JOURNAL OF COMPUTATIONAL PHYSICS 57, 157–187 (1985)

A Variable Node Finite Element Method

M. CALVIN MOSHER

David W. Taylor Naval Ship Research & Development Center, Bethesda, Maryland 20084

Received August 20, 1982; revised May 23, 1984

A new finite element method with variable nodes for integrating partial differential equations in one space dimension is presented. The method utilizes the approximation power of piecewise polynomial functions better than a fixed-node finite element method because it distributes grid points in a nonuniform way to suit the peculiarities of the solution *at each instant of time*. The method uses the gradient and curvature of the approximating piecewise polynomial itself to determine the position of the nodes rather than physical properties which vary from problem to problem. The method has been implemented in a program VFE which has the exceptional feature of a variable space step as well as a variable time step. Since the method concentrates nodes in regions where small inter-nodal spacing is needed, it is very efficient in integrating equations with shock-like solutions. Finally, the method can be extended to systems of partial differential equations in one dimension and to equations in two dimensions.

1. INTRODUCTION

A number of finite difference and finite element methods which allow the grid points to move have been described in the literature, in particular, K. Miller and R. Miller [1], K. Miller [2], Dwyer, Kee, and Sanders [3], and Davis and Flaherty [4]. A discussion of some of these methods with a list of references appears in Gelinas, Doss, and K. Miller [5]. These methods have proven very efficient in integrating time-dependent, nonlinear partial differential equations in one dimension with shock-like solutions. Indeed, many important equations have solutions which develop shocks or moving fronts. Examples are:

(a) nonlinear diffusion equations,

(b) Burgers' equation,

(c) the quench front equation which models the cooling of a hot nuclear fuel rod by water,

(d) the Stefan equation for a melting ice front,

(e) the Hodgkin–Huxley equations for the propagation of electrochemical impulses down a nerve fiber,

(f) the equations of gas dynamics in a shock tube,

0021-9991/85 $3.00

(g) the equations of combustion kinetics in a shock tube,

(h) the shallow water wave equations.

This paper describes a new finite element method with variable nodes and presents the results of the application of a computer program written to implement the method to the first three examples. The method consists of a modification of the standard finite element method which uses piecewise polynomial functions and equally spaced grid points.

A finite element method with equally spaced nodes does not take full advantage of the approximation power of piecewise polynomial functions. This power appears to lie in the possibility of distributing the grid points in a nonuniform way to suit the peculiarities of the function being approximated [6]; that is, approximation of a function ϕ by a piecewise polynomial function of a specific order with a specific number of equally spaced grid points can be greatly improved by redistributing the grid points to suit the peculiarities of the function ϕ [7, p. 180]. However, the solutions to initial-boundary value problems for partial differential equations change with time. Consequently, to utilize better the power of piecewise polynomial functions, a finite element method must place grid points in a nonunifirm way to suit the peculiarities of the solution *at each instant of time.* In particular, the position of the grid points must be a function of time.

Since it is difficult to obtain the best possible approximation for a given number of grid points [6], the new method was devised with the objective of obtaining a good and inexpensive piecewise polynomial approximation at each instant of time rather than the best possible one. Hence, it gives a piecewise polynomial approximation at each instant of time with a nonuniform grid and few grid points although the approximation may not be the best possible one for that number of nodes.

Criteria for the distribution of grid points to obtain this type of approximation to the solution of an initial-boundary value problem have been derived in this paper for continuous piecewise linear functions although similar criteria for higher order piecewise polynomial functions can also be derived. Since this finite element method will use these criteria to adjust the nodes automatically, it will utilize the approximation power of piecewise polynomial functions better than a fixed-node finite element method.

2. The Standard Finite Element Method

This section describes the modifications of the standard finite element method which are a consequence of allowing the grid points to be a function of time. The standard finite element, or Galerkin, method approximates the solution of a partial differential equation with piecewise polynomial functions and fixed grid points. The partial differential equation can be considered as an ordinary differential equation

$$\dot{u} = L(u), \qquad t \geqslant 0$$

where L is a differential operator on the Hilbert space $L^2[0, 1]$ and $[0, 1]$ is the space interval. The approximations lie in a finite dimensional space with basis functions α^i for $i = 1,..., N$, where N is the number of nodes. Since continuous piecewise linear approximations will be used in this method, the α^i are defined as hat functions, i.e.,

$$\alpha^i(x) = \begin{cases} (x - s_{i-1})/(s_i - s_{i-1}) & \text{if} \quad s_{i-1} \leqslant x \leqslant s_i \\ (s_{i+1} - x)/(s_{i+1} - s_i) & \text{if} \quad s_i < x \leqslant s_{i+1} \\ 0 & \text{otherwise} \end{cases}$$

for $i = 1,..., N$. Here $s_1,..., s_N$ are the nodes with $s_1 = 0$ and $s_N = 1$. The piecewise linear function v can be written

$$v(t) = \sum_{i=1}^{N} a_i(t)\, \alpha^i.$$

The coefficients $a_i(t)$ are the values $v(t, s_i)$ of the function at the nodes. If the nodes $s_1,..., s_N$ were fixed in time, the derivative of v with respect to t would be

$$\dot{v}(t) = \sum_{i=1}^{N} \dot{a}_i \alpha^i.$$

Since the nodes s_i can move with time,

$$\dot{v}(t) = \sum_{i=1}^{N} \dot{a}_i \alpha^i + \dot{s}_i \beta^i.$$

The β^i are defined as

$$\beta^i(x) = \begin{cases} -m_{i-1}(x - s_{i-1})/(s_i - s_{i-1}) & \text{if} \quad s_{i-1} \leqslant x \leqslant s_i \\ -m_i(s_{i+1} - x)/(s_{i+1} - s_i) & \text{if} \quad s_i < x \leqslant s_{i+1} \\ 0 & \text{otherwise} \end{cases}$$

for $i = 1,..., N$. The slopes m_i of v are

$$m_i = (a_{i+1} - a_i)/(s_{i+1} - s_i)$$

for $i = 1,..., N - 1$.

To approximate the solution of the initial-boundary value problem, we derive N ordinary differential equations. The L^2-norm of the residual of the partial differential equation, defined with the piecewise linear approximation v,

$$\|\dot{v} - L(v)\|_2$$

is formally minimized with respect to the $\dot{a}_i$ by requiring that the ordinary differential equations

$$\partial \|\dot{v} - L(v)\|_2^2 / \partial \dot{a}_i = 0$$

hold for $i = 1,..., N$. These equations are the same as those derived by the Galerkin method and can be represented as

$$\sum_{j=1}^{N} \dot{a}_j \int_0^1 \alpha^j \alpha^i \, dx + \sum_{j=1}^{N} \dot{s}_j \int_0^1 \beta^j \alpha^i \, dx - \int_0^1 L(v)\, \alpha^i \, dx = 0 \tag{0}$$

for $i = 1,..., N$. However, there are $2N$ unknowns

$$a_1,..., a_N, \qquad s_1,..., s_N$$

and only N equations. Hence, an extra N equations must be derived to distribute the grid points.

In [1] the extra N equations are obtained by formally minimizing

$$\|\dot{v} - L(v)\|_2$$

with respect to $\dot{s}_i$ by requiring that the ordinary differential equations

$$\partial \|\dot{v} - L(v)\|_2^2 / \partial \dot{s}_i = 0$$

hold for $i = 1,..., N$. (These equations are modified in [1] to prevent them from becoming degenerate.) In this paper, on the other hand, explicit criteria for distributing the nodes are derived in the form of algebraic equations. The extra N ordinary differential equations are obtained by differentiating the algebraic equations with respect to time.

3. Criteria for Distributing Nodes

The criteria use the gradient and curvature to distribute the nodes. Similar criteria have appeared in the literature of approximation theory [6, 7] and even in an adaptive method for solving time-dependent partial differential equations [4]. We use the gradient as the primary criterion to distribute nodes, which is a fundamental difference of our method from [4], even though the curvature criterion would distribute them more efficiently with respect to the (spatial) accuracy of the piecewise linear approximation. Our reason is that nodes need to be concentrated in regions where the solution of the partial differential equation has a steep gradient, because the values a_i of the piecewise linear approximation may change rapidly with respect to time in those regions. We formulate the criteria as N algebraic equations where N is the number of grid points and differentiate them with respect to time to obtain N ordinary differential equations. We solve simultaneously these N equations together

with the N finite element Eqs. (0); hence, we solve for the values $a_1,\ldots, a_N$ of the piecewise linear approximation simultaneously with the node positions $s_1,\ldots, s_N$, which is another fundamental difference of our method from that in [4]. The main reason for employing a simultaneous solution is that we want to concentrate nodes where the shock is at the current time step rather than use nodes concentrated where the shock was at the previous time step.

We will discuss first the cirterion which uses the gradient of a function ϕ and secondly the criterion which uses the curvature. Here ϕ is the solution of a partial differential equation at a specific instant of time.

The first criterion consists of the equations

$$(s_{i+1} - s_i)^q \left(\int_{s_i}^{s_{i+1}} |\phi'(x)|^p \, dx \right)^{1/p} = \text{constant}$$

for $i = 1,\ldots, N-1$ with $s_1 = 0$ and $s_N = 1$ and arbitrary positive rational numbers p and q. If the function ϕ were known explicitly, we would choose the constant to be

$$\frac{1}{N-1} \left(\int_0^1 |\phi'(x)|^p \, dx \right)^{1/p}.$$

This criterion concentrates nodes where the gradient of ϕ is large and places few nodes where the gradient is small. But ϕ is not known explicitly, so an implicit approximation is made using only the ordered pairs $\langle s_1, a_1\rangle,\ldots, \langle s_N, a_N\rangle$ where $a_i = \phi(s_i)$ for $i = 1,\ldots, N$ and a discrete analog for the criterion. We choose $q = 1/2$ for simplicity and $p = 2$ for a reason that will be made clear. If a piecewise linear approximation of ϕ is used, then the discrete analog of the criterion is

$$|m_i| \, (s_{i+1} - s_i) = \text{constant}$$

for $i = 1,\ldots, N-1$ with $s_1 = 0$ and $s_N = 1$ and slopes $m_1,\ldots, m_{N-1}$ of the piecewise linear approximation. The criterion is reformulated by defining f_i to be

$$f_i(s_i, s_{i+1}) = |m_i| \, (s_{i+1} - s_i)$$

for $i = 1,\ldots, N-1$ and requiring that

$$f_i = f_{i+1}$$

for $i = 1,\ldots, N-2$. Since

$$|m_i| \, (s_{i+1} - s_i) = |\phi(s_{i+1}) - \phi(s_i)|$$

the discrete criterion requires that

$$|\phi(s_{i+1}) - \phi(s_i)| = |\phi(s_{i+2}) - \phi(s_{i+1})|$$

for $i = 1,\ldots, N-2$. That is, the nodes are distributed on the interval $|0, 1|$ in such a

way that the differences in the values of ϕ evaluated at the endpoints of the intervals $[s_i, s_{i+1}]$ for $i = 1,..., N-1$ are the same in absolute value.

This criterion for the placement of nodal points is adequate for the approximation of an arbitrary function but not for the finite element approximation of the solution of an initial-boundary value problem. Experience with finite element and finite difference approximations indicate that a few grid points are needed where the solution has no gradient. Hence, a constant ε_1 is inserted into the original formula, specifically,

$$(s_{i+1} - s_i)^q \left(\int_{s_i}^{s_{i+1}} (|\phi'(x)|^p + \varepsilon_1) \right)^{1/p} = \text{constant}$$

for $i = 1,..., N-1$. This has the discrete analog for $q = 1/2$ and $p = 2$

$$f_i(s_i, s_{i+1}) = \sqrt{m_i^2 + \varepsilon_1}\ (s_{i+1} - s_i)$$

for $i = 1,..., N-1$ and

$$f_i = f_{i+1}$$

for $i = 1,..., N-2$. The value of p was chosen to be 2 rather than 1 to make f_i differentiable with respect to m_i when m_i vanishes. Although $|\phi'(x)|$ may be very small in some subinterval of $[0, 1]$, the constant ε_1 can be made large enough relative to the maximum value of $|\phi'(x)|$ on $[0, 1]$ to insure that some nodes are placed in the subinterval. As ε_1 is made larger, the distribution of the grid points will depend less on the gradient of ϕ and, consequently, will become more uniform. Indeed, if ε_1 is made large enough, the grid points will be nearly equally spaced.

Experience with finite element and finite difference approximations also indicates that nodes are needed where the second derivative of the solution with respect to the space variable is large. That is, nodes must be placed where the curvature of the solution is large. In particular, if the solution is a shock wave, nodes must be placed at the front and back of the wave. The criterion we would like to use is

$$(s_{i+1} - s_i)^q \left(\int_{s_i}^{s_{i+1}} |\phi''(x)|^p \, dx \right)^{1/p} = \text{constant}$$

for $i = 1,..., N-1$ with $s_1 = 0$ and $s_N = 1$ and arbitrary positive rational numbers p and q. Since a piecewise linear approximation is to be used, it is difficult to formulate a discrete analog of this criterion. Instead, we adopt the criterion

$$B_1(s_{i+1} - s_{i-1})^q \left(\int_{s_{i-1}}^{s_{i+1}} |\phi''(x)|^p \, dx \right)^{1/p}$$
$$+ B_2(s_{i+2} - s_i)^q \left(\int_{s_i}^{s_{i+2}} |\phi''(x)|^p \, dx \right)^{1/p} = \text{constant}$$

for $i = 1,..., N-1$ with $s_1 = 0$ and $s_N = 1$ and arbitrary positive rational numbers p

and q. For $i=1$, B_1 is zero and for $i=N-1$, B_2 is zero. Otherwise, B_1 and B_2 are positive real constants. The discrete analog of this criterion for $q=3/2$ and $p=2$ is

$$B_1\sqrt{(m_i-m_{i-1})^2+\varepsilon_2}\,(s_{i+1}-s_{i-1})+B_2\sqrt{(m_{i+1}-m_i)^2+\varepsilon_2}\,(s_{i+2}-s_i)=\text{constant}$$

for $i=1,\ldots,N-1$. Here ε_2 is a positive real constant which prevents the derivatives with respect to m_{i-1}, m_i, and m_{i+1} of the discrete analog from being infinite when $m_{i-1}=m_i$ or $m_i=m_{i+1}$. The value of ε_2 will be small relative to the maximum of $|\phi''(x)|$ for all x in $[0,1]$. To see how the discrete criterion is an analog of the original, we note that

$$(m_i-m_{i-1})/(s_{i+1}-s_{i-1})$$

approximates $\phi''(s_i)$ and that

$$\begin{aligned}&\sqrt{((m_i-m_{i-1})/(s_{i+1}-s_{i-1}))^2(s_{i+1}-s_{i-1})}\,(s_{i+1}-s_{i-1})^{3/2}\\&\quad=\sqrt{(m_i-m_{i-1})^2}\,(s_{i+1}-s_{i-1}).\end{aligned}$$

The right-hand side of this equation is the term multiplied by B_1 in the discrete analog if ε_2 is set to zero. Other equations used to place nodes in regions where the curvature of ϕ is large are described in Section 11.

To derive one equation for each of the node positions s_i for $i=1,\ldots,N$, we combine the criterion for the gradient with that for the curvature to obtain one criterion for the placement of the nodes

$$\begin{aligned}&f_i(s_{i-1},s_i,s_{i+1},s_{i+2})\\&\quad=\sqrt{m_i^2+\varepsilon_1}\,(s_{i+1}-s_i)+B_1\sqrt{(m_i-m_{i-1})^2+\varepsilon_2}\,(s_{i+1}-s_{i-1})\\&\qquad+B_2\sqrt{(m_{i+1}-m_i)^2+\varepsilon_2}\,(s_{i+2}-s_i)=\text{constant}\end{aligned}\tag{1}$$

for $i=1,\ldots,N-1$. Here B_1 is zero if $i=1$ and B_2 is zero if $i=N-1$. The criterion is reformulated by using the f_i to define the functions g_i

$$g_i(s_1,\ldots,s_N)=\begin{cases}s_1 & \text{if } i=1\\ f_{i-1}-f_i & \text{if } 2\leqslant i\leqslant N-1\\ s_N-1 & \text{if } i=N.\end{cases}\tag{2}$$

Equating each g_i to zero gives N algebraic equations

$$g_i(s_1,\ldots,s_N)=0,\qquad i=1,\ldots,N$$

which determine the distribution of the grid points. Equations (2) require each interval $[s_{i-1},s_i]$ to have the same total amount of gradient and curvature. If an interval $[s_{i-1},s_i]$ has more gradient than other intervals, it must be smaller. If an interval $[s_{i-1},s_{i+1}]$ has more curvature than other intervals, it too must be smaller. The equations $g_1=0$ and $g_N=0$ define the boundary points.

4. The Method as a System of Ordinary Differential Equations

The algebraic Eqs. (2), $g_i = 0$ for $i = 1,..., N$, which determine the position of the nodes $s_1,..., s_N$ can now be used to define N ordinary differential equations. Differentiating each of the functions g_i with respect to time gives

$$\dot{g}_i = 0, \qquad i = 1,..., N.$$

These equations are presented in Appendix I. Adding these N equations to the N equations derived from the Galerkin or finite element approximation of the original partial differential equation gives $2N$ ordinary differential equations in all for the $2N$ unknowns $a_1,..., a_N$, $s_1,..., s_N$. The entire system of equations has the form

$$A(y)\dot{y} = G(y)$$

with $y = \langle a_1, s_1, a_2, s_2,..., a_N, s_N \rangle$. The mass matrix $A(y)$ has the form shown in Appendix II.

To complete the method requires an ordinary differential equation solver to integrate this system of $2N$ equations. However, the equations will be stiff [8] if the variables a_i and s_i change rapidly. This could occur, for example, when the solution of the partial differential equations contains a shock wave. Hence, the Gear–Hindmarsh method is used to integrate the system of equations since it employs Gear's implicit methods [9–11] for stiff equations with a variable time step and a Newton method. The variable node finite element method for integrating partial differential equations is thus complete. It has the exceptional feature that it utilizes a variable time step provided by the Gear–Hindmarsh method and a variable space step provided by Eqs. (2).

5. Advantages of the Method

The method takes advantage of the power of piecewise polynomial functions by distributing the grid points in a nonuniform way to suit the peculiarities of the solution at each instant of time. As a consequence, the method requires many fewer grid points than finite element and finite difference methods with fixed grids. Furthermore, the method allows a larger time step because its ordinary differential equations are smoother or less stiff. These advantages result in the use of less computer storage as well as less CPU time because many fewer computations are needed.

The ordinary differential equations of the method are smoother because they allow nodes in a steep wave front or shock to move with it. The smoothest equations and the largest time step would result if the nodes within the shock were allowed to move with the same speed as the shock. But, this cannot occur. Since the constant ε_1 in Eqs. (2) places nodes in regions where the slope $|m_i|$ is small, nodes are forced through the wave front as shown in Fig. 1. This can be explained as follows. Since the

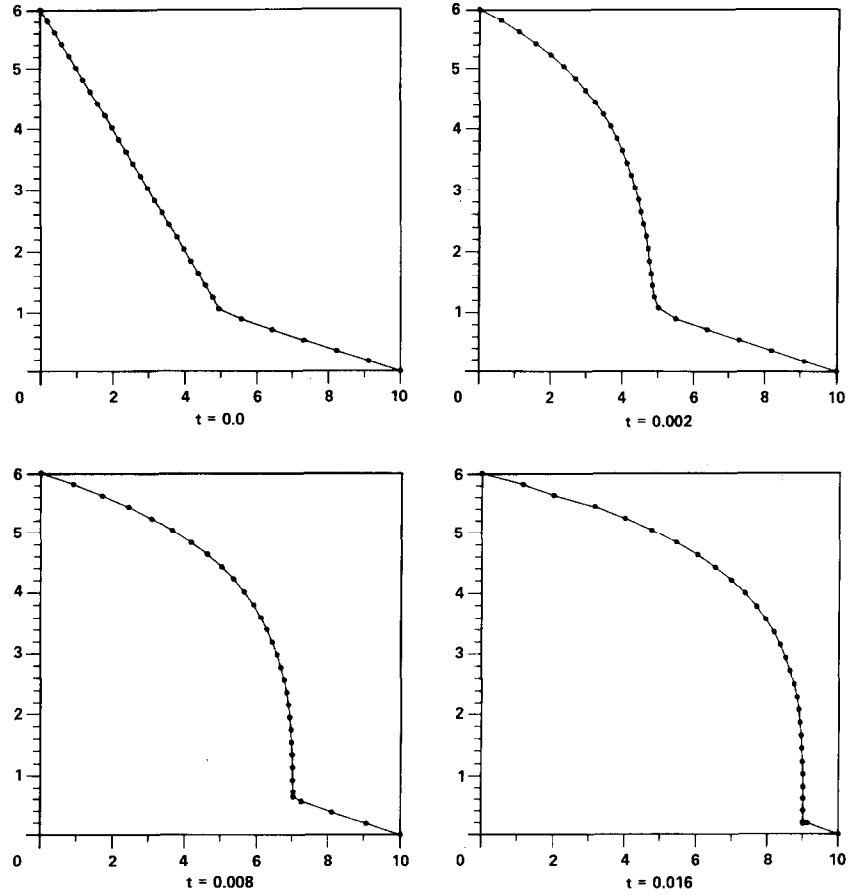

FIG. 1. Solution of nonlinear parabolic equation $u_t = (u^5)_{xx}$ by VFE with $u(0, t) = 6$, $u(10, t) = 0$, 32 nodes, $\varepsilon_1 = 0.01$, $\varepsilon_2 = 10^{-6}$, and $B_1 = B_2 = 0.025$.

gradient of the solution in the shock is constant and equal to zero outside of it, Eqs. (2) allow fewer and fewer nodes to remain in the interval between the front of the shock and the right boundary point as the wave moves to the right. Likewise, more and more nodes must move into the region between the left boundary point and the back of the shock. Hence, nodes do not move with the same speed as the shock but move slowly through it. Nevertheless, the ordinary differential equations of the method will still be smoother than those of the fixed node methods because the difference between the speed of the nodes in the shock and the speed of the shock itself will be small. That is, the value a_i of the piecewise linear approximation at the node s_i which lies in the shock will change less rapidly than in the fixed node method because the difference in the speed of s_i and the speed of the shock is small.

6. Initial Values for the Nodes

A difficult problem remains. Since the grid points $s_1,..., s_N$ are functions of time and are dependent variables in the ordinary differential equations of the method, their initial values must be computed. That is, if this method is applied to an initial-boundary value problem with initial condition $u(x, 0)$, then we must compute the grid points for a piecewise linear interpolation of $u(x, 0)$ as determined by Eqs. (2). Specifically, the system of nonlinear algebraic equations

$$g_i(u(s_1, 0),..., u(s_N, 0), s_1,..., s_N) = 0, \qquad i = 1,..., N$$

must be solved for $s_1,..., s_N$. Although the standard methods for solving nonlinear equations are inefficient when applied to these equations, we have devised a method which solves the system very quickly so that no significant amount of computing time is used. This method is described in Appendix III.

7. Implementing Program

A computer program has been written which uses this new finite element method. It is called VFE for variable node finite element method. A subroutine in VFE uses the method described in Appendix III to solve Eqs. (2) for the initial values of the nodes $s_1,..., s_N$. It is called GIV for g-function initial value solver. A second program was written which uses a fixed node finite element method and is called FFE for fixed node finite element method. Both the VFE and the FFE programs use the Gear–Hindmarsh program for stiff equations with Newton's method. The VFE program has been applied to three equations: a nonlinear parabolic equation, Burgers' equation and the equation for the quench front problem. The program FFE has been used to compare the output of a fixed node method with that of the new method. All computer runs were done on a UNIVAC 1100 with double precision and 18 digits of accuracy.

Since the program VFE requires values for the parameters in Eqs. (2), specifically, ε_1, ε_2, B_1, B_2, and the number of nodes N, a method is needed to estimate values for these parameters. We do not, however, have a numerical method but only a heuristic procedure for estimating acceptable values. In addition to heuristics the procedure uses the program GIV and requires estimates of $\max |\phi'|$ and $\min |\phi'|$. Here ϕ defined on the interval $[0, 1]$ is the exact solution of the initial-boundary value problem at time zero or an estimate of the solution at some later time, and the maximum and minimum are for all x between zero and one.

Since the method is designed to place many nodes where they are needed most rather than to determine the best possible location for the nodes, the values given to the parameters should not be critical. More specifically, the accuracy and time step of VFE should not be sensitive to changes in the parameters. Since only crude estimates are needed, the procedure can use heuristics to estimate values for the parameters. In

addition, we assume that the gradient will be the primary criterion for determining the position of nodes for the reason discussed in Section 3 and because of exprience which indicates that in all but one case it provides sufficient nodes for regions of large curvature. The exceptional case occurs when a steep wave meets a region of small gradient. Hence, weights B_1, B_2 will be given values no larger than needed for this situation.

The first step in the procedure is to select a value for ε_1. We assume that ε_2 is much smaller than ε_1 and that the piecewise linear approximation of ϕ will have slopes m_i such that

$$|m_{i'}| = \max |m_i| \qquad \text{for} \quad 1 \leqslant i \leqslant N-1.$$

$|m_{i'}|$ approximates

$$\max |\phi'(x)| \qquad \text{for} \quad x \text{ in } [0, 1]$$

and

$$m_{i'-1} = m_{i'} = m_{i'+1}.$$

Also, we assume that

$$|m_{i''}| = \min |m_i| \qquad \text{for} \quad 1 \leqslant i \leqslant N-1.$$

$|m_{i''}|$ approximates

$$\min |\phi'(x)| \qquad \text{for} \quad x \text{ in } [0, 1]$$

and

$$m_{i''-1} = m_{i''} = m_{i''+1}.$$

Given these assumption the B terms in $f_{i'}$ and $f_{i''}$ defined by Eqs. (1) will be very small. Since

$$f_{i'} = f_{i''}$$

by Eqs. (2), we obtain

$$\delta = \Delta s_{i''}/\Delta s_{i'} \cong \sqrt{\max |\phi'| + \varepsilon_1}/\sqrt{\min |\phi'| + \varepsilon_1}$$

where the maximum and minimum are taken on the interval [0, 1]. Since experience indicates that values of δ in the interval

$$10^2 \leqslant \delta \leqslant 10^4$$

work well, we select δ in this interval and compute ε_1 from the equation. We now select a specific value for ε_2 much smaller than ε_1. In the test problems we selected ε_2 to be 10^{-6}.

We now replace B_1 and B_2 by a single parameter B and estimate a value for it. (In VFE B_1 equals B_2.) The three factors determining the value of B are the accuracy of the piecewise linear approximation and the stability and stiffness of the system of $2N$ ordinary differential equations. Larger values of B give more weight to the curvature criterion in Eqs. (2) and, consequently, distribute nodes so that the piecewise linear function will approximate better sharp edges which occur where a steep gradient region of ϕ adjoins a region of little or no gradient. However, from experience larger values of B will make the $2N$ system of ordinary differential equations more stiff. On the other hand, if B is zero, the equations may become unstable; in Fig. 7, for example, the node s_i may jump ahead of s_{i+1}, i.e.,

$$s_{i+1} < s_i.$$

From experience values of B in the interval

$$0.0 < B \leqslant 0.1$$

work well.

We select a value of B somewhat smaller than 0.1, namely, 0.025 and apply GIV to ϕ using the values for ε_1 and ε_2 already chosen. Since a value for s_2 is needed in the application of GIV to ϕ, we select an s_2 such that the piecewise linear approximation has the desired accuracy in the interval $[s_1, s_2]$. If the piecewise linear approximation computed by GIV is not accurate enough, we increase the number of nodes by decreasing the value of s_2 used in GIV and apply GIV again. If the piecewise linear approximation is sufficiently accurate, we proceed to check the values of

$$\Delta s_i, \qquad \Delta s_{i+1}, \qquad \Delta s_{i+2}$$

where a steep gradient region of ϕ meets a region where ϕ has little or no gradient as shown in Fig. 7. From experience, if the values of Δs_i, Δs_{i+1}, and Δs_{i+2} satisfy

$$\Delta s_{i+1} = \Delta s_{i+2}(1-\eta) + \Delta s_i \eta \qquad \text{for} \quad 0.1 \leqslant \eta \leqslant 0.9$$

then the system of ordinary differential equations of the method should be stable and not too stiff for the value of B used. If Δs_{i+1} is too small, then decrease B and apply GIV again. If Δs_{i+1} is too large, increase the number of nodes rather than increase B and apply GIV again. In all the test problems B was given the value 0.025.

8. A Nonlinear Parabolic Equation

The new method was first applied to the nonlinear parabolic equation

$$u_t = (u^5)_{xx}$$

on the interval $0 \leqslant x \leqslant 10$. This equation develops a steep wave front or shock. A finite difference scheme for integrating the equation is described by Richtmyer and Morton [12]. The VFE program with 32 grid points was applied to the problem with initial condition

$$u(x, 0) = \begin{cases} -x + 6 & \text{if } 0 \leqslant x \leqslant 5 \\ -1/5(x - 5) + 1 & \text{if } 5 \leqslant x \leqslant 10 \end{cases}$$

and Dirichlet boundary conditions $u(0, t) = 6$ and $u(10, t) = 0$. In Eqs. (2) the parameter ε_1 was set to 0.01, ε_2 to 10^{-6}, and B_1 and B_2 to 0.025. A plot of the output of the VFE program is shown in Fig. 1. The solution quickly develops a steep wave which moves to the right. As expected, the nodes move into the region in which the gradient of the solution $|u_x|$ and the curvature $|u_{xx}|$ are large. The nodes move slowly but smoothly through the wave as it moves toward the right boundary. The fixed node FFE program was then applied to the same initial-boundary value problem and the output was compared, using linear interpolation, to the output of the VFE program at $t = 0.01212$. The output of the FFE program with 251 grid points is in agreement with the output of the VFE program with 32 grid points for at least three digits except in the interval [8.08, 8.16] which contains the front of the wave. Here, there is at most one digit in agreement. When the number of nodes in FFE is increased to 1001, the output is in agreement with the VFE output with 32 nodes for at least three digits except in a much smaller interval [8.11, 8.13] which also contains the front of the wave. Since the fixed node method appears to converge for this problem, the output of the VFE program appears to be accurate. In particular, the VFE program appears to achieve a much better resolution at the front of the wave.

Since the distance between the nodes within the shock is as small as 0.001106 for the VFE program with 32 grid points, FFE with 1001 grid points is expected to achieve the same accuracy at the front of the wave as VFE. However, the fixed node program with 1001 nodes took 40 minutes of CPU time on the UNIVAC 1100 to integrate to $t = 0.01212$, and the variable node method with 32 nodes took 1.5 minutes. Although the Gear–Hindmarsh parameter EPS, which controls the size of the time step, was set to 10^{-5} [9, 11] for both programs, the variable node method used a larger time step. To integrate to $t = 0.01212$, FFE with 1001 nodes used five times as many time steps as VFE, and once the wave front formed, VFE used a time step at least ten times larger. Hence, the variable node method is much more accurate in the front region, requires many fewer grid points and therefore much less computer memory, achieves a larger time step, and uses considerably less CPU time than the fixed node method.

9. Burgers' Equation

The method was next applied to

$$u_t = -1/2(u^2)_x$$

on the interval $0 \leqslant x \leqslant 1$. This equation is of interest because it is a simple nonlinear hyperbolic equation with solutions that develop discontinuities or shocks. To integrate this equation numerically, the diffusion term $(1/R)u_{xx}$ is added to obtain Burgers' equation,

$$u_t = -1/2(u^2)_x + (1/R)u_{xx}.$$

Here, R is a real positive number. The solution of this equation will be continuous and will closely approximate the solution of the original equation. In fact, the larger R is made the better the approximation obtained. In addition, if the shock lies within the interval $[x-, x+]$, the speed of the shock is approximately

$$(u(x-) + u(x+))/2$$

for large R [13] and the width of the shock is $O(1/R)$ [14]. Hence, the location of the shock can be used as a test for the accuracy of our method. Experiment has shown that finite difference and finite element methods become unstable when applied to Burgers' equation unless several grid points lie within the shock. Consequently, if the shock width is small, the fixed node methods will require a large number of nodes. Since our method concentrates nodes only where they are needed, i.e., in the shock region, and places few nodes elsewhere, it uses many fewer nodes. The method has been applied to Burgers' equation in conservation form [15] for three different problems.

For the first problem, $R = 10^3$, the initial condition was

$$u(x, 0) = \begin{cases} 1 & \text{if } 0 \leqslant x \leqslant 0.1 \\ -5(x - 0.3) & \text{if } 0.1 \leqslant x \leqslant 0.3 \\ 0 & \text{if } 0.3 \leqslant x \leqslant 1 \end{cases}$$

and the Dirichlet boundary conditions were $u(0, t) = 1$ and $u(1, t) = 0$. Here 38 nodes were used. In Eqs. (2), the parameter ε_1 was set to 1, ε_2 was set to 10^{-6}, and B_1 and B_2 were set to 0.025. The Gear–Hindmarsh parameter EPS was set to 10^{-5}.

A plot of the output of the VFE program is shown in Fig. 2. A shock wave quickly forms and moves to the right. As expected, nodes move into the shock as it forms and slowly move through the shock as it travels toward the right boundary. To measure the accuracy of the method, we assume that the location of the shock discontinuity is the point on the x-axis where the solution has value 0.5, i.e., the point x_0 where at time t

$$u(x_0, t) = 0.5.$$

Since the shock wave travels at speed 0.5 and is initially located at 0.2, it should be at 0.7 at $t = 1$. Using linear interpolation on the output of VFE at $t = 1$, we find the location of the shock to be 0.69993. This problem is also a check on the size of the

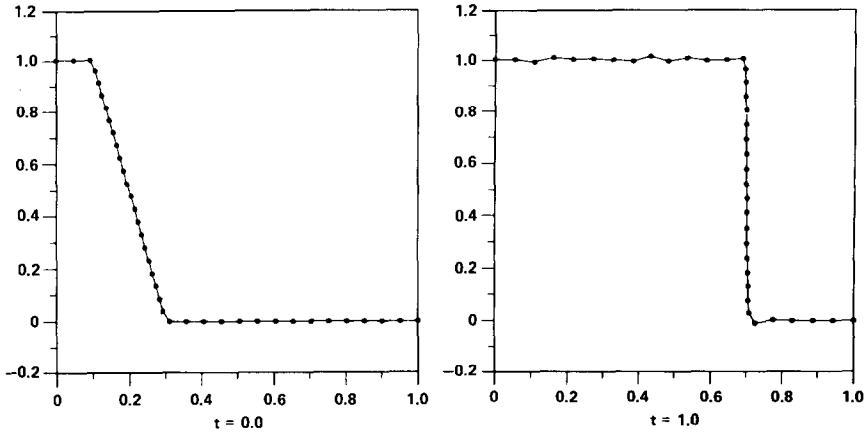

FIG. 2. Solution of Burgers' equation by VFE with $R = 10^3$, $u(0, t) = 1$, $u(1, t) = 0$, 38 nodes, $\varepsilon_1 = 1$, $\varepsilon_2 = 10^{-6}$, and $B_1 = B_2 = 0.025$.

time step. Once the shock has formed, it behaves like a traveling wave with speed c. Specifically, the solution u has the form

$$u = f(x - ct)$$

for some function $f(x)$ and satisfies the equation

$$u_t = -cu_x.$$

For an explicit numerical method for this equation, the Courant–Friedrichs–Lewy condition on the time step Δt and space step Δx is

$$\Delta t / \min \Delta x \leqslant 1/c.$$

Here, $\min \Delta x$ is the minimum space step for any time t. Since our numerical method is implicit, $\Delta t / \min \Delta x$ should be larger than 2. Ratios many times larger than 2 can be obtained, depending on the accuracy required. Since the minimum space step was approximately

$$\min \Delta x = 0.0004$$

a fixed node method would require nearly 1000 nodes to solve this problem with the same accuracy as the variable node method.

In the second problem for Burgers' equation, the same initial and boundary conditions as in the first problem were used, but $R = 10^4$. We used 48 nodes and set the parameters in Eqs. (2) and the Gear–Hindmarsh parameter EPS to the same values used in the previous problem. A plot of the output of the VFE program is

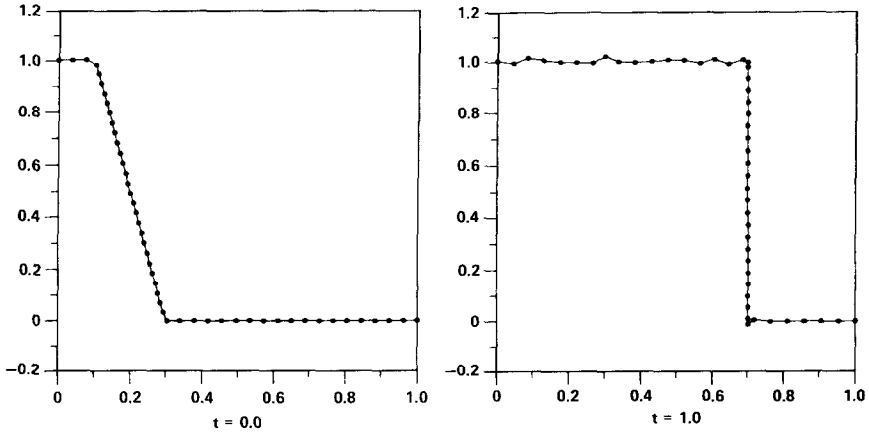

FIG. 3. Solution of Burgers' equation by VFE with $R = 10^4$, $u(0,t) = 1$, $u(1,t) = 0$, 48 nodes, $\varepsilon_1 = 1$, $\varepsilon_2 = 10^{-6}$, and $B_1 = B_2 = 0.025$.

shown in Fig. 3. The nodes move into the shock as it forms and slowly move through the shock as it travels toward the right boundary. As expected, the slope of the shock is ten times steeper than for $R = 10^3$ because the width of the shock is $O(1/R)$. As for the accuracy of our method, the shock should be located at 0.7 at $t = 1$. Using linear interpolation on the output of VFE at $t = 1$, we find the location of the shock to be 0.69997. As a check on the size of the time step, ratios $\Delta t/\min \Delta x$ many times larger than 2 were obtained. Since the minimum space step within the shock was approximately

$$\min \Delta x = 0.37 \times 10^{-4}$$

a fixed node method would require nearly 10,000 nodes to solve this problem with the same accuracy as the variable node method.

In the third problem for Burgers' equation, $R = 10^3$, the initial condition was

$$u(x,0) = \sin(2\pi x) + 0.5 \sin(\pi x)$$

on the interval $0 \leqslant x \leqslant 1$, the Dirichlet boundary conditions were $u(0,t) = 0$ and $u(1,t) = 0$. This test problem was studied by K. Miller and R. Miller [1] and by Gelinas, Doss, and K. Miller [5]. The initial condition consists of two colliding waves, one moving to the right and the other moving to the left. A shock will form, will move to the right, and will die when it hits the right boundary point because of the Dirichlet boundary conditions. Here 57 nodes were used, and in Eqs. (2) the parameter ε_1 was set to 5, ε_2 to 10^{-6}, and B_1 and B_2 to 0.025. The Gear–Hindmarsh parameter EPS was set to 10^{-5}. A plot of the output of the VFE program appears in Fig. 4. The nodes move into the shock as it forms, slowly move though the shock as it moves to the right, and slowly leave the shock as it dies on the right boundary.

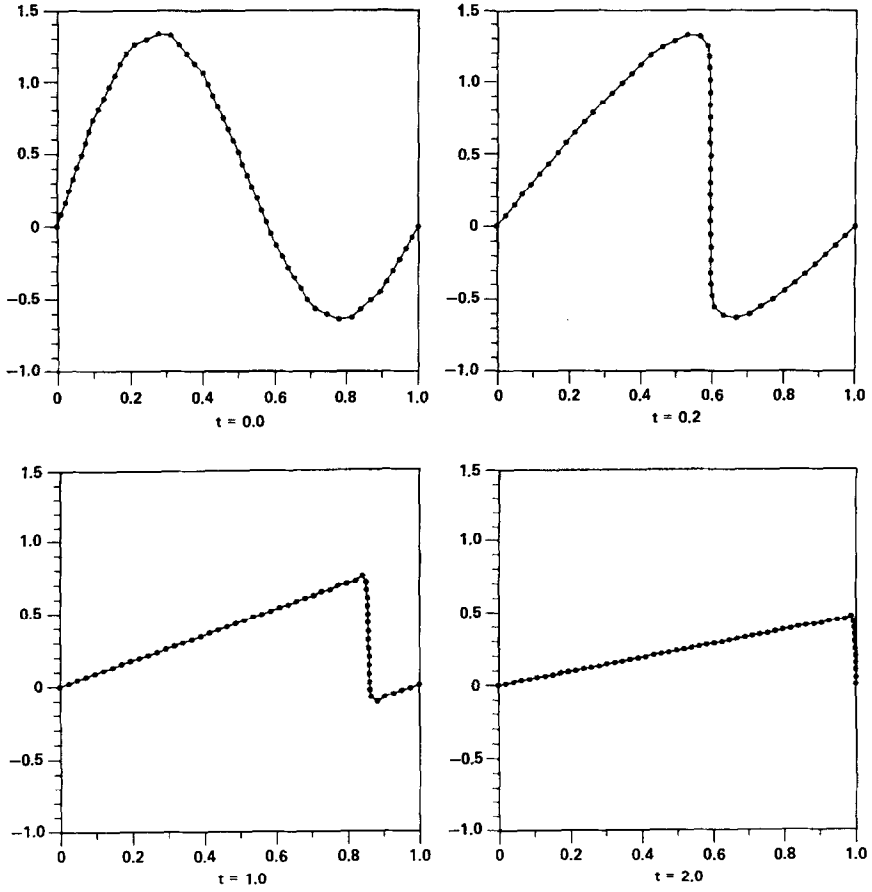

FIG. 4. Solution of Burger's equation by VFE with $R = 10^3$, $u(0, t) = u(1, t) = 0$, 57 nodes, $\varepsilon_1 = 5$, $\varepsilon_2 = 10^{-6}$, and $B_1 = B_2 = 0.025$.

10. THE QUENCH FRONT PROBLEM

The method was next applied to a third test problem, the quench front problem, which has also been studied by Dendy, Schwartz, and Wendroff [16] and by K. Miller [2]. The problem consists of the heat equation with a discontinuous source function, specifically,

$$u_t = u_{xx} - f(u)$$

where

$$f(u) = \begin{cases} Au & \text{if } 0 \leqslant u \leqslant u_c \\ 0 & \text{if } u_c < u. \end{cases}$$

Here, A is positive and u_c is a fixed given value.

The equation models the emergency cooling of a hot nuclear fuel rod by flooding from the bottom with cool water. Here, u is the temperature, x is the distance from the bottom of the rod, and the temperature of the water is zero. There will be little heat transfer at a point x on the rod if the temperature at that point is greater than a critical temperature u_c, even if the rod at that point is immersed in water. That is, if the temperature at a point is greater then u_c, the water is not in direct contact with the rod, but an insulating film of steam lies between the rod and the water. Once the temperature u has fallen below u_c, the water comes into direct contact with the rod and a large heat transfer coefficient A reduces the temperature rapidly toward the water temperature of zero. A large gradient in the temperature develops at the point on the rod with the critical temperature u_c. The result is a wave or quench front which moves up the rod. Since the fuel rod may be four meters long and the quench front is only several milimeters in width, a numerical method with fixed grid points is inadequate and some form of local mesh refinement is required.

If the rod is assumed infinitely long, the solution of the equation for large t is a traveling wave [17] defined by

$$u(x,t) = u(\xi) = \begin{cases} 1 - (1-u_c)\, e^{-c\xi}, & \xi \geqslant 0 \\ u_c e^{c\xi(1-u_c)/u_c}, & \xi \leqslant 0 \end{cases}$$

$$\xi \equiv x - x_0 - ct.$$

Here, x_0 is the initial position of the wave and c is the speed of the wave. Since Dirichlet boundary conditions will not affect the solution as long as the wave is sufficiently far from the boundary points, this exact traveling wave solution can be used to check the accuracy of our method.

The heat transfer coefficient A was set to 80,000, the critical temperature u_c to 0.5, the initial condition to

$$u(x,0) = \begin{cases} 0 & \text{if } 0 \leqslant x \leqslant 0.1 \\ 5(x-0.1) & \text{if } 0.1 \leqslant x \leqslant 0.3 \\ 1 & \text{if } 0.3 \leqslant x \leqslant 1 \end{cases}$$

and Dirichlet boundary conditions to $u(0,t) = 0$ and $u(1,t) = 1$. Here 38 nodes were used and in Eqs. (2) ε_1 was set to 1, ε_2 to 10^{-6}, and B_1 and B_2 to 0.025. The Gear–Hindmarsh parameter EPS was set to 10^{-5}. A plot of the output of the VFE program is shown in Fig. 5. The nodes move into the quench front as it forms and slowly move through the front as it moves toward the right boundary.

To measure the accuracy of the method, we assume that the position of the front is a point on the x-axis where the solution has the value 0.5, i.e., the point x_0 where at time t

$$u(x_0, t) = 0.5.$$

Since the front travels at a speed of 200, it travels a distance of 0.1 in time 0.0005.

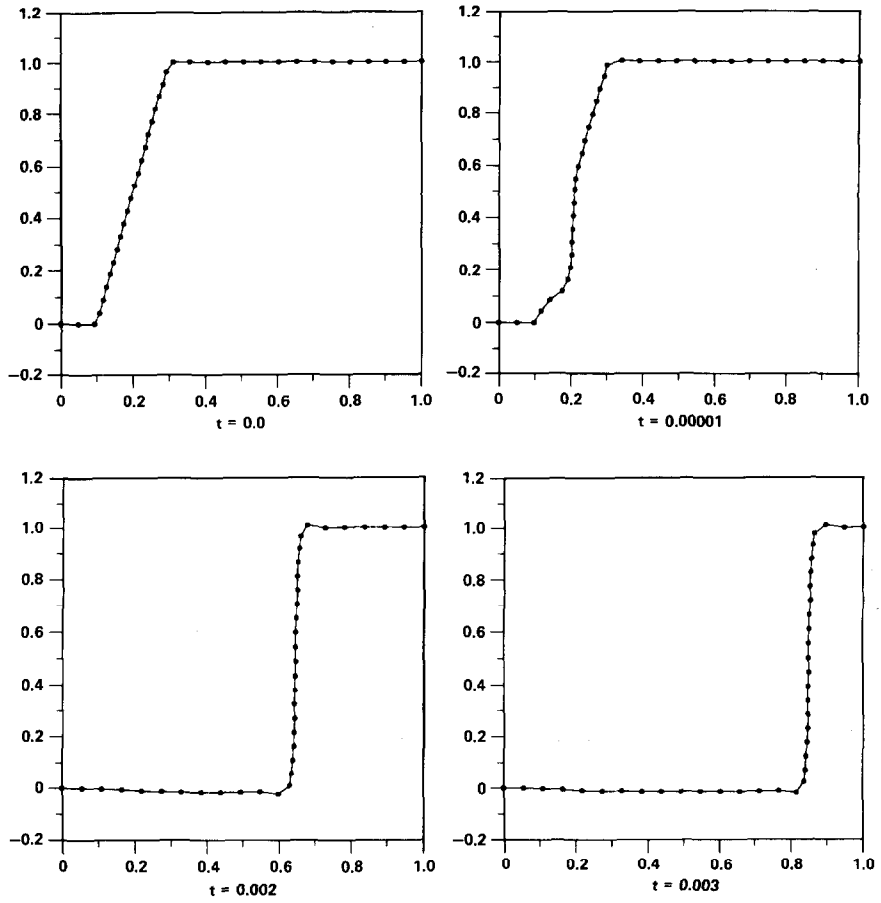

FIG. 5. Solution of the quench front problem by VFE with $A = 80{,}000$, $u_c = 0.5$, $u(0, t) = 0$, $u(1, t) = 1$, 38 nodes, $\varepsilon_1 = 1$, $\varepsilon_2 = 10^{-6}$, and $B_1 = B_2 = 0.025$.

Adding 0.1 to the position of the front computed by linear interpolation from the output of VFE at $t = 0.002$ and comparing this to the position computed from the output of VFE at $t = 0.0025$, we found nearly three digits of agreement. Making the same comparison for $t = 0.0025$ and $t = 0.003$, we found the same accuracy.

In addition, we compared the solution of VFE at each nodal point to the exact traveling wave solution. To do this we equated t to zero in the formula for the exact solution and let the initial position x_0 of the wave be the location of the front computed by linear interpolation from the output of VFE. Computing the difference between the exact and numerical solutions at each node position between $t = 0.002$ and $t = 0.003$, we found the largest error occurred at $t = 0.003$ at a node where the exact solution was 0.0 and the numerical solution was -0.02. This node can be seen clearly in Fig. 5.

Increasing the number of nodes to 75 while keeping the same values for the other

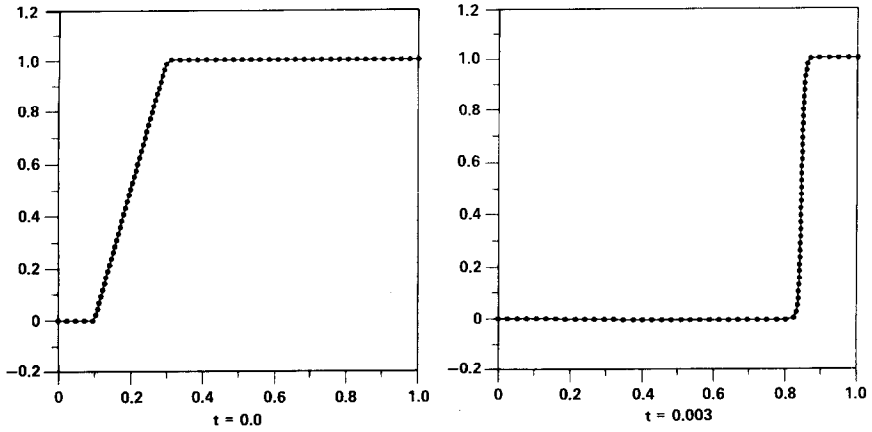

FIG. 6. Solution of the quench front problem by VFE with $A = 80{,}000$, $u_c = 0.5$, $u(0, t) = 0$, $u(1, t) = 1$, 75 nodes, $\varepsilon_1 = 1$, $\varepsilon_2 = 10^{-6}$, and $B_1 = B_2 = 0.025$.

parameters, we applied VFE to the same problem. A plot of the output is shown in Fig. 6. Comparing the numerical solution to the exact traveling wave solution between $t = 0.002$ and $t = 0.003$, we found the largest error at the nodal points to be four times smaller than the largest error with 38 nodes while the error in the location of the front at $t = 0.0025$ and $t = 0.003$ decreased by 25 percent.

Since the solution of the equation is a plane traveling wave for large t, it must satisfy the equation

$$u_t = -cu_x$$

for large t where c is the speed of the wave. Since our numerical method is implicit, $\Delta t/\min \Delta x$ should be larger than 1/200. Ratios many times larger than 1/200 can be obtained, depending on the accuracy required. Since the minimum space step for 38 nodes was approximately

$$\min \Delta x = 0.0006$$

a fixed node method would require nearly 1000 nodes to solve this problem with the same accuracy as the variable node method.

11. Other Equations for Distributing the Nodes

Program VFE was modified to use Eqs. (2) with several new f_i functions different from (1) and then applied to the test problem for $u_t = (u^5)_{xx}$. The simplest modification was

$$f_i = \sqrt{m_i^2 + \varepsilon}\ (s_{i+1} - s_i)$$

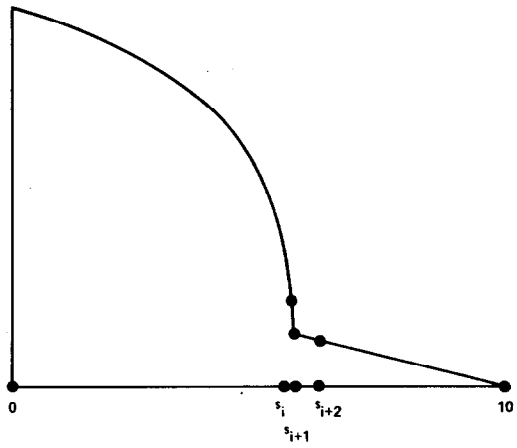

FIG. 7. Position of nodes s_i, s_{i+1} and s_{i+2} at the front of the wave in the solution to the equation $u_t = (u^5)_{xx}$.

for $i = 1,..., N-1$. Here, the m_i are the slopes for $i = 1,..., N-1$ of the piecewise linear approximation. Equations (2) with these f_i terms work well on the test problem until the ratio $(s_{i+1} - s_i)/(s_{i+2} - s_{i+1})$ becomes too small (see Fig. 7). Then s_i jumps ahead of s_{i+1}, i.e., becomes larger than s_{i+1}. The remedy is to add a term to each f_i which will place nodes where the second space derivative of the solution is large. For instance, the term

$$B_1 \sqrt{(m_i - m_{i-1})^2 + \varepsilon_2}\,(s_{i+1} - s_{i-1}) + B_2 \sqrt{(m_{i+1} - m_i)^2 + \varepsilon_2}\,(s_{i+2} - s_i)$$

added to each f_i for $i = 1,..., N-1$ would produce the original formula for f_i given in Eqs. (1). In these f_i functions (and in the ones that follow), B_1 is zero for $i = 1$ and B_2 is zero for $i = N-1$.

We also experimented with Eqs. (2) using

$$f_i = \sqrt{m_i^2 + B_1(m_i - m_{i-1})^2 + B_2(m_{i+1} - m_i)^2 + \varepsilon}\,(s_{i+1} - s_i)$$

for $i = 1,..., N-1$. The ordinary differential equations generated by these equations appear to be unstable when applied to the test problem. Specifically, they appear to be unstable in regions where the slopes $|m_i|$ are small in value. These equations were used because they are easy to solve for the initial $s_1,..., s_N$. Although many different formulas can be used for the f_i, we are restricted to those for which an efficient method is available for solving the nonlinear Eqs. (2) to obtain the initial $s_1,..., s_N$.

Equations (2) were also tried with

$$\begin{aligned} f_i = {} & \sqrt{m_i^2 + \varepsilon_1}\,(s_{i+1} - s_i) \\ & + B_1 \sqrt{((m_i - m_{i-1})/(s_{i+1} - s_{i-1}))^2 + \varepsilon_2}\,(s_{i+1} - s_{i-1}) \\ & + B_2 \sqrt{((m_{i+1} - m_i)/(s_{i+2} - s_i))^2 + \varepsilon_2}\,(s_{i+2} - s_i) \end{aligned}$$

for $i = 1,..., N-1$. When these equations were applied to the test problem, it was found that for the values of B_1 and B_2 used the nodes at the shock front did not move smoothly into the shock. Consequently, the time step was unacceptably small.

12. Conclusion

The new method presented in this paper consists of a modification of the standard finite element method for initial-boundary value problems. It utilizes the approximation power of piecewise polynomial functions better than a fixed-node finite element method because it distributes the grid points in a nonuniform way to suit the peculiarities of the solution at each instant of time. The criteria for placement of the nodes use the gradient and curvature of the approximating piecewise polynomial itself rather than physical properties which vary from problem to problem. These criteria is completed by converting the algebraic Eqs. (2) to ordinary differential equations and adding them to the equations of the standard finite element method to form a simultaneous system. The Gear–Hindmarsh method for stiff equations is used to solve the system.

The method has been implemented in a program VFE which has the exceptional feature of a variable space step as well as a variable time step. Solution of Eqs. (2) for the initial values of the nodes $s_1,..., s_N$ is difficult, but a fast numerical method for solving them has been developed and implemented in the program GIV. When the program VFE was applied to the nonlinear parabolic equation $u_t = (u^5)_{xx}$, to Burgers' equation, and to the equation of the quench front problem, the new method, as expected, was much more efficient than a standard finite element method (in which the grid points are fixed) with respect to the number of grid points used, computer storage, size of the time step, and CPU time. Numerical tests on these equations demonstrated that the method is very accurate and stable.

Finally, and most importantly, the new method can be extended in the obvious ways to systems of partial differential equations in one dimension such as the gas dynamics equations [18, 19] and to equations in two dimensions [19].

Appendix I

In this appendix equations $\dot{g}_1 = 0,..., \dot{g}_N = 0$ are presented. They are derived by differentiating the algebraic Eqs. (2) with respect to time t with $f_1,..., f_{N-1}$ defined by (1). The coefficients B_1 and B_2 are replaced by a single constant B. The following notation is used:

$$\Delta s_i = s_{i+1} - s_i$$

$$D_1 = m_{i+1}(s_{i+2} - s_{i+1})/\sqrt{m_{i+1}^2 + \varepsilon_1}$$

$$D_2 = \sqrt{m_{i+1}^2 + \varepsilon_1}$$

$$D_3 = -m_i(s_{i+1} - s_i)/\sqrt{m_i^2 + \varepsilon_1}$$

$$D_4 = -\sqrt{m_i^2 + \varepsilon_1}$$

$$D_5 = B(m_{i+2} - m_{i+1})(s_{i+3} - s_{i+1})/\sqrt{(m_{i+2} - m_{i+1})^2 + \varepsilon_2}$$

$$D_6 = B\sqrt{(m_{i+2} - m_{i+1})^2 + \varepsilon_2}$$

$$D_7 = -B(m_i - m_{i-1})(s_{i+1} - s_{i-1})/\sqrt{(m_i - m_{i-1})^2 + \varepsilon_2}$$

$$D_8 = -B\sqrt{(m_i - m_{i-1})^2 + \varepsilon_2}.$$

The variables D_7 and D_8 vanish in $\dot{g}_2 = 0$ and D_5 and D_6 vanish in $\dot{g}_{N-1} = 0$. The equations for $i = 1$ and $i = N$ are

$$\dot{g}_1 = \dot{s}_1 = 0$$

and

$$\dot{g}_N = \dot{s}_N = 0.$$

The equations for $i = 1,..., N-2$ are

$$\begin{aligned}
-\dot{g}_{i+1} = {} & \dot{a}_{i-1} D_7/\Delta s_{i-1} + \dot{a}_i(-D_7/\Delta s_{i-1} - (D_3 + D_7)/\Delta s_i) \\
& + \dot{a}_{i+1}((D_3 + D_7)/\Delta s_i + (D_5 - D_1)/\Delta s_{i+1}) \\
& + \dot{a}_{i+2}((D_1 - D_5)/\Delta s_{i+1} - D_5/\Delta s_{i+2}) + \dot{a}_{i+3}(D_5/\Delta s_{i+2}) \\
& + \dot{s}_{i-1}(-D_7 m_{i-1}/\Delta s_{i-1} - D_8) \\
& + \dot{s}_i((D_3 + D_7)\, m_i/\Delta s_i - D_4 + D_7 m_{i-1}/\Delta s_{i-1}) \\
& + \dot{s}_{i+1}(-(D_3 + D_7)\, m_i/\Delta s_i + (D_1 - D_5)\, m_{i+1}/\Delta s_{i+1} \\
& - D_2 + D_4 - D_6 + D_8) \\
& + \dot{s}_{i+2}((-D_1 + D_5)\, m_{i+1}/\Delta s_{i+1} + D_5 m_{i+2}/\Delta s_{i+2} + D_2) \\
& + \dot{s}_{i+3}(-D_5 m_{i+2}/\Delta s_{i+2} + D_6) \\
= {} & 0.
\end{aligned}$$

Appendix II

This appendix presents the mass matrix $A(y)$ for the system of ordinary differential equations of the method

$$A(y)\,\dot{y} = G(y).$$

If there are just eight nodes,

$$y = (a_1, s_1, a_2, s_2,..., a_8, s_8).$$

Here the x's represent non-zero entries from the equations $\dot{g}_1 = 0,..., \dot{g}_8 = 0$. Also, $A(2, 2) = A(16, 16) = 1$, because $\dot{s}_1 = \dot{s}_N = 0$, and $A(1, 1) = A(15, 15) = 1$ because a Dirichlet boundary condition is assumed. The mass matrix $A(y)$ is

$$
\begin{array}{ccccccccccccccccccccccccccc}
1 & 0 & 0 & 0 & 0 & 0 & 0 & & & & & & & & & & & & & & & & & & \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & & & & & & & & & & & & & & & & & & \\
(\alpha^1,\alpha^2) & (\beta^1,\alpha^2) & (\alpha^2,\alpha^2) & (\beta^2,\alpha^2) & (\alpha^3,\alpha^2) & (\beta^3,\alpha^2) & 0 & 0 & & & & & & & & & & & & & & & & & \\
x & x & x & x & x & x & 0 & & & & & & & & & & & & & & & & & & \\
0 & 0 & (\alpha^2,\alpha^3) & (\beta^2,\alpha^3) & (\alpha^3,\alpha^3) & (\beta^3,\alpha^3) & (\alpha^4,\alpha^3) & (\beta^4,\alpha^3) & 0 & 0 & & & & & \bar{0} & & & & & & & & & \\
x & x & x & x & x & x & x & x & x & 0 & & & & & & & & & & & & & & \\
 & 0 & 0 & 0 & (\alpha^3,\alpha^4) & (\beta^3,\alpha^4) & (\alpha^4,\alpha^4) & (\beta^4,\alpha^4) & (\alpha^5,\alpha^4) & (\beta^5,\alpha^4) & 0 & 0 & & & & & & & & & & & & \\
 & & x & x & x & x & x & x & x & x & x & x & 0 & & & & & & & & & & & \\
 & & & 0 & 0 & 0 & (\alpha^4,\alpha^5) & (\beta^4,\alpha^5) & (\alpha^5,\alpha^5) & (\beta^5,\alpha^5) & (\alpha^6,\alpha^5) & (\beta^6,\alpha^5) & 0 & 0 & & & & & & & & & \\
 & & & & x & x & x & x & x & x & x & x & x & x & 0 & & & & & & & & \\
 & & & & & 0 & 0 & 0 & (\alpha^5,\alpha^6) & (\beta^5,\alpha^6) & (\alpha^6,\alpha^6) & (\beta^6,\alpha^6) & (\alpha^7,\alpha^6) & (\beta^7,\alpha^6) & 0 & & & & & & & & \\
 & & & & & & x & x & x & x & x & x & x & x & x & 0 & & & & & & & \\
 & & & & & & & 0 & 0 & 0 & (\alpha^6,\alpha^7) & (\beta^6,\alpha^7) & (\alpha^7,\alpha^7) & (\beta^7,\alpha^7) & (\alpha^8,\alpha^7) & (\beta^8,\alpha^7) & & & & & & & \\
 & & & & & & & & x & x & x & x & x & x & x & x & & & & & & & \\
 & & & & & & & & & 0 & 0 & 0 & 0 & 0 & 1 & 0 & & & & & & & \\
 & & & & & & & & & & 0 & 0 & 0 & 0 & 0 & 1 & & & & & & & \\
\end{array}
$$

APPENDIX III

This appendix describes a method for solving the nonlinear Eqs. (2), $g_1 = 0,..., g_N = 0$ with $f_1,...,f_{N-1}$ defined by (1). We assume there is a continuous piecewise linear approximation ψ to the initial function $u(x, 0)$ with (L_2-norm)

$$\|u(x, 0) - \psi(x)\|_2 < \varepsilon$$

for a specified ε and for a minimal number of nodes $x_1,..., x_M$. We compute a continuous piecewise linear interpolation of ψ with nodal points which satisfy the algebraic Eqs. (2). Let $\mu_1,..., \mu_{M-1}$ be the slopes of ψ in intervals $[x_1, x_2],...,$ $[x_{M-1}, x_M]$, respectively; let $s'_1,..., s'_N$ be the nodes of the piecewise linear interpolation of ψ. The interpolation will have the form $\sum_{i=1}^{N} \psi(s'_i)\, \alpha^i$ with slopes

$$m_i = (\psi(s'_{i+1}) - \psi(s'_i))/(s'_{i+1} - s'_i)$$

for $i = 1,..., N-1$. Although we have no convergence proof for the method, it has converged in all our tests.

The method, consisting of seven steps, solves Eqs. (2) by solving Eqs. (1) with an iterative bisection method on the second node $s_{2,(n)}$ where n is the iteration number. For any n, $s_{1,n} = x_1$. The following rule is used in the method to simplify Eqs. (1):

Rule. Each interval $[x_j, x_{j+1}]$ for $j = 1,..., M-1$ must contain at least three nodes $s_{i,(n)}$, $s_{i+1,(n)}$, $s_{i+2,(n)}$ for any iteration n on $s_{2,(n)}$. Also, in each iteration n on $s_{2,(n)}$, Eqs. (1) are simplified by substituting values from the previous iteration $n-1$ for some of the unknowns in the B_2 terms of the equations. In iteration $n = 1$, B_2 is assumed to be zero.

On the first iteration, $n = 1$, the seven steps take the following form. In Step 1, the second node $s_{2,(1)}$ is chosen in the interval $[x_1, x_2]$ and the parameter B_2 in Eqs. (1) is assumed to be zero. In Step 2, a value for the constant in Eqs. (1) is computed, which will be called C. In Steps 3 through 6, nodes $s_{3,(1)},..., s_{N,(1)}$ are computed, which satisfy Eqs. (1) approximately such that

$$x_1 < s_{2,(1)} < s_{3,(1)} < \cdots < s_{N,(1)}.$$

Hence, these nodes satisfy Eqs. (2) approximately except possibly $g_N = 0$ because $s_{N,(1)}$ may be somewhat larger than x_M. The number of nodes N is an unknown which is computed by the method itself, being simply the index of the first node $s_{N,(1)}$ computed in Step 5 such that $s_{N-1,(1)} < x_M \leqslant s_{N,(1)}$. In Step 7, $s_{2,(2)}$ defined as $(x_1 + s_{2,(1)})/2$ is computed for the next iteration $n = 2$.

On the second and subsequent iterations, $n \geqslant 2$, on $s_{2,(n)}$, B_2 is assumed equal to B_1. In Steps 2 through 6, the nodes $s_{3,(n)},..., s_{N,(n)}$ are computed by using $s_{2,(n)}$ determined in Step 7 of the previous iteration and by substituting values $s_{1,(n-1)},..., s_{N,(n-1)}$ from the previous iteration for some of the unknowns in the B_2 terms of Eqs. (1). Substitution of values from the previous iteration $n-1$ are needed because B_2 is nonzero. Also, in Step 2 a new value C for the constant in Eqs. (1) is computed. The

nodes $s_{1,(n)},\ldots, s_{N,(n)}$ satisfy Eqs. (1) approximately and, therefore, satisfy Eqs. (2) approximately except possibly $g_N = 0$. In Step 7, $s_{2,(n+1)}$ is computed for the next iteration $n+1$ by a bisection argument on the interval $[x_1, s_{2,(1)}]$. That is, the value for the last node $s_{N,(n)}$ computed by Steps 2 through 6 is assumed to be an increasing function of $s_{2,(n)}$. It is also assumed that if a value for $s_{2,(n)}$ is chosen close enough to x_1, then $s_{N,(n)}$ will be strictly less than x_M. Hence, if $s_{N,(n)}$ is larger than x_M, then $s_{2,(n+1)}$ should be smaller than $s_{2,(n)}$. On the other hand, if $s_{N,(n)}$ is smaller than x_M, then $s_{2,(n+1)}$ should be larger than $s_{2,(n)}$. Generally, Step 7 will need to be repeated only a few times because small perturbations in $s_{2,(n)}$ create large perturbations in $s_{N,(n)}$. As a consequence, the method is very fast.

As the number of iterations n goes to infinity, $s_{N,(n)}$ apparantly converges to x_M and $s_{1,(n)},\ldots, s_{N,(n)}$ satisfy Eqs. (2) approximately including $g_N = 0$. That is, there is apparently a solution with N nodes $s'_1,\ldots, s'_N$ which satisfies Eqs. (2) including $g_N = 0$ such that s'_2 lies in the interval $[x_1, s_{2,(1)}]$; as n goes to infinity, $s_{1,(n)},\ldots, s_{N,(n)}$ apparently converges to $s'_1,\ldots, s'_N$.

We will now describe the seven steps of the method, using a square □ to mark the end of each step.

Step 1. For the first iteration, $n = 1$, on $s_{2,(n)}$, choose $s_{2,(1)}$ between x_1 and x_2 and let $B_2 = 0$ in Eqs. (1). □

Step 2. Compute C where $C = f_1(s_{1,(n)}, s_{2,(n)}, s_{3,(n-1)})$. (A value for $s_{3,(n-1)}$ is not needed for $n = 1$ because $B_2 = 0$.) □

Step 3. Assuming $m_2 = m_3 = \mu_1$, solve $f_2(x_1, s_{2(n)}, s_3, s_{4,(n-1)}) = C$ for s_3. (This equation can be solved directly for s_3. A value for $s_{4,(n-1)}$ is not needed for $n = 1$ because $B_2 = 0$.) If $n = 1$ and $s_3 > x_2$ (the Rule is not satisfied), then replace $s_{2,(1)}$ be $(s_{2,(1)})/2$ and go back to Step 2. Otherwise, let the solution s_3 of the equation by $s_{3,(n)}$. Assuming $m_3 = m_4 = \mu_1$, solve $f_3(s_{2,(n)}, s_{3,(n)}, s_4, s_{5,(n-1)}) = C$ for s_4. If $s_4 \leqslant x_2$, let $s_{4,(n)} = s_4$. Solve for $s_{5,(n)},\ldots, s_{K,(n)}$ in the same way such that $s_{5,(n)} < \cdots < s_{K,(n)} \leqslant x_2 < s_{K+1}$, where s_{K+1} is the solution to

$$\begin{aligned} f_K(s_{K-1,(n)}, s_{K,(n)}, s_{K+1}, s_{K+2,(n-1)}) &= \sqrt{\mu_1^2 + \varepsilon_1}\,(s_{K+1} - s_{K,(n)}) \\ &\quad + B_1\sqrt{\varepsilon_2}\,(s_{K+1} - s_{K-1,(n)}) \\ &\quad + B_2\sqrt{\varepsilon_2}\,(s_{K+2,(n-1)} - s_{K,(n)}) \\ &= C \end{aligned} \tag{A3-1}$$

under the condition that $m_K = m_{K+1} = \mu_1$. □

Since ψ is a continuous piecewise linear approximation of the initial function $u(x, 0)$ using a minimal number of nodes $x_1,\ldots, x_M$, slope μ_i does not equal μ_{i+1} for $i = 1,\ldots, M-2$. Hence, ψ cannot have both $m_K = \mu_1$ and $x_2 < s_{K+1} \leqslant x_3$. Concluding that s_{K+1} must be greater than x_2 and using the Rule, we make the assumption that

$$x_1 < s_{K-2} < s_{K-1} < s_K \leqslant x_2 < s_{K+1} < s_{K+2} < s_{K+3} \leqslant x_3 \tag{A3-2}$$

holds. Consequently, we have $m_{K-2} = m_{K-1} = \mu_1$ and $m_{K+1} = m_{K+2} = \mu_2$. The graph of ψ near its node x_2 is shown in Fig. 8 with the relative positions for s_{K-1}, s_K, x_2, s_{K+1}, and s_{K+2} indicated. Looking at Fig. 8, we see that there must be a line connecting $\psi(s_K)$ and $\psi(s_{K+1})$ with slope m_K such that

$$\min(\mu_1, \mu_2) \leqslant m_K \leqslant \max(\mu_1, \mu_2)$$

and such that s_K, s_{K+1}, and s_{K+2} solve equations $f_{K-1} = C$, $f_K = C$, and $f_{K+1} = C$. These equations have the form

$$\begin{aligned} f_{K-1} &= \sqrt{\mu_1^2 + \varepsilon_1}\,(s_K - s_{K-1,(n)}) + B_1\sqrt{\varepsilon_2}\,(s_K - s_{K-2,(n)}) \\ &\quad + B_2\sqrt{(\mu_1 - m_K)^2 + \varepsilon_2}\,(s_{K+1} - s_{K-1,(n)}) = C \qquad \text{(A3-3a)} \\ f_K &= \sqrt{m_K^2 + \varepsilon_1}\,(s_{K+1} - s_K) + B_1\sqrt{(\mu_1 - m_K)^2 + \varepsilon_2}\,(s_{K+1} - s_{K-1,(n)}) \\ &\quad + B_2\sqrt{(\mu_2 - m_K)^2 + \varepsilon_2}\,(s_{K+2} - s_K) = C \qquad \text{(A3-3b)} \\ f_{K+1} &= \sqrt{\mu_2^2 + \varepsilon_1}\,(s_{K+2} - s_{K+1}) + B_1\sqrt{(\mu_2 - m_K)^2 + \varepsilon_2}\,(s_{K+2} - s_K) \\ &\quad + B_2\sqrt{\varepsilon_2}\,(s_{K+3,(n-1)} - s_{K+1}) = C. \qquad \text{(A3-3c)} \end{aligned}$$

(A value for $s_{K+3,(n-1)}$ is not needed for $n = 1$ because $B_2 = 0$.) Hence, we must solve Eqs. (A3-3) simultaneously for s_K, s_{K+1}, and s_{K+2}.

Equations (A3-3), however, are linear in s_K, s_{K+1}, and s_{K+2} if m_K *is held constant*; the equations are presented in Appendix IV in matrix form. Consequently, if m_K is replaced by a specific value in these equations, we can solve them uniquely for s_K, s_{K+1}, and s_{K+2}. Since m_K lies in the interval $[\min(\mu_1, \mu_2), \max(\mu_1, \mu_2)]$, it can be represented as

$$m_K = \phi_K(s_K, s_{K+1}, s_{K+2}) = \frac{\mu_2(s_{K+1} - x_2) + \mu_1(x_2 - s_K)}{s_{K+1} - s_K}.$$

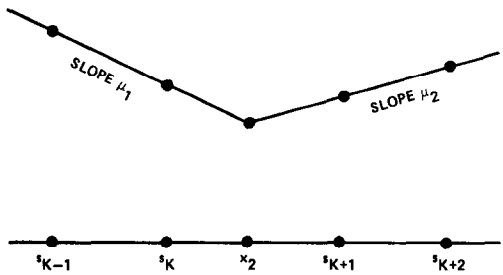

FIG. 8. Graph of ψ near its nodal point x_2 with nodes s_{K-1}, s_K, s_{K+1}, and s_{K+2} which satisfy equations $f_{K-1} = C$, $f_K = C$, and $f_{K+1} = C$.

Hence, we can solve Eqs. (A3-3) for s_K, s_{K+1}, and s_{K+2} by adding a new unknown, m_K, and a new equation, (A3-4d); that is, we solve

$$f_{K-1}(m_K, s_K, s_{K+1}, s_{K+2}) = C \tag{A3-4a}$$

$$f_K(m_K, s_K, s_{K+1}, s_{K+2}) = C \tag{A3-4b}$$

$$f_{K+1}(m_K, s_K, s_{K+1}, s_{K+2}) = C \tag{A3-4c}$$

$$m_K = \phi_K(s_K, s_{K+1}, s_{K+2}) \tag{A3-4d}$$

for s_K, s_{K+1}, s_{K+2}, and m_K.

The following step, Step 4, describes an iterative algorithm for solving Eqs. (A3-4). For each iteration p, the value of the iterate $m_K^{(p)}$ is computed first. With this value substituted for m_K in Eqs. (A3-3), the iterates $s_K^{(p)}$, $s_{K+1}^{(p)}$, $s_{K+2}^{(p)}$ are computed. For the first iteration, $p = 1$, $m_K^{(1)} = (\mu_1 + \mu_2)/2$. The next iterate of m_K is determined by Eq. (A3-4d); that is, $m_K^{(p+1)}$ is determined by whether $\phi_K(s_K^{(p)}, s_{K+1}^{(p)}, s_{K+2}^{(p)})$ is greater than or less than $m_K^{(p)}$. When $|m_K^{(p)} - \phi_K(s_K^{(p)}, s_{K+1}^{(p)}, s_{K+2}^{(p)})|$ is small enough for some iteration p, we let $s_K^{(p)}$, $s_{K+1}^{(p)}$, $s_{K+2}^{(p)}$ be $s_{K,(n)}$, $s_{K+1,(n)}$, $s_{K+2,(n)}$. However, for some iteration p an iterate $m_K^{(p)}$ of the algorithm may produce a solution $s_K^{(p)}$, $s_{K+1}^{(p)}$, and $s_{K+2}^{(p)}$ of Eqs. (A3-3) which is not consistent with the assumption (A3-2) (Cases b and c in Step 4). In addition, the algorithm may not converge because the assumption (A3-2) itself may not be valid. Further explanation of the algorithm is postponed until all the steps of the main method have been described.

Step 4. Determine $s_{K,(n)}$, $s_{K+1,(n)}$, and $s_{K+2,(n)}$ by solving the four Eqs. (A3-4) by iterating on $m_K^{(p)}$, $s_K^{(p)}$, $s_{K+1}^{(p)}$, and $s_{K+2}^{(p)}$ where p is the iteration number. Let $z_1 = \min(\mu_1, \mu_2)$ and $z_2 = \max(\mu_1, \mu_2)$. Since $z_1 \leqslant m_K \leqslant z_2$, the first guess for m_K is $m_K^{(1)} = (z_1 + z_2)/2$. Solve $f_{K-1} = C$, $f_K = C$, and $f_{K+1} = C$ for $s_K^{(1)}$, $s_{K+1}^{(1)}$, $s_{K+2}^{(1)}$ with $m_K^{(1)}$ substituted for m_K. Substitute the solution $\langle s_K^{(1)}, s_{K+1}^{(1)}, s_{K+2}^{(1)} \rangle$ into ϕ_K. If $|m_K^{(1)} - \phi_K(s_K^{(1)}, s_{K+1}^{(1)}, s_{K+2}^{(1)})|$ is less than a prescribed error bound, we are done. If it is, let $s_K^{(1)}$, $s_{K+1}^{(1)}$, and $s_{K+2}^{(1)}$ be $s_{K,(n)}$, $s_{K+1,(n)}$, and $s_{K+2,(n)}$, respectively. If not, another iteration, $p = 2$, is required. Start the new iteration, $p = 2$, by updating z_1 and z_2 in order to compute $m_K^{(2)}$. The updating of z_1 and z_2 involves three cases: in Case a the solution $s_K^{(1)}$, $s_{K+1}^{(1)}$, $s_{K+2}^{(1)}$ is consistent with the assumption (A3-2), and in Cases b and c it is not. The three cases are:

a. $s_K^{(1)} \leqslant x_2 \leqslant s_{K+1}^{(1)}$. If $m_K^{(1)} - \phi_K(s_K^{(1)}, s_{K+1}^{(1)}, s_{K+2}^{(1)})$ is negative, let $z_1 = m_K^{(1)}$. Otherwise, $z_2 = m_K^{(1)}$.

b. $s_{K+1}^{(1)} < x_2$. If z_1 equals μ_1, let $z_2 = m_K^{(1)}$. Otherwise, $z_1 = m_K^{(1)}$.

c. $x_2 < s_K^{(1)}$. If z_1 equals μ_1, let $z_1 = m_K^{(1)}$. Otherwise, $z_2 = m_K^{(1)}$.

Now compute $m_K^{(2)}$, i.e., $m_K^{(2)} = (z_1 + z_2)/2$. Use $m_K^{(2)}$ to solve for $\langle s_K^{(2)}, s_{K+1}^{(2)}, s_{K+2}^{(2)} \rangle$ and then determine whether $|m_K^{(2)} - \phi_K(s_K^{(2)}, s_{K+1}^{(2)}, s_{K+2}^{(2)})|$ is small enough. If it is, let $s_K^{(2)}$, $s_{K+1}^{(2)}$, and $s_{K+2}^{(2)}$ be $s_{K,(n)}$, $s_{K+1,(n)}$, and $s_{K+2,(n)}$, respectively. If not, compute $m_K^{(3)}$ and repeat the process.

If $s_{K+1}^{(p)}$ comes very close to x_2 with respect to some tolerance or $m_K^{(p)}$ comes very

close to μ_1 and $|m_K^{(p)} - \phi_K(s_K^{(p)}, s_{K+1}^{(p)}, s_{K+2}^{(p)}|$ is not small enough after several iterations p (i.e., the algorithm of this step is not converging), then restore $s_{K,(n)}$ to its value obtained in Step 3 and let $s_{K+1,(n)} = (s_{K,(n)} + x_2)/2$. Solve $f_K = C$, $f_{K+1} = C$, and $f_{K+2} = C$ for $s_{K+1,(n)}$, $s_{K+2,(n)}$, and $s_{K+3,(n)}$. □

Step 5. Solve directly as in Step 3 for $s_{K+3,(n)}$, $s_{K+4,(n)}$,..., $s_{K+K',(n)}$, $s_{K+K'+1,(n)}$ where $s_{K+K'+1,(n)} > x_3$. Solve for $\langle s_{K+K',(n)}, s_{K+K'+1,(n)}, s_{K+K'+2,(n)} \rangle$ as in Step 4. □

Step 6. Repeat Step 5 until N nodes $x_1 < s_{2,(n)} < \cdots < s_{N,(n)}$ are computed, using Steps 3 and 4 with x_2, x_3, μ_1, and μ_2 replaced by x_j, x_{j+1}, μ_{j-1}, and μ_j, respectively, for $j = 3,..., M-1$. The number N is the index of the first node computed in Step 5 in the first iteration, $n = 1$, such that $s_{N-1,(1)} < x_M \leqslant s_{N,(1)}$. (It is assumed that $\psi(x)$ is defined for x greater than x_M with slope equal to μ_{M-1}. If the iteration $n \geqslant 2$, $s_{N,(n)}$ may be strictly less than x_M.) □

The last step, Step 7, determines the second node $s_{2,(n+1)}$ for the next iteration $n+1$, using the bisection argument, previously described, on the interval $[x_1, s_{2,(1)}]$.

Step 7. If $n = 1$, let $XL = x_1$, $XR = s_{2,(1)}$, and $s_{2,(2)} = (s_{2,(1)} + x_1)/2$. Using $s_{2,(2)}$ and letting $B_2 = B_1$, repeat Steps 2 through 6 to obtain $s_{N,(2)}$.

If $n \geqslant 2$ and $s_{N,(n)}$ is close enough to x_M, stop; the solution to Eqs. (2) is $s_{1,(n)},..., s_{N,(n)}$. If $s_{N,(n)}$ is not close enough, let $s_{2,(n+1)} = (s_{2,(n)} + XR)/2$ and $XL = s_{2,(n)}$ if $s_{N,(n)} < x_M$, and let $s_{2,(n+1)} = (s_{2,(n)} + XL)/2$ and $XR = s_{2,(n)}$ if $s_{N,(n)} \geqslant x_M$. Repeat Steps 2 through 6 to obtain $s_{N,(n+1)}$. □

We will now explain further the algorithm used in Step 4 to solve Eqs. (A3-3). The algorithm solves the three Eqs. (A3-3) for $s_{K,(n)}$, $s_{K+1,(n)}$, and $s_{K+2,(n)}$ by solving the four Eqs. (A3-4) for s_K, s_{K+1}, s_{K+2}, and m_K by iterating on $s_K^{(p)}$, $s_{K+1}^{(p)}$, $s_{K+2}^{(p)}$, and $m_K^{(p)}$ where p is the iteration number. Choosing for the first guess of m_K, $m_K^{(1)} = (z_1 + z_2)/2$, we compute $s_K^{(1)}$, $s_{K+1}^{(1)}$, and $s_{K+2}^{(1)}$. If these values are consistent with the assumption (A3-2) of the algorithm, i.e., $s_K^{(1)} \leqslant x_2 \leqslant s_{K+1}^{(1)}$ (Case a in the updating of z_1 and z_2), then we use them to evaluate ϕ_K. If $\phi_K^{(1)}$ is greater than $m_K^{(1)}$, we must have chosen $m_K^{(1)}$ too small; consequently, we let $z_1 = m_K^{(1)}$. On the other hand, if $\phi_K^{(1)}$ is less than $m_K^{(1)}$, then we chose $m_K^{(1)}$ too large and let $z_2 = m_K^{(1)}$.

If $s_{K+1}^{(1)}$ is less than x_2 (Case b in the updating of z_1 and z_2), however, then the solution $s_K^{(1)}$, $s_{K+1}^{(1)}$, $s_{K+2}^{(1)}$ of Eqs. (A3-3) is not consistent with the assumption (A3-2) of the algorithm. Values for m_K must be found which will force the value of s_{K+1} to be greater than x_2. We note that the value of s_{K+1} computed in Step 3 under the condition that $m_K = m_{K+1} = \mu_1$ was strictly greater than x_2. By comparing Eq. (A3-1) with (A3-3b), we conclude that s_{K+1} will apparently be greater than x_2 if $|\mu_1 - m_K|$ is made small enough. Hence, if z_1 equals μ_1, m_K should be smaller to make $|\mu_1 - m_K|$ smaller; consequently, we make $z_2 = m_K^{(1)}$. If z_2 equals μ_1, we need a larger m_K to make $|\mu_1 - m_K|$ smaller and, therefore, let $z_1 = m_K^{(1)}$. If $s_K^{(1)}$ is greater than x_2 (Case c in the updating of z_1 and z_2), then $|\mu_1 - m_K|$ is made larger, forcing s_{K+1} and, therefore, s_K to be smaller.

Finally, we will discuss the case in Step 4 in which $s_{K+1}^{(p)}$ is very close to x_2 or $m_K^{(p)}$ is very close to μ_1 even though $\phi_K^{(p)}$ is not close to $m_K^{(p)}$ for some iteration p; i.e., the algorithm in Step 4 is not converging. It apparently does not converge because the assumption of the algorithm in Step 4 that s_{K+1} is greater than x_2 is invalid. To see why this bad assumption was made, we need to examine Step 3. In this step, s_{K+1} was found to be greater than x_2 by solving $f_K = C$ with the assumption that m_K and m_{K+1} were equal to μ_1. This assumption gave the B_2 term in $f_K = C$ the form

$$B_2 \sqrt{\varepsilon_2}\,(s_{K+2} - s_K).$$

That the expression under the square root sign is as small as it can be apparently made s_{K+1} greater than x_2.

To correct the bad assumption made in Step 4, we assume that s_{K+1} must be less than x_2 and s_{K+2} greater. To see why s_{K+1} can be less than x_2, we need to examine the equation $f_K = C$. With this new assumption the equation $f_K = C$ will be the same as it was in Step 3 except for the B_2 term which will now have the form

$$B_2 \sqrt{(m_{K+1} - \mu_1)^2 + \varepsilon_2}\,(s_{K+2} - s_K)$$

where m_{K+1} lies between $\min(\mu_1, \mu_2)$ and $\max(\mu_1, \mu_2)$. That the new expression under the square root sign is larger apparently allows s_{K+1} to be less than x_2. Hence, we do not have to solve for $s_{K,(n)}$ but can restart Step 4 to solve $f_K = C$, $f_{K+1} = C$, and $f_{K+2} = C$ for $s_{K+1,(n)}$, $s_{K+2,(n)}$, and $s_{K+3,(n)}$.

Appendix IV

This appendix presents the equations $f_{K-1} = C$, $f_K = C$, and $f_{K+1} = C$ for the situation shown in Fig. 8 where m_K is held constant. Here m_K is defined by

$$m_K = \frac{\psi(s_{K+1}) - \psi(s_K)}{s_{K+1} - s_K}.$$

Coefficients B_1 and B_2 in the equations are set equal to a single constant B. Also, μ_1 and μ_2 are the slopes of ψ in the intervals $[x_1, x_2]$ and $[x_2, x_3]$, respectively.

$$\begin{aligned}
D(1) &= B\sqrt{\varepsilon_2}, \\
D(2) &= \sqrt{\mu_1^2 + \varepsilon_1}, \\
D(3) &= B\sqrt{(\mu_1 - m_K)^2 + \varepsilon_2}, \\
D(4) &= \sqrt{\mu_1^2 + \varepsilon_1}, \\
D(5) &= B\sqrt{(\mu_2 - m_K)^2 + \varepsilon_2}, \\
D(6) &= \sqrt{\mu_2^2 + \varepsilon_1}.
\end{aligned}$$

The equations represented in matrix form

$$A \begin{pmatrix} s_K \\ s_{K+1} \\ s_{K+2} \end{pmatrix} = b$$

are

$D(1)+D(2)$	$D(3)$	0	s_K		$D(1)\,s_{K-2}+$ $D(2)\,s_{K-1}+$ $D(3)\,s_{K-1}+C$
$-D(4)-D(5)$	$D(3)+D(4)$	$D(5)$	s_{k+1}	$=$	$D(3)\,s_{K-1}+C$
$-D(5)$	$-D(1)-D(6)$	$D(5)+D(6)$	s_{k+2}		$-D(1)\,s_{K+3}+C$

REFERENCES

1. K. MILLER AND R. MILLER, *SIAM J. Numer. Anal.* **18** (1981), 1019–1032.
2. K. MILLER, *SIAM J. Numer. Anal.* **18** (1981) 1033–1057.
3. H. A. DWYER, R. J. KEE, AND B. R. SANDERS, *in* "Proceedings, 7th International Colloquium on Gas Dynamics of Explosions and Reactive Systems," pp. 195–203, AIAA, 1979.
4. S. DAVIS AND J. FLAHERTY, *SIAM J. Sci. Stat. Comput.* **3** (1982), 6–27.
5. R. J. GELINAS, S. K. DOSS, AND K. MILLER, *J. Comput. Phys.* **40** (1981), 202–249.
6. C. DE BOOR, *in* "Spline Functions and Approximation Theory" (A. Meir and A. Sharma, Eds.), pp. 57–72, Birkhäuser, Basel, 1973.
7. C. DE BOOR, "A Practical Guide to Splines," Chap. 12, Springer-Verlag, New York, 1978.
8. C. W. GEAR, "Numerical Initial Value Problems in Ordinary Differential Equations," p. 209, Prentice–Hall, Englewood Cliffs, N.J., 1971.
9. C. W. GEAR, *Math. Comp.* **21** (1967), 146–156.
10. C. W. GEAR, *in* "Information Processing, 68" (A. J. H. Morrel, Ed.), pp. 187–193, North–Holland, Amsterdam, 1969.
11. A. C. HINDMARSH, "GEARIB: Solution of Implicit Systems of Ordinary Differential Equations with Banded Jacobian," Lawrence Livermore Laboratory, 1976, UCID-30130.
12. R. D. RICHTMYER AND K. W. MORTON, "Difference Methods for Initial-Value Problems," p. 201, Interscience, New York, 1967.
13. G. B. WHITHAM, "Linear and Nonlinear Waves," p. 101, Wiley, New York, 1974.
14. G. B. WHITHAM, "Linear and Nonlinear Waves," p. 102, Wiley, New York, 1974.
15. R. D. RICHTMYER AND R. W. MORTON, "Difference Methods for Initial-Value Problems," p. 300, Interscience, New York, 1967.
16. J. DENDY, B. SWARTZ, AND B. WENDROFF, *in* "Topics in Numerical Analysis" (J. J. H. Miller, Ed.), pp. 447–463, Academic Press, New York, 1977.
17. R. SEMERIA AND B. MARTINET, *Proc. Inst. Mech. Engrg.* **130**, Part 30 (1965–1966), 192–205.
18. M. C. MOSHER, "Proceedings, Fifth GAMM Conference on Numerical Methods in Fluid Mechanics," Springer-Verlag, New York, 1983.
19. M. C. MOSHER, "A Variable Node Method Applied to the Gas Dynamics Equations," David W. Taylor Naval Ship R & D Center Report, to be published.