot{\mathbf{x}}, \mathbf{x} \rangle = \langle \mathbf{A}^{-1} \mathbf{B}\,\mathbf{x}, \mathbf{x} \rangle \tag{35}$$

---

a weak form of the constitutive equations (20.b) is used [13].

Proposition 3. The solutions of the initial value problem (28) and (29) preserve energy.

Proof. By the definition of **A** and **B** and of the energy inner product, it follows that

$$
\begin{aligned}
&\langle \mathbf{A}^{-1}\mathbf{B}\,\mathbf{x},\mathbf{x}\rangle = \\
&-M_{ab}[(M^{-1})^{ac}\sum_{e=1}^{Nel}\sum_{k=1}^{M} w^{e,k}\boldsymbol{\sigma}^{e,k}\cdot\mathbf{B}_c^{e,k}]\cdot\mathbf{v}^b + \sum_{e=1}^{Nel}\sum_{k=1}^{M} w^{e,k}(\mathbf{D}^{e,k}:\mathbf{B}_a^{e,k}\mathbf{v}^a):\mathbf{C}^{e,k}:\boldsymbol{\sigma}^{e,k} = \\
&-\sum_{e=1}^{Nel}\sum_{k=1}^{M} w^{e,k}\boldsymbol{\sigma}^{e,k}:\mathbf{B}_a^{e,k}\mathbf{v}^a + \sum_{e=1}^{Nel}\sum_{k=1}^{M} w^{e,k}\boldsymbol{\sigma}^{e,k}:\mathbf{B}_a^{e,k}\mathbf{v}^a = 0
\end{aligned}
$$

Hence, by eq. (35), $\frac{d}{dt}\left(\frac{1}{2}||\mathbf{x}||^2\right) = 0.$

We next note that the evolution operator **B** and forcing term $\bar{\mathbf{f}}$ in (32) admit the following additive decomposition

$$
\mathbf{B} = \sum_{e=1}^{Nel}\mathbf{B}_e \quad ; \qquad \bar{\mathbf{f}} = \sum_{e=1}^{Nel}\bar{\mathbf{f}}_e \tag{36}
$$

This provides an specific example of the situation expressed in (9) and suggests the possibility of using product formulas based on this decomposition. Here, the individual element operators $\mathbf{B}_e$ and forcing terms $\bar{\mathbf{f}}_e$ are

$$
\mathbf{B}_e\mathbf{x} = \begin{Bmatrix}
-\sum_{k=1}^{M} w^{e,k}\boldsymbol{\sigma}^{e,k}\cdot\mathbf{B}_c^{e,k} \\
0 \\
\vdots \\
0 \\
\mathbf{D}^{e,1}:\mathbf{B}_a^{e,1}\mathbf{v}^a \\
\vdots \\
\mathbf{D}^{e,M}:\mathbf{B}_a^{e,M}\mathbf{v}^a \\
0 \\
\vdots \\
0
\end{Bmatrix} \quad ; \quad \bar{\mathbf{f}}_e = \begin{Bmatrix} \mathbf{f}_a^e \\ 0 \end{Bmatrix} \tag{37}
$$

Note that this decomposition only involves element quantities.

Proposition 4. The solutions of the equation

$$\mathbf{A}\dot{\mathbf{x}} = \mathbf{B}_e\,\mathbf{x} \tag{38}$$

preserve energy.

Proof. It follows from the definition of $\mathbf{B}_e$ that

$$\frac{d}{dt}\left(\frac{1}{2}||\mathbf{x}||^2\right) = <\mathbf{A}^{-1}\mathbf{B}_e\,\mathbf{x},\mathbf{x}> =$$

$$-M_{ab}[(M^{-1})^{ac}\sum_{k=1}^{M} w^{e,k}\boldsymbol{\sigma}^{e,k}\cdot\mathbf{B}_c^{e,k}]\cdot\mathbf{v}^b + \sum_{k=1}^{M} w^{e,k}(\mathbf{D}^{e,k}:\mathbf{B}_a^{e,k}\mathbf{v}^a):\mathbf{C}^{e,k}:\boldsymbol{\sigma}^{e,k} =$$

$$-\sum_{k=1}^{M} w^{e,k}\boldsymbol{\sigma}^{e,k}:\mathbf{B}_a^{e,k}\mathbf{v}^a + \sum_{k=1}^{M} w^{e,k}\boldsymbol{\sigma}^{e,k}:\mathbf{B}_a^{e,k}\mathbf{v}^a = 0$$

It is interesting to note that if the mass matrix is diagonal eqs. (38) only affect the degrees of freedom that are attached to element e. This being the case, the integration of these equations can be carried out locally, at the element level.

The solution of the individual eqs. (38) can be numerically obtained by the application of an algorithm $\mathbf{F}_e(h)$, such as the trapezoidal rule

$$\mathbf{F}_e(h) = (\mathbf{A} - \frac{h}{2}\mathbf{B}_e)^{-1}\cdot(\mathbf{A} + \frac{h}{2}\mathbf{B}_e) \tag{39}$$

It is again noted that this algorithm is applied only to the degrees of freedom associated with the element e under consideration, provided the mass matrix is diagonal.

A global algorithm $\mathbf{F}(h)$ which is consistent with eqs. (28) and (29) can be then obtained from the individual algorithms $\mathbf{F}_e(h)$ by means of the product formula (10), which in this case takes the form

$$\mathbf{F}(h) = \prod_{e=1}^{Nel}\mathbf{F}_e(h) \tag{40}$$

In practice, the above product formula requires looping over all the elements in the structure. For each element, the trapezoidal rule is applied based on element coefficient matrices, which results in an update of the global degrees of freedom associated with the element under consideration. The next element is then considered and the same operation is repeated on the updated solution vector resulting from the previous element and so on, until all the elements have been traversed.

Given that, by Proposition 4, the individual operators $\mathbf{B}_e$ are energy preserving, the trapezoidal rule algorithm $\mathbf{F}_e(h)$ associated with $\mathbf{B}_e$ is also energy preserving. As a result, the proof of Proposition 2 shows that the global product algorithm $\mathbf{F}(h)$ preserves energy, as the exact solution does.

We also recall that the trapezoidal rule is a second order accurate algorithm. This property can be retained globally through the use of the double pass formula (18), which in this case takes the form

$$\mathbf{F}(h) = \prod_{e=Nel}^{1} \mathbf{F}_e(h/2) \prod_{e=1}^{Nel} \mathbf{F}_e(h/2) \tag{41}$$

The application of this double pass algorithm entails the same computational operations as for the single pass one described above which in this case are carried out twice, with half the time step, traversing the elements from first to last and vice versa.

We finally remark that the present formalism can be easily extended to include damping, provided that the damping matrix has an element-by-element additive decomposition into positive definite element matrices. The consistent, mass-proportional and stiffness-proportional damping matrices all exhibit these properties.

5. Numerical Examples

A test example is presented that has been designed to illustrate the strong points and limitations of the method. The problem under consideration is defined in Fig. 1. The material properties are such that that a very stiff initial value problem results and the connectivity of the elements is chosen so as to test the ability the method to deal with arbitrary mesh topologies.

All the results shown correspond to an exact solution of the element equations of motion (38). Exact integration was adopted to allow an assessment of the accuracy of the product formula by eliminating the truncation error associated with the element algorithms. When exact integration is replaced by a second order accurate element algorithm, such as the trapezoidal rule, our numerical experiments have indicated that the results obtained are similar to those reported here. The direct results of the integration process are the nodal velocities and axial

forces, while the nodal displacements may be computed separately from the velocities. The trapezoidal rule

$$\mathbf{u}_{n+1} = \mathbf{u}_n + \frac{h}{2}(\mathbf{v}_n + \mathbf{v}_{n+1}) \tag{42}$$

was employed for this purpose.

Single pass results. Figs. 2 to 4 show single pass results corresponding to the case (i) in Fig. 1, in which the three degrees of freedom are subjected to an equal initial displacement. Fig. 2 shows the displacement and velocity solutions at node 1 corresponding to a time step $h = 0.01$, which is slightly under the forward-Euler critical time step for the system, $t_{cr} = 0.014$. It is seen that both solutions are virtually exact, which illustrates the convergence properties of the element-by-element method. The total energy of the system was exactly preserved throughout all single pass and double pass solutions. This illustrates the unconditional stability of the method. Results for time steps over t_{cr} are shown in Figs. 3 and 4. The curves appear jagged due to the influence of the high frequency components, although the disturbances introduced by them are seen to remain small. A significant period shift is observed to occur for the higher time steps while the amplitude of the solutions remains very close to the exact value.

Double-pass results. Figs. 5 to 7 illustrate the gain in accuracy achieved through the use of the double pass technique. To this effect, Fig.6 may be compared to the single pass result for the same time step, Fig. 4. Even for a time step of 0.2, which is about 14 times greater than t_{cr}, the accuracy of the method appears to be satisfactory. In particular, the period of the solution is obtained almost exactly.

Fig. 8 illustrates one of the limitations of the method. The results shown correspond to an initial displacement of node 3, as in (ii), Fig. 1. In this case, the high frequency components of the system are given a high energy content. It is observed that the accuracy of the solution deteriorates for time steps that are over t_{cr}. Given that the energy of the system is exactly preserved, this effect may be attributed to the energy transfer between modes. Although this situation is not frequently encountered in practice, it nevertheless suggests the

need for further research.

Acknowledgements

The authors would like to thank Prof. T.J.R. Hughes for helful discussions.

Grants for the partial support of this work from the Lawrence Livermore National Laboratory and General Motors Research Laboratories are gratefully acknowledged.

References

1. A. Pazy, **Semi-Groups of Linear Operators and Applications to Partial Differential Equations,** Lecture Note #10, Dept. of Mathematics, University of Maryland, 1974.

2. H. Brézis, **Opérateurs Maximaux Monotones et Semi-Groupes de Contractions dans les Espaces de Hilbert,** North-Holland, 1973.

3. V. Barbu, **Nonlinear Semigroups and Differential Equations in Banach Spaces,** Noordhoff, 1976.

4. N.N. Yanenko, **The Method of Fractional Steps,** Springer-Verlag, 1971.

5. G.I. Marchuk, **Numerical Methods in Weather Prediction,** Academic Press, 1974.

6. G.I. Marchuk, **Methods of Numerical Mathematics,** Springer-Verlag, 1975.

7. T.J.R. Hughes, I. Levit and J. Winget, "Unconditionally Stable Element-by-Element Implicit Algorithms for Heat Conduction Analysis", **Computer Methods in Applied Mechanics and Engineering,** (to appear).

8. K.C. Park, "Partitioned Transient Analysis Procedures for Coupled-Field Problems: Stability Analysis", **Journal of Applied Mechanics,** Vol. 47, 1980, pp. 370-376.

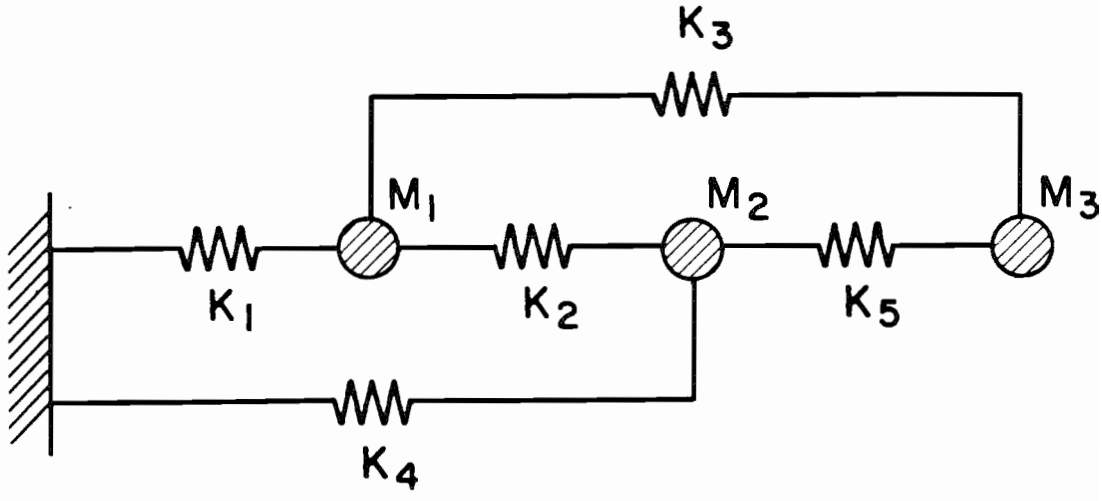
9. K.C. Park and C.A. Felippa, "Partitioned Transient Analysis Procedures for Coupled-Field Problems: Accuracy Analysis", **Journal of Applied Mechanics,** Vol. 47, 1980, pp. 919-926.

10. O.C. Zienkiewicz, E. Hinton, K.H. Leung and R.L. Taylor, "Staggered Time Marching Schemes in Dynamic Soil Analysis and Selective Explicit Extrapolation Algorithm", Proc. 2nd International Symposium for Innovative Numerical Analysis in Applied Engineering Science, Montreux, June 1981.

11. O.C. Zienkiewicz, **The Finite Element Method** , McGraw-Hill, New York-London, 1977.

12. G.W. Gear, **Numerical Initial Value Problems in Ordinary Differential Equations,** Prentice-Hall, 1971.

13. R.L. Taylor and O.C. Zienkiewicz, "Mixed Finite Element Solution of Fluid Flow Problems", in: R.H. Gallagher et al. (eds.), **Finite Elements in Fluids,** Chapter 1, Vol. 4, John Wiley and Sons, 1982.



$M_1 = M_2 = M_3 = 1$

$K_1 = 1$

$K_2 = 100$

$K_3 = 200$

$K_4 = 2$

$K_5 = 10{,}000$

INITIAL CONDITIONS

(i) $\underset{\sim}{u}_0 = \begin{Bmatrix} 4 \\ 4 \\ 4 \end{Bmatrix}$

(ii) $\underset{\sim}{u}_0 = \begin{Bmatrix} 0 \\ 0 \\ 4 \end{Bmatrix}$

Fig. 1. Definition of test problem

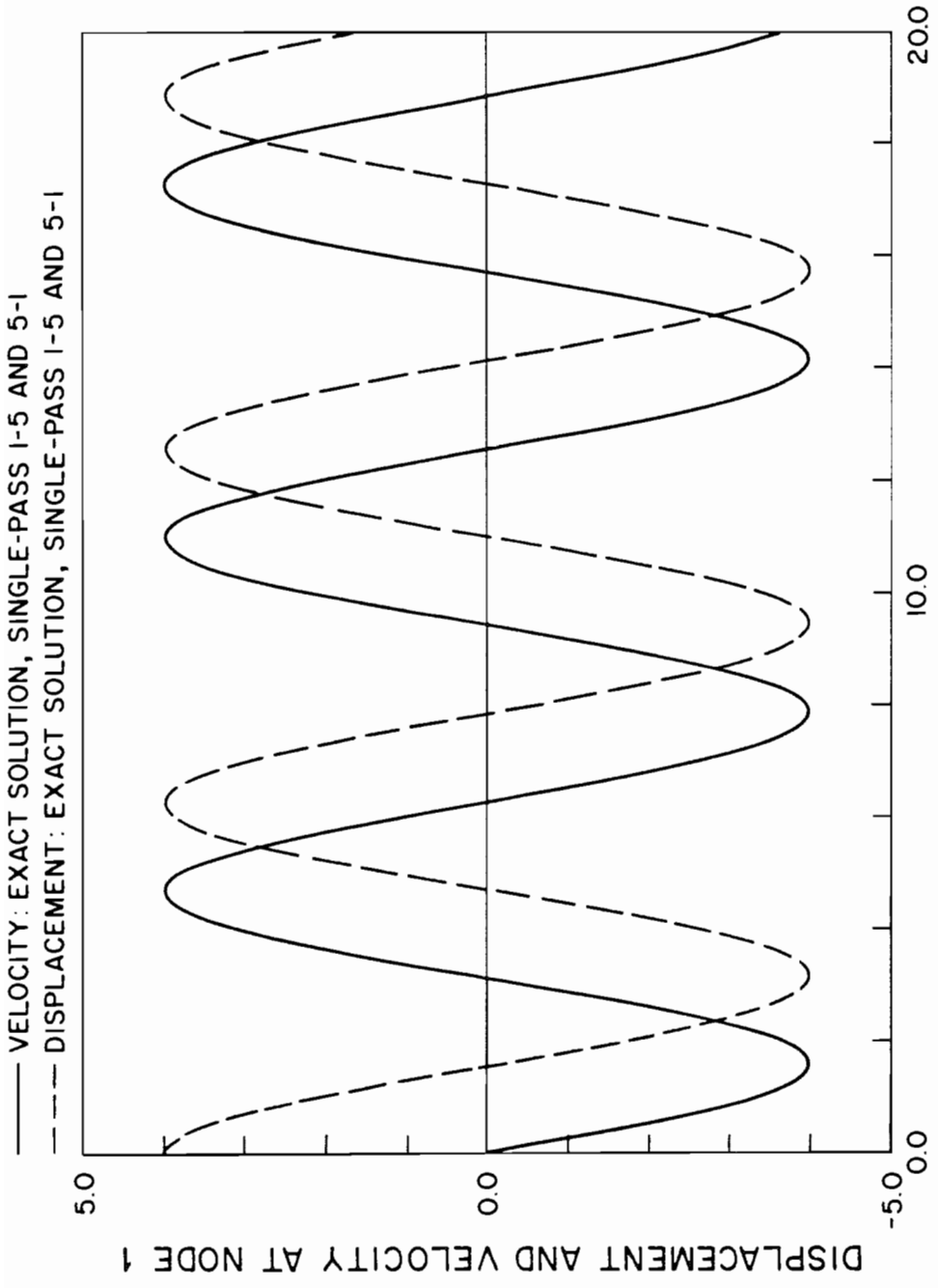


Fig. 2. Single pass solutions for initial conditions (i), time step = 0.01

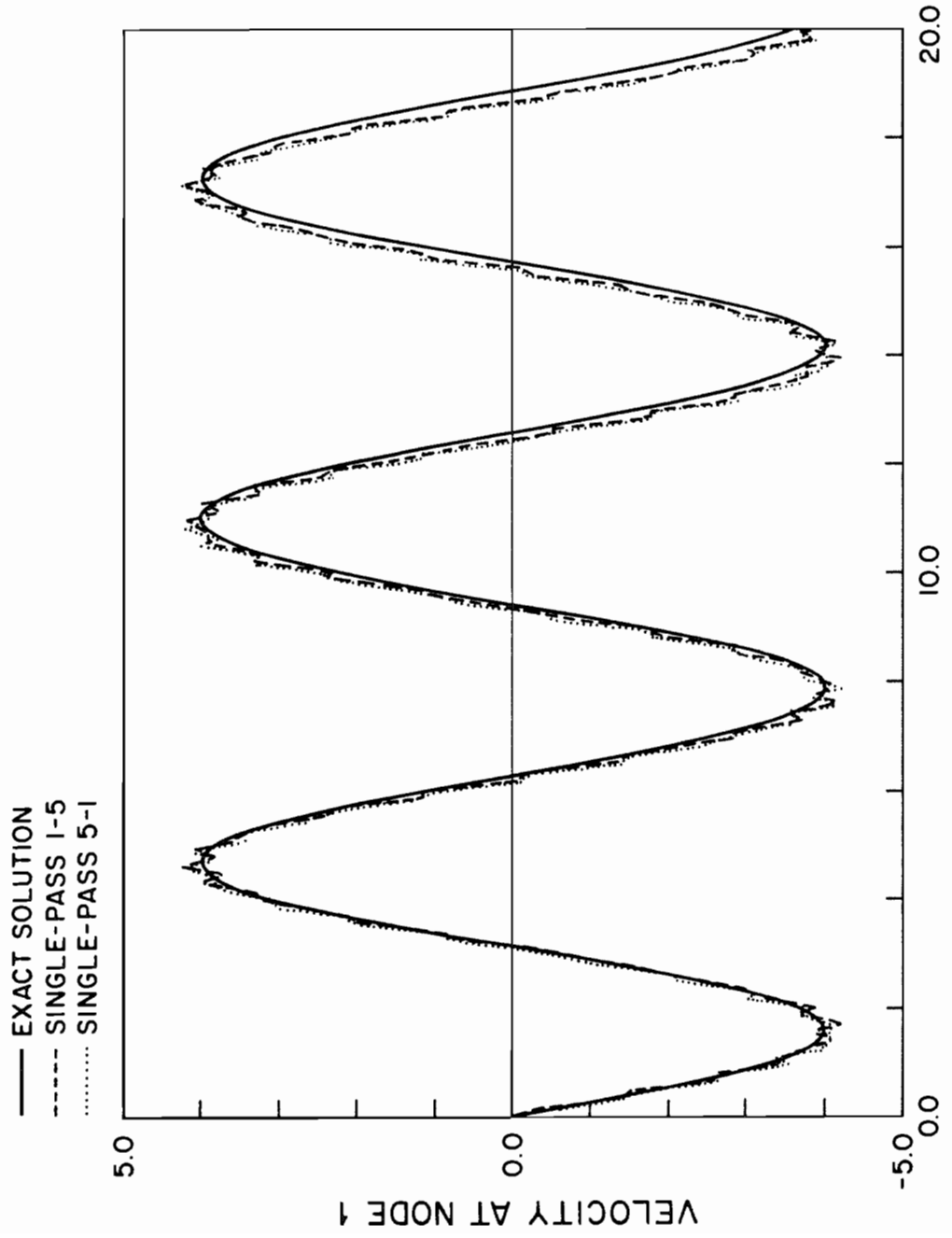


Fig. 3. Single pass solutions for initial conditions (i), time step = 0.04

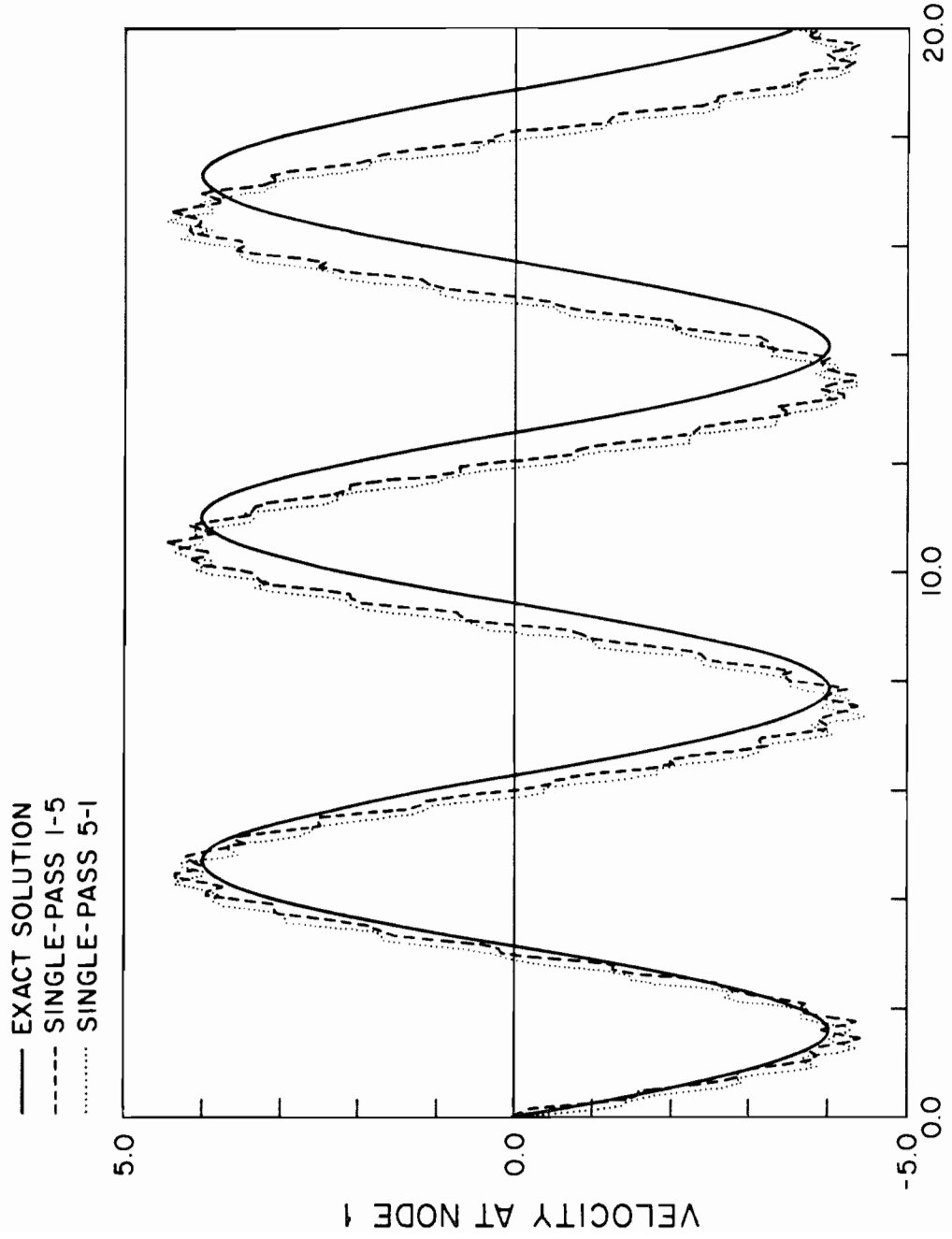


Fig. 4. Single pass solutions for initial conditions (i), time step = 0.08

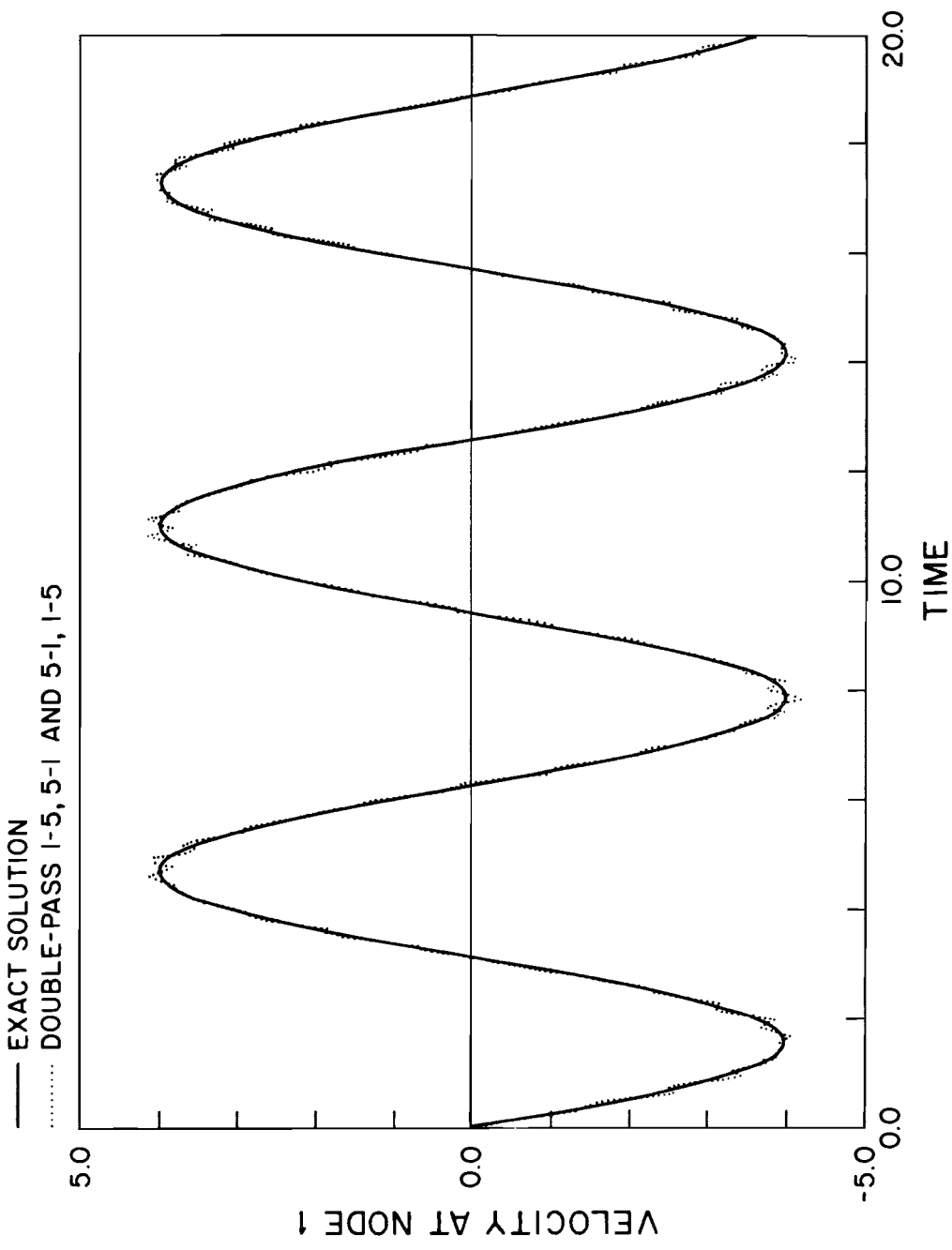


Fig. 5. Double pass solutions for initial conditions (i), time step = 0.04

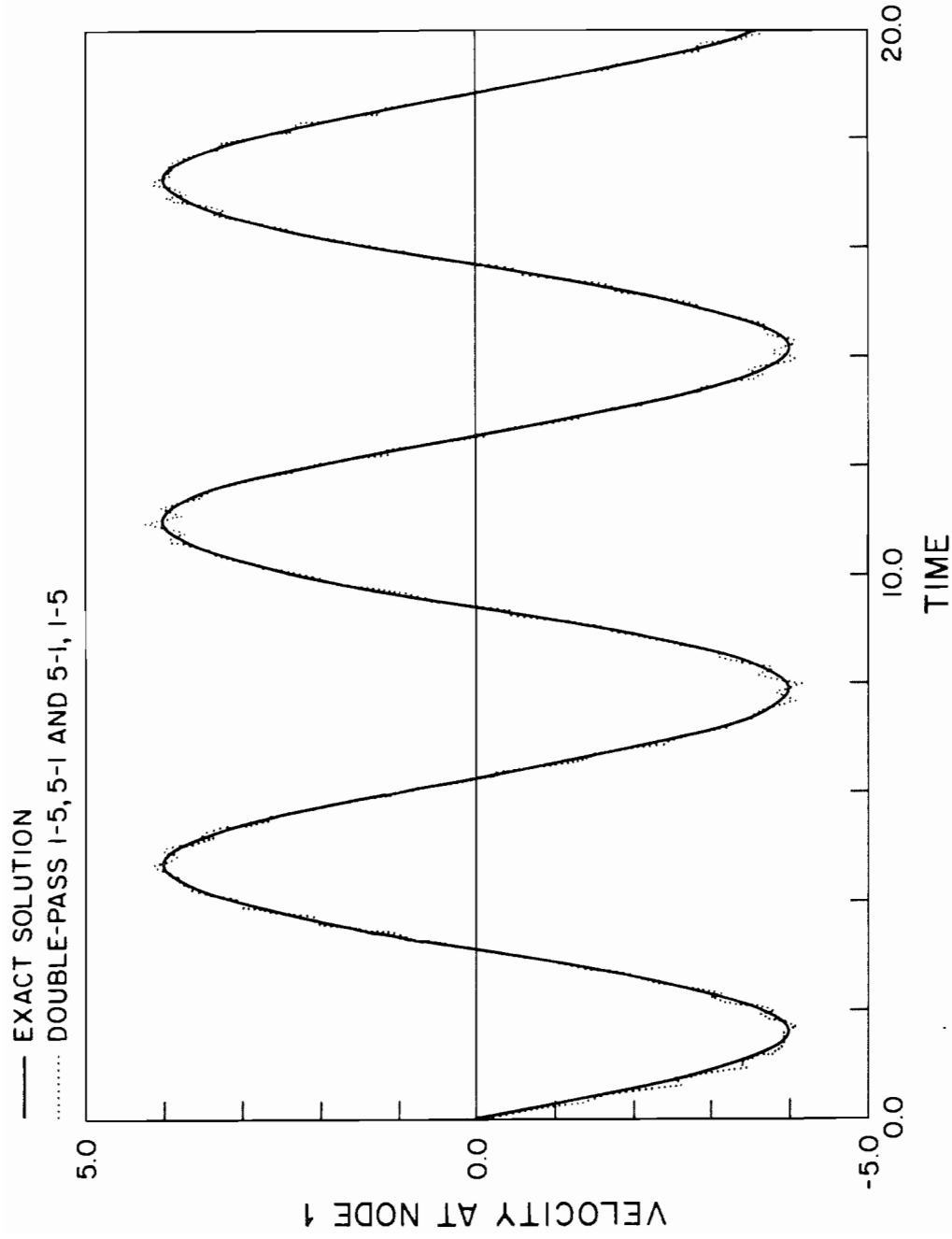


Fig. 6. Double pass solutions for initial conditions (i), time step = 0.08

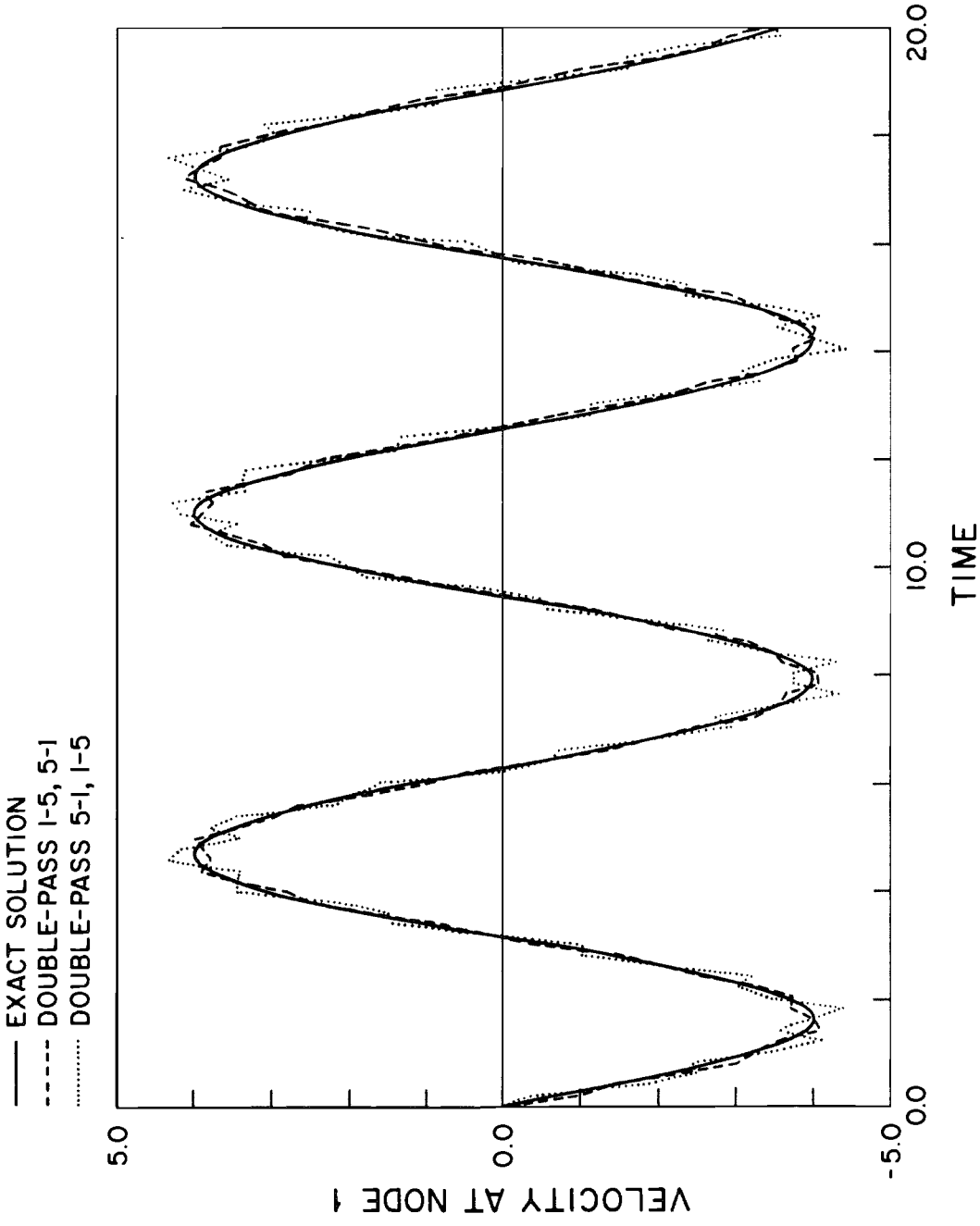


Fig. 7. Double pass solutions for initial conditions (i), time step = 0.20

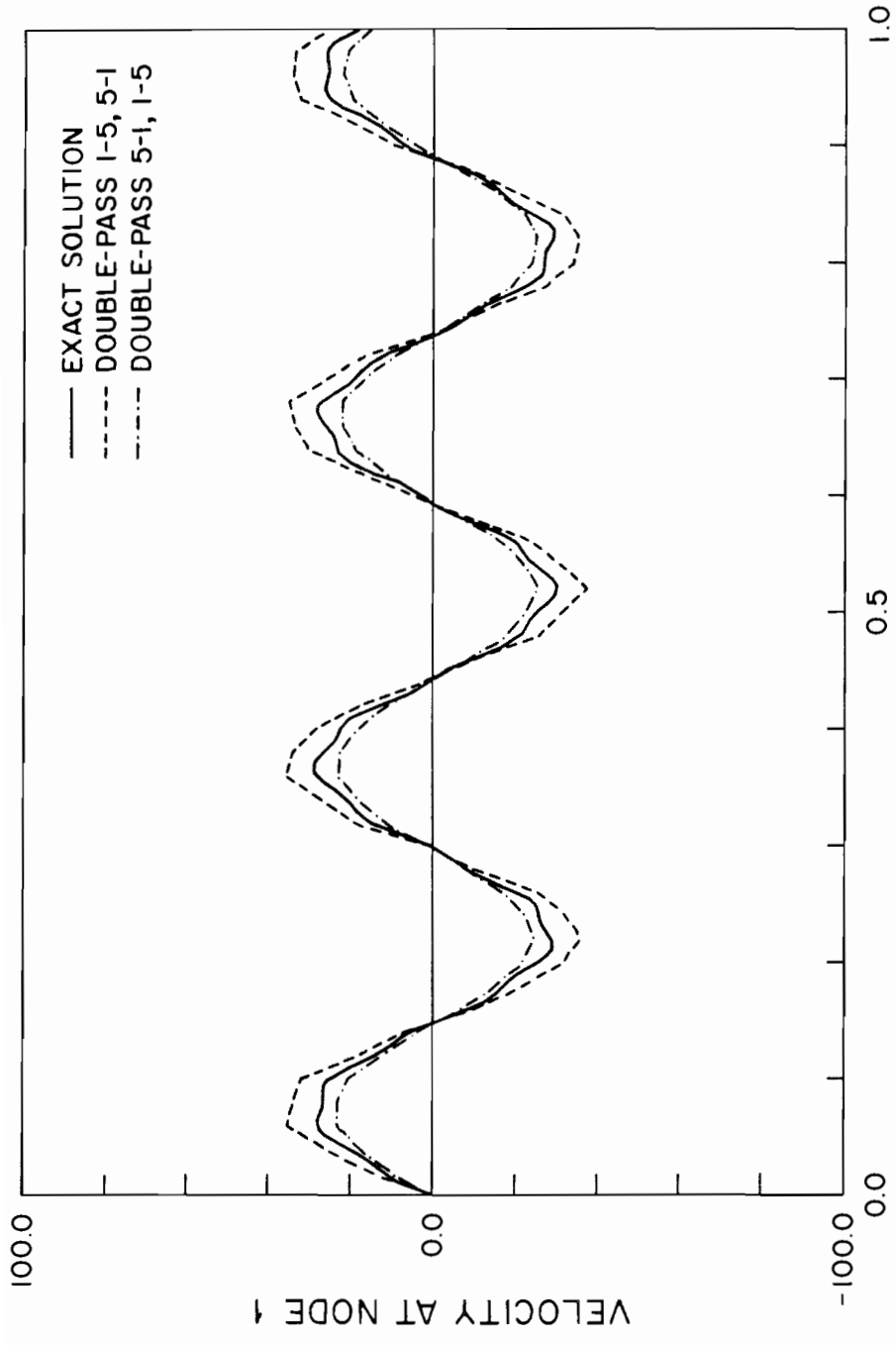


Fig. 8. Double pass solutions for initial conditions (ii), time step = 0.02