JOURNAL OF COMPUTATIONAL PHYSICS 63, 168–190 (1986)

An Implementation of a Moving Finite Element Method

ANDREW N. HRYMAK, GREGORY J. MCRAE, AND ARTHUR W. WESTERBERG

Department of Chemical Engineering Carnegie-Mellon University, Pittsburgh, Pennsylvania 15213

Received September 18, 1984; revised May 6, 1985

The moving finite element method, an adaptive gridding procedure for systems of partial differential equations whose solutions contain steep gradients, has been implemented in a very straightforward way. Accurate results can be obtained using fewer nodes than that required by fixed mesh solution methods. Performance of the method, using some minor changes to a standard ordinary differential equation algorithm, is illustrated with solutions to Burger's equation. © 1986 Academic Press, Inc.

1. INTRODUCTION

In a deforming finite element framework nodal amplitudes and nodal positions are moved continuously in time, minimizing partial differential equation (PDE) residuals. The moving finite element method (MFEM) is analogous to the method of lines, where one discretizes the spatial part of the partial differential equation but leaves the temporal part as a continuous operator, usually solved by an ordinary differential equation integrator; however, in this case the spatial nodes move [14, 15].

The method is especially suitable for parabolic or evolutionary problems where a steep front sweeps through the spatial domain. Nodes move smoothly to regions where they can minimize the error residuals and move continuously with steep fronts. Even for problems where a steep front may form later in time, i.e., the steep gradient is not present at initial time, the method gives accurate solutions. The advantage is that one needs fewer nodes than for an approximation method that has an evenly spaced fixed grid across the spatial part of the solution domain with a mesh size fine enough to capture the steepest gradients.

Partial differential equation systems that would benefit most from such an analysis are those that contain phenomena with very different time scales and spatial profiles that are very steep in some regions, while essentially flat in other parts of the domain. Examples of such systems are combustors, and oil reservoirs. Unlike stiff sets of ordinary differential equations (ODEs), we then have stiffness in space and time. Allowing the grid points to become degrees of freedom within the PDE solution allows the spatial stiffness to be resolved by allowing grid points to

0021-9991/86 $3.00
Copyright © 1986 by Academic Press, Inc.
All rights of reproduction in any form reserved.

come closer together. At the same time, the set of ordinary differential equations calculating nodal amplitudes and nodal positions can be solved with a stiff temporal solver.

The moving finite element is not a panacea for all problems. For a problem where there is a great deal of information content required and where high accuracy demands small time increments, the overhead due to the implicit set of ordinary differential equations resulting from the basic MFEM formulation may be prohibitively expensive. Just as there is a break-even point in temporal stiffness for solving ordinary differential equations when one shifts from Runge-Kutta-type methods to Gear-type methods, for example, so there is also a point in spatial stiffness when deforming element-type methods will be advantageous. MFEM has also been used for problems where there are two different characteristic velocities, such as in two pulses crossing each other in opposite directions, the soliton problem [5]. The method resolves this problem with great accuracy, but at the time of the pulses crossing we have found that the solution must be solved on an almost fixed mesh. Moving finite elements have also been extended to two dimensions which is not true of some other adaptive grid methods [2, 3, 7, 8].

Moving finite element methods presented thus far have shown substantially better solutions over those available with fixed grid methods, however, the implementation of these algorithms has not been studied in a fashion such that adaptive methods can become an alternative to existing solution methods. This work will show that the MFEM can be simply implemented using an existing ordinary differential equation integrator, LSODI [12]. No special software is needed. Use of the MFEM is very much like a typical method of lines solution, but there are some special considerations. There are two types of problems that must be dealt with when the spatial nodes are allowed to become degrees of freedom in the minimization of the approximation error. The first is that singularities are introduced by the choice of the approximation which can be dealt with by the systematic use of penalty functions. The second problem is that there is no unique solution to the positions of the nodes, and therefore nodes cross, destroying the initial mesh topology of the system. This problem has been dealt with by a minor addition to the LSODI code, which was the only modification to the ODE integrator necessary.

2. The One-Spatial Dimension Formulation

Once the order of the numerical approximation to be used in an analysis has been established there are essentially two viewpoints with regard to reducing the solution error. One is to minimize the error, for a given number of nodes that will be used throughout the problem solution, by continuously varying the node positions. A second view is to add or delete nodes in local intervals to attain a desired performance. Of course, there are hybrids of the two approaches where for part of the temporal domain the grid number is fixed but the positions may or may not move, and for other parts of time nodes are added or deleted. The moving finite element method is essentially the former approach, though it is possible to add or delete nodes. However, for many problems the insertion or deletion of nodes is obviated because of the very good solution properties of the basic method.

The moving finite element method is different from most other adaptive grid methods in several important respects. No dependent variable transformations are involved, nor are any characteristic velocities, time scales, or spatial scales needed. One important practical feature is that the method actually minimizes the global error residual. Even though there may be many physical phenomena creating steep gradients at different times and spatial locations the solution is not tied to a particular property. Since previous time-step information is used to calculate new node positions, large time-steps are allowed.

Consider a single evolutionary equation of the form,

$$\dot{u} = L(u), \qquad t > 0, \tag{1}$$

where $L(u)$ is a nonlinear partial differential operator. The solution to u is approximated by a piecewise continuous function v,

$$v(x, t) = \sum_{j=1}^{n} a_j(t)\, \alpha_j(x, t), \tag{2}$$

where $a_j(t)$ is the magnitude of the solution at node j and $\alpha_j(x, t)$ is the corresponding basis function. In a deforming element approach the approximate solution is a time varying function of both the nodal amplitudes $a_j(t)$ and nodal positions $x_j(t)$

$$v(t) = v(a_1(t),..., a_n(t), x_1(t),..., x_n(t)). \tag{3}$$

The nodal amplitudes are restricted to be finite, and the nodal coordinates are ordered in the following manner:

$$x_0 < x_1(t) < x_2(t) < \cdots < x_n(t) < x_{n+1}. \tag{4}$$

The fixed left and right boundaries are denoted by x_0 and x_{n+1}.

Several of the current approaches to continually deforming finite elements share the same formulation for calculating the time-dependent values of the field variables (Lynch [13]). The basis functions α_i depend upon the space coordinates and the grid points x_j

$$\alpha_i(x, t) = \alpha_i(x, x_j(t)), \qquad j = 1, 2,..., n. \tag{5}$$

If the approximation to u is differentiated with respect to time, then the temporal variation of v is given by

$$\dot{v} = \sum_{j=1}^{n} \dot{a}_j \alpha_j + \dot{x}_j \beta_j, \tag{6}$$

where now the basis functions are given by,

$$\alpha_j = \frac{\partial v}{\partial a_j}, \qquad \beta_j = \frac{\partial v}{\partial x_j}. \tag{7, 8}$$

Unlike a conventional fixed mesh finite element formulation $\dot{v}$ has two sets of basis functions α_j and β_j. Given a transformation between an element or local coordinate system ξ, and the global coordinate system x,

$$x = \sum_j x_j \Psi_j(\xi) \tag{9}$$

it is readily shown that the gradients of the basis functions, α_i, with respect to the node positions are

$$\nabla_j \alpha_i = -\Psi_j \nabla \alpha_i, \tag{10}$$

where ∇_j is the gradient with respect to nodal positions.

Note that the basis functions α_i are changing with time. Differentiating α_i with respect to time, at constant x, gives

$$\frac{\partial \alpha_i}{\partial t} = \sum_j \nabla_j \alpha_i \dot{x}_j = \sum_j -\Psi_j \nabla \alpha_i \dot{x}_j = -V^e \nabla \alpha_i, \tag{11}$$

where the node velocity V^e is given by

$$V^e = \sum_j \dot{x}_j \Psi_j. \tag{12}$$

Clearly the node velocity is a function of all the mesh node velocities with the basis functions in the underlying transformed space. For this system the standard method of weighted residuals formulation is,

$$\left(\sum_j \dot{a}_j \alpha_j - V^e \cdot \nabla v - L(v), W_i \right) = 0. \tag{13}$$

If W_i are the basis functions, α_i and the nodes are fixed, i.e., $V^e = 0$, then the usual Galerkin inner products result. The only difference between a moving node and a fixed node formulation is the presence of the node convection term $V^e \cdot \nabla v$ that corrects the time derivative to account for element deformation. A deforming element approach may be specified by the weighting functions W_i, and the manner in which V^e is determined. The node velocity terms can be specified by the velocities of the fronts in the solution, for example, if they are known.

For example, Herbst *et al.* [9–11] have used a Petrov–Galerkin formulation or an H^{-1} norm, to define the weighting functions for the inner products to calculate n nodal amplitudes and n nodal coordinates,

$$(\dot{v} - L(v), S_i) = 0, \tag{14}$$

$$(\dot{v} - L(v), T_i) = 0, \tag{15}$$

where S_i and T_i are the even and odd piecewise Hermite cubic polynomials.

$$S_i = [\alpha_i(x)]^2[3 - 2\alpha_i(x)], \tag{16}$$

$$T_i = [\alpha_i(x)]^2[\alpha_i(x) - 1][d\alpha_i/dx]^{-1}$$

$$\begin{aligned} \alpha_i &= [x - x_{i-1}]/[x_i - x_{i-1}] && x_{i-1} \leqslant x < x_i, && (17)\\ &= [x_{i+1} - x]/[x_{i+1} - x_i] && x_i \leqslant x < x_{i+1} \\ &= 0 && \text{elsewhere}. && (18) \end{aligned}$$

The moving finite element method of Miller *et al.* [4–6, 14, 15] seeks to calculate the node amplitudes and node positions at the same time. Given a residual R defined by

$$R = \dot{v} - L(v) \tag{19}$$

Miller *et al.* used a least squares formulation that minimized the L^2 norm of R with respect to $\dot{a}_i$ and $\dot{x}_i$, thus defining both W_i and V^e,

$$\left(\sum_j \dot{a}_j\alpha_j + \sum_j \dot{x}_j\beta_j - L(v), \alpha_i\right) = 0, \tag{20}$$

$$\left(\sum_j \dot{a}_j\alpha_j + \sum_j \dot{x}_j\beta_j - L(v), \beta_i\right) = 0 \quad \text{for} \quad i = 1,..., n. \tag{21}$$

There are $2n$ variables and $2n$ inner product equations, making the node amplitudes and node positions completely defined through the set of coupled ODEs.

Note that in this approach,

$$\beta_i = -\Psi_i \sum_j a_j\nabla\alpha_j = -\Psi_i\nabla v. \tag{22}$$

The weights in the two sets of equations are α_i, and β_i, and an indeterminate set of equations results when the weights are not linearly independent. In the isoparametric case, $\alpha_i = \Psi_i$, degeneracy occurs when some component of ∇v is constant for any of the Ψ_i's. This difficulty can also occur in the Herbst *et al.* [9] formulation. There are two approaches to remedy this problem. The first approach is to provide some diagonal dominance to the implicit set of ODEs and can be accomplished through the use of penalty functions. An alternative is to recognize that there are too many degrees of freedom and remove enough degrees of freedom to make the equation set nonsingular.

Herbst *et al.* [10] have shown that for both the Petrov–Galerkin and least squares formulations of the moving finite element methods the resulting coupled set of equations leads to a difference replacement of the PDE to calculate the nodal amplitudes. The addition of the node movement equations provide an equidistributing principle of the form,

$$h_i^{r+1} u_{xx}(x_i^-) = h_{i+1}^{r+1} u_{xx}(x_i^+) + O(h^{r+2}), \tag{23}$$

where $r=0$ for least squares, and $r=1$ for Petrov–Galerkin methods. The equidistributing principle result is useful analytically because it shows that even though completely different criteria are being used to specify the node movement, a similar type of error distribution results. Operationally it is not as useful since the $O(h^{r+2})$ terms dominate for very steep fronts, and for a purely convective problem there are more important criteria driving the nodes than just the second-order error distribution.

Consider a system of partial differential equations in one dimension

$$u = (u^1, u^2, \ldots, u^p), \qquad L = (L^1, L^2, \ldots, L^p), \qquad v = (v^1, v^2, \ldots, v^p). \tag{24}$$

Let each component function v^l, $l = 1, \ldots, p$, be defined by a piecewise continuous function

$$v^l = \sum a_j^l \alpha_j. \tag{25}$$

The minimization is performed with respect to $\dot{a}_i^l$, $\dot{x}_i$, $l = 1, \ldots, p$ and $i = 1, \ldots, n$. The penalized least squares function to be minimized is [5],

$$\min \left\{ \sum_{l=1}^{p} (\dot{v}^l - L^l(v))^2 + \sum_{j=1}^{n} (X_j \Delta\dot{x}_j - \omega_j)^2 \right\}, \tag{26}$$

where the second summation includes terms to prevent degeneracy by penalizing relative node motion through the use of terms involving $\Delta\dot{x}_j$.

If linear basis functions are used,

$$\begin{aligned} \alpha_j(x) &= \frac{x - x_{j-1}}{x_j - x_{j-1}}, && x_{j-1} \leqslant x < x_j, \\ &= \frac{x_{j+1} - x}{x_{j+1} - x_j} && x_j \leqslant x < x_{j+1} \\ &= 0 && \text{elsewhere.} \end{aligned} \tag{27}$$

If the gradient of v over the segment $[x_{j-1}, x_j]$ is denoted by,

$$m_j \equiv (a_j - a_{j-1})/(x_j - x_{j-1}) \tag{28}$$

then the second basis function β_j is given by

$$\begin{aligned} \beta_j &= -m_j\alpha_j & x_{j-1} \leqslant x < x_j, \\ &= -m_{j+1}\alpha_j & x_j \leqslant x < x_{j+1} \\ &= 0 & \text{elsewhere.} \end{aligned} \tag{29}$$

The vector of unknowns containing the nodal amplitudes and coordinates is defined as,

$$y \equiv (a_1^1,..., a_1^p, x_1; a_2^1,..., a_2^p, x_2;...; a_n^1,..., a_n^p, x_n)^T.$$

Then the minimization of (26) results in a set of coupled ordinary differential equations of the form,

$$A(y)\,\dot{y} = g, \tag{30}$$

where A is a block tridiagonal, symmetric, and positive definite matrix, with submatrices of size $[p+1]\,x[p+1]$, and of the form

$$D_{i,j} = \begin{matrix} (\alpha_i, \alpha_j) & & & (\alpha_i, \beta_j^1) \\ & (\alpha_i, \alpha_j) & & (\alpha_i, \beta_j^2) \\ & & (\alpha_i, \alpha_j) & (\alpha_i, \beta_j^p) \\ (\beta_i^1, \alpha_j) & (\beta_i^2, \alpha_j) & (\beta_i^p, \alpha_j) & \sum_{l=1}^{p} (\beta_i^l, \beta_j^l) + AT_{i,j}. \end{matrix} \tag{31}$$

The subdiagonal, diagonal, and superdiagonal blocks are $D_{i,i-1}$, $D_{i,i}$, and $D_{i,i+1}$, respectively, and the additional terms $AT_{i,j}$ are given by

$$\begin{aligned} D_{i,i-1} \quad (i=2,..., n), \qquad & AT_{i,i-1} = -(X_i)^2, \\ D_{i,i} \quad (i=1,..., n), \qquad & AT_{i,i} = (X_i)^2 + (X_{i+1})^2, \\ D_{i,i+1} \quad (i=1,..., n-1), \qquad & AT_{i,i+1} = -(X_{i+1})^2. \end{aligned} \tag{32}$$

The corresponding segment in the right hand side vector g is given by

$$((\alpha_i, L^1(v)),...,(\alpha_i, L^p(v)), \sum_{l=1}^{p} (\beta_i^l, L^l(v)) + X_i\omega_i - X_{i+1}\omega_{i+1}). \tag{33}$$

It is important to note that if a least squares formulation is used with linear basis functions then some of the inner products are not clearly defined. For example, if second-order differential operators are present then there will be discontinuities in the β terms. Since delta functions are not part of an L^2 space, mollification or regularization can be used to obtain a smooth solution [1]. In practise, terms of the form (β_i, v_{xx}) assume a mean value between intervals

$$\beta_i^{\text{mean}} = -(m_i + m_{i+1})/2 \qquad \text{at}\, x_i. \tag{34}$$

If v_{xx} has a delta function of weight

$$v_{xx} = m_{i+1} - m_i \tag{35}$$

then the delta function after mollification becomes

$$(\beta_i, v_{xx}) = -(m_i + m_{i+1})(m_{i+1} - m_i)/2. \tag{36}$$

Similarly,

$$(\alpha_i, v_{xx}) = m_{i+1} - m_i.$$

A catalogue of inner products is available in Gelinas, Doss, and Miller [5].

3. Node Controls

The penalty terms used in the least squares formulation will prevent the system of ODEs from becoming singular. They are required only when,

$$\Delta a_i = \Delta a_{i+1} = 0$$

or

$$m_i = m_{i+1}. \tag{37}$$

The penalty terms X_i and ω_i also prevent nodes from coming too close together. By controlling the node spacing the stiffness of the ODE set can be kept at manageable levels. Therefore, the selection of the X_i and ω_i functions is important in a robust and efficient implementation of the MFEM. However, one must remember they are required only because the basic formulation of the moving finite element method is singular for a number of important cases. X_i specifically monitors the relative node spacing, while ω_i takes into account the possibility that there may be no relative node movement, as steady state is approached or when nodes are brought together into a shock. In the degenerate case the penalty terms solely determine the solution in the local interval containing the singular set of equations.

Though many different types of penalty forms have been tried, the basic requirements of X_i and ω_i are simple. X_i and ω_i cannot be functions of the nodal amplitudes, because this would change the classical finite element formulation embedded within the moving finite element method and destroy the conservation properties of the approximation. The penalty terms must only be a function of the nodal positions x_i and must increase as the nodal positions approach one another.

Consider solving a diffusion problem with a diffusivity of μ. The inner products in the coupling matrix A are of the form

$$\sim \frac{\Delta a_1^2}{\Delta x_i} + X_i^2. \tag{38}$$

Therefore, to maintain balance between the solution and penalty terms Djohmeri [4] and Miller [16] have suggested,

$$X_i^2 \sim \frac{(C_1)^2}{\Delta x_i - \delta}, \tag{39}$$

where C_1 is a constant and δ is a minimum approach distance. Note that as Δa_i becomes very small then the X_1^2 terms become important. Since Δa_i is known only within an order of magnitude of the relative error tolerance ε from the ODE integration, this then suggests that,

$$C_1 \sim O(\varepsilon).$$

If the constant C_1 is chosen a few times larger than the error tolerance then this will cause a smoothing of the node movement due to more drag on the nodes.

Similarly, diffusive forces tend to keep the nodes apart, and in the degenerate case the right-hand side terms are augmented by

$$X_i \omega_i \sim \mu \frac{(C_2)^2}{(\Delta x_i - \delta)^2}, \tag{40}$$

and an adequate form is

$$C_2 \sim O(\varepsilon).$$

In this case, it is better to have C_2 smaller than the truncation error because the nodes will come apart too quickly if there are no convective forces present and the solution has zero gradient.

The only constants left for the user to choose are the ODE truncation error and the minimum node separation. In the examples to be presented it was found that δ is determined by the expected gradients within the problem itself, and therefore, is not really at the user's discretion. The ODE truncation error ε is very important because an excessively small error tolerance will lead to very many iterations in the ODE solver. The form of the penalty function given by (39) has been found to lead to nodes being convected upstream, leaving downstream regions depleted of nodes [4, 16].

To examine the effect of these simple node controls consider the solution of the MFE equations on a segment of domain $[x_{i-1}, x_{i+1}]$ with nodal amplitudes a_{i-1}, a_i, and a_{i+1} at nodes x_{i-1}, x_i, and x_{i+1} respectively. Consider the ith node equations,

$$[D_{i,i-1}]\{\dot{a}_{i-1}, \dot{x}_{i-1}\}^T + [D_{i,i}]\{\dot{a}_i, \dot{x}_i\}^T + [D_{i,i+1}]\{\dot{a}_{i+1}, \dot{x}_{i+1}\}^T$$
$$= \{(\alpha_i, L(v)), (\beta_i, L(v))\}^T. \tag{41}$$

For linear finite elements, the left-hand side matrix submatrices for the ith rows appear as

$$D_{i,i-1} = \begin{bmatrix} \Delta x_i/6 & -\Delta a_i/6 \\ -\Delta a_i/6 & m_i\, \Delta a_i/6 - X_i^2 \end{bmatrix}, \tag{42}$$

$$D_{i,i} = \begin{bmatrix} (\Delta x_i + \Delta x_{i+1})/3 & -(\Delta a_i + \Delta a_{i+1})/3 \\ -(\Delta a_i + \Delta a_{i+1})/3 & (m_i\, \Delta a_i + m_{i+1}\, \Delta a_{i+1})/3 + X_i^2 + X_{i+1}^2 \end{bmatrix}, \tag{43}$$

$$D_{i,i+1} = \begin{bmatrix} \Delta x_{i+1}/6 & -\Delta a_{i+1}/6 \\ -\Delta a_{i+1}/6 & m_{i+1}\, \Delta a_{i+1}/6 - X_{i+1}^2 \end{bmatrix}. \tag{44}$$

Consider the operator $L(v)$ to be the diffusion term μu_{xx}. Therefore, using inner products for diffusion terms in Gelinas *et al.* [5]

$$(\alpha_i, L(v)) = \mu(m_{i+1} - m_i), \tag{45}$$

$$(\beta_i, L(v)) = -(\mu/2)(m_i + m_{i+1})(m_{i+1} - m_i). \tag{46}$$

Both go to zero if $m_{i+1} = m_i$, or if $\Delta a_i, \Delta a_{i+1} = 0$. For simplicity let the form of the penalty terms be

$$X_i^2 = C_1^2/\Delta x_i, \tag{47}$$

$$X_i \omega_i = C_2^2/\Delta x_i^2. \tag{48}$$

If $\Delta a_i, \Delta a_{i+1} = 0$, then only penalty terms will be left in the $\dot{x}_i$th equation (41).

$$-\frac{C_1^2}{\Delta x_i}\dot{x}_{i-1} + C_1^2\left(\frac{1}{\Delta x_i} + \frac{1}{\Delta x_{i+1}}\right)\dot{x}_i - \frac{C_1^2}{\Delta x_{i+1}}\dot{x}_{i+1} = \frac{C_2^2}{\Delta x_i^2} - \frac{C_2^2}{\Delta x_{i+1}^2}. \tag{49}$$

If $C_2 = 0$, then the node velocity of x_i is just a distance weighted average of its neighbors' velocities (the constant C_1 becomes irrelevant for that equation though it is still acting as a velocity damping term for all the other MFE equations).

For the case $m_{i+1} = m_i$, (after pivoting) and $C_2 = D$ (41) becomes,

$$\frac{C_1^2}{\Delta a_i}\dot{x}_{i-1} - C_1^2\left(\frac{1}{\Delta a_i} + \frac{1}{\Delta a_{i+1}}\right)\dot{x}_i + \frac{C_1^2}{\Delta a_{i+1}}\dot{x}_{i+1} = 0 \tag{50}$$

and the velocity $\dot{x}_i$ is the weighted average of its neighbors

$$w = \frac{\Delta a_i}{\Delta a_{i+1}},$$

$$\dot{x}_i = (\dot{x}_{i-1} + w\dot{x}_{i+1})/(1 + w).$$

Again, the constant C_1 is not important for that particular equation, and the nodes move to equalize the difference in nodal amplitudes between elements (an equidistribution of the nodal amplitudes). Similar results are seen when there are convective terms present in the differential operator. This equidistribution causes a slight lift-up of the front edge of a shock [15, 16].

As stated earlier, these penalty terms are really only a device to add diagonal dominance to the equations in the case of singularity in the basic formulation; therefore, other schemes are possible. For example, node controls can be based on the use of gradients [6, 16] in a manner similar to techniques used in transformation methods for adaptive gridding. In the MFEM, penalty forms using gradients have the effect of dampening the relative motion of the nodes as v becomes steeper.

$$X_i^2 \sim \frac{(C_3)^2(\text{gradient})^2}{\Delta x_i - \delta} + \frac{(C_4)^2}{\Delta x_i - \delta}. \tag{51}$$

Since diffusion in the β equations forces nodes apart, after they have resolved a steep gradient, another proposed technique is to add extra diffusion in the β equation set [4, 16]. In the β equations only,

$$\mu \to \mu(1+E), \qquad \text{where} \quad 0 < E < 1.$$

The value of E can actually be determined from a characteristic type analysis for simple equations to determine an E that will not give oscillations in the solution.

When the problem has a near shock, i.e., becomes nearly a delta function, then the equations are essentially hyperbolic. In an L^2 norm nodes are forced into the shock region. A solution to this problem is to use a different norm, a weighted L^2 norm [16]. An example would be a gradient weighted norm, where $m = u_x$,

$$w = (1 + u_x^2)^{1/2}.$$

With this weight the time derivative is normalized using an arc length transformation resulting in a new set of weighted residuals,

$$\begin{aligned} (u - L(u), \alpha_i) &\to (u - L(u), \alpha_i w) = 0, \\ (u - L(u), \beta_i) &\to (u - L(u), \beta_i w) = 0. \end{aligned} \tag{52}$$

As brought to our attention by Dr. R. Gelinas, a reviewer of this paper, Wathen *et al.* have some results with the MFEM with no penalty terms and we note their work for completeness. Wathen and Baines [17, 18] have studied the problems of adjacent elements having the same slope and shock formation. Instead of using penalty terms to prevent matrix singularity, Wathen suggests a two-part approach in maintaining A nonsingular. In the first step, the β equation associated with the singular node is deleted and the equation $\dot{x}_i = 0$ is used. The second step involves adding a multiple of the vector which spans the null space. The exact multiple is determined by some arbitrary criterion.

Wathen deals with the problem of nodes overtaking one another as indicative of shock formation in hyperbolic systems. The node amplitudes are made discon- [illegible]

$$\dot{x}_{i+1} = \dot{x}_i = \lim \frac{\int_{x_1}^{x_2} L(u)\,dx}{a_2 - a_1},$$

as

$$\begin{aligned} &a_1 \to a_i, \qquad && x_1 \to x_i = x_{i+1} \qquad && \text{from the left,} \\ &a_2 \to a_{i+1}, && x_2 \to x_i = x_{i+1} && \text{from the right,} \\ &a_i \neq a_{i+1}, && x_i = x_{i+1}. && \end{aligned}$$

If the system is not hyperbolic this scheme will not represent the physical system correctly as pointed out by Wathen. The next section presents a simple scheme, using LSODI, which provides a way of recovering from an entangled mesh topology.

4. The problem of Node Crossing

The assumption at the beginning is that

$$x_1 < x_2 < \cdots < x_i < \cdots < x_N$$

for all time, i.e., the node topology must stay the same and nodes may never cross. While the penalty functions will provide some resistance to node crossing, once they have crossed there is no force available to correct the mesh topology. By now the reader has probably observed that the nodes will follow characteristics of the governing PDE, if the exist. Therefore, it is easy to conceive of a case, for example counterstreaming pulses, where characteristics cross. Nodes following the respective pulses would cross when the pulses crossed, and the solution could still be defined. In the solution of certain types of problems with moving finite elements one encounters situations where a minimization of the error criterion leads to node crossing, which is an infeasible solution with regards to mesh topology restrictions. The objective was not to change any code in LSODI [11] (which was used as the temporal integrator for all the examples) to make it easier to use the method directly as though it were another method of lines. The problem of node crossing required a small addition to the LSODI code. Essentially another error norm is added for the corrector in the predictor–corrector algorithm of LSODI to check. The proposed fix is quite simple but seems to work well on cases tried thus far.

Consider at time t an increasing sequence of positions of nodes

$$x_1 < x_2 < \cdots < x_i < x_{i+1} < \cdots < x_n \qquad \text{at time } t.$$

After taking a successful step Δt using LSODI based only on the error in the corrector, the mesh could appear as

$$x_1 < x_2 < \cdots < x_{i+1} < x_i < \cdots < x_n \qquad \text{at time } t+\Delta t.$$

The mesh topology has become tangled. There could be multiple crossings, or a node may move down the order of nodes many places.

When the supposedly successful step has been made, find the pair of nodes x_i and x_{i+1}, that have the most negative value of $x_{i+1} - x_i$ at time $t+\Delta t$ (therefore, this is a max-norm), take the absolute value and call it Δx. Let the value of $x_{i+1} - x_i$ at time t be $\Delta x'$, where $\Delta x' > 0$. The larger the ratio of $\Delta x / \Delta x'$, the more there is an overstep in time, and therefore Δt must be decreased by a larger amount. Call the time step h and the time step that would not cross the nodes h'. If the node positions are assumed to move linearly (a conservative assumption) in a time step, then by similar triangles,

$$\frac{h-h'}{h'} = \frac{\Delta x}{\Delta x'}. \tag{53}$$

If

$$h' = fh \tag{54}$$

then the fraction f, by which the time step Δt must be reduced to maintain the node topology is given by

$$f = \frac{\Delta x'}{\Delta x + \Delta x'}. \tag{55}$$

It is reduced further to give a safety margin, $h' = \frac{1}{2} fh$. LSODI has an algorithm to determine automatically whether the order of the approximation should be increased or decreased, and what the optimal step size reduction or increase should be. We take advantage of this simply replacing the corrector error normally used in the determination of the possible change in step-size, for the same order of approximation, by using the factor f. This addition has not affected the performance of LSODI appreciably. If there is no code crossing the usual corrector error is used to determine step-size changes.

It is important to stress a number of points regarding the penalty functions. Node crossing generally occurs when a solution is forming a shock. The penalty functions will prevent nodes from coming too close together but once the nodes have crossed there are many solutions to the stationary conditions associated with the node topologies. The penalty functions are maintaining the solutions within a local region constrained by the initial node ordering. The reduction in time-step ensures that the solution does not leave the local region. If there were no penalty functions the integrator would continue to allow the nodes to cross since this would be a valid solution to the error minimization. For a hyperbolic problem, Wathen's jump conditions [17, 18] at the singularity would be an alternative. However, even in Wathen's approach, as noted by Wathen, a mechanism for reducing the time-step is necessary.

5. Examples Using the Basic Method

The set of subroutines called LSODI was used to solve the set of ordinary differential equations resulting from the application of moving finite elements. The only modification made to LSODI was that previously mentioned to account for node crossing. Many examples of other systems are presented in the work of Miller, Djomehri, and Gelinas, using their special codes, showing the accurate solutions possible with MFEM. Implementation of MFEM with LSODI is very straightforward. The penalty function form we used was

$$X_i^2 = \frac{C_1^2}{\Delta x_i - \delta}, \tag{56}$$

$$X_i \omega_i = \frac{C_2^2}{(\Delta x_i - \delta)^2}, \tag{57}$$

where C_1 was 10^{-2}, ten times the ODE solver truncation error requested which was 10^{-3}. C_2^2 was set at 10^{-8}, so that its effect would be negligible. The minimum approach distance was set at half the minimum shock width of our test problems, $\delta = 5 \times 10^{-5}$. These values were used for all MFE examples.

The test equation was the viscous Burger's equation

$$u_t = -uu_x + \mu u_{xx} \tag{58}$$

with $\mu = 10^{-3}$, 10^{-4}. This equation is a simple analogue to the Navier–Stokes equations and serves as a model for many nonlinear wave propagation problems.

Two sets of initial conditions were used.

Case A. Step function

$$x \in [0, 2],$$

$$\begin{aligned} u(x, 0) &= 0, & x &\in [0.00, 0.48], \\ &= \text{linear}, & x &\in [0.48, 0.52], \\ &= 1, & x &\in [0.52, 1.48], \\ &= \text{linear}, & x &\in [1.48, 1.52], \\ &= 0, & x &\in [1.52, 2.00], \end{aligned}$$

$$u(0, t) = u(2, t) = 0, \qquad t \geqslant 0.$$

Case B. Sine function

$$x \in [0, 1],$$
$$u(x, 0) = \sin(2\pi x) + \tfrac{1}{2}\sin(\pi x),$$
$$u(0, t) = u(1, t) = 0, \qquad t \geqslant 0.$$

The first set of initial conditions is very difficult to solve because of the sharp gradients present in the initial conditions. The second set of initial conditions are smooth and so conventional fixed mesh methods are adequate up to the point where the shock width becomes comparable to the grid spacing. Beyond that point the solution deteriorates rapidly.

All cases were solved for $t \in [0, 1]$. This was sufficient to see a shock produced with a width of $O(\mu)$. Figures 1–7 are profiles for different cases at $t = 0.5$.

Solving case A with a fixed mesh Galerkin finite element formulation with 100 nodes using LSODI for the time integration produced solutions as shown in Figs. 1a and 1b. For the case of $\mu = 10^{-3}$ the solution has some oscillations but one can discern the profile, while with $\mu = 10^{-4}$ the results are quite unacceptable.

The same problem is solved using moving finite elements with 4, 11, and 21 nodes for $\mu = 10^{-3}$ and 10^{-4}, with the results shown in Figs. 2a–c, and 3a–c, respectively. Note that 4 nodes is the absolute minimum number of nodes needed to mark off the initial step profile. With a tenfold decrease in μ the MFEM is still able to handle the solution with only 11 nodes. The initial positions of the nodes x_i were,

4 pts.	0.48, 0.52, 1.48, 1.52,
11 pts.	0.20, 0.48, 0.50, 0.52, 1.00, 1.48, 1.50, 1.52, 1.60, 1.70, 1.80,
21 pts.	0.20, 0.40, 0.48, 0.50, 0.52, 0.60, 0.70, 0.80, 0.90, 1.00, 1.10, 1.20, 1.30, 1.40, 1.48, 1.50, 1.52, 1.60, 1.70, 1.80, 1.90.

Table I presents some statistics from the runs for Case A. Each triple in the table presents the number of time steps, the number of Jacobian evaluations, and the time (in DEC-20 CPU sec) required for solution. For GFEM we present both the times for analytical and finite difference Jacobians; the difference is negligible. Finite differences are used to calculate the Jacobians used in the MFEM. With 11 nodes the MFEM gives very good solutions for about twice the computer time of the Gakerkin solution, but with much better solution accuracy. To obtain a fixed mesh solution with the same order of accuracy would require about ten times as many nodes, and this would require, using LSODI, more than ten times as much com-

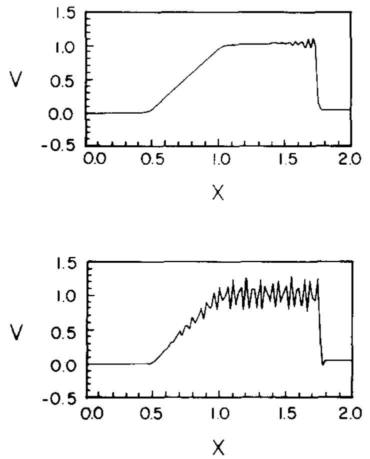

FIG. 1. Galerkin finite element solution on a fixed mesh of 100 nodes, at time $t = 0.5$ for step pulse initial conditions; (a) $\mu = 10^{-3}$, (b) $\mu = 10^{-4}$.

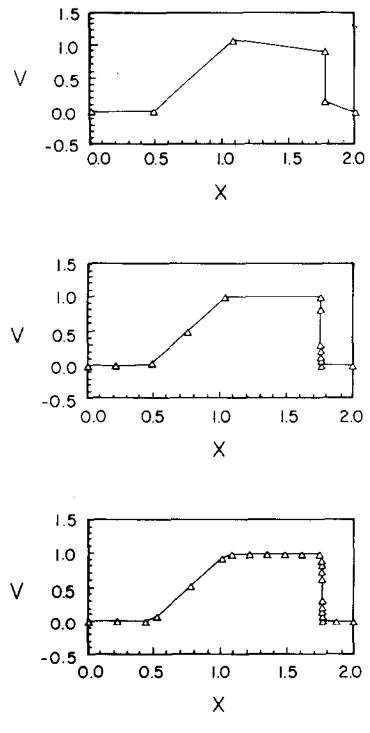

FIG. 2. Moving finite element solution, at time $t = 0.5$ for step pulse initial conditions, $\mu = 10^{-3}$; (a) 4 nodes, (b) 11 nodes, (c) 21 nodes.

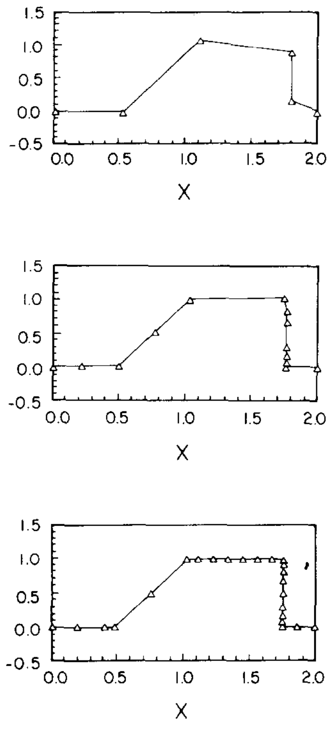

FIG. 3. Moving finite element solution, at time $t = 0.5$ for step pulse initial conditions, $\mu = 10^{-4}$; (a) 4 nodes, (b) 11 nodes, (c) 21 nodes.

TABLE I

Computational Statistics for the Numerical Solution of $u_t = -uu_x + \mu u_{xx}$ for Step Pulse Initial Conditions for Time Interval $[0, 1]$[a]

	GFEM-100 points		MFEM		
	Analytical Jacobian	Difference Jacobian	4 pts	11 pt	21 pts
$\mu = 10^{-3}$	107	107	66	150	208
	10	10	14	119	158
	10.8	11.0	3.4	22.7	52.7
$\mu = 10^{-4}$	142	142	75	182	315
	9	9	57	139	235
	11.7	11.9	4.2	24.6	79.2

[a] Table values: time-steps, Jacobian evaluations, DEC-20 CPU sec.

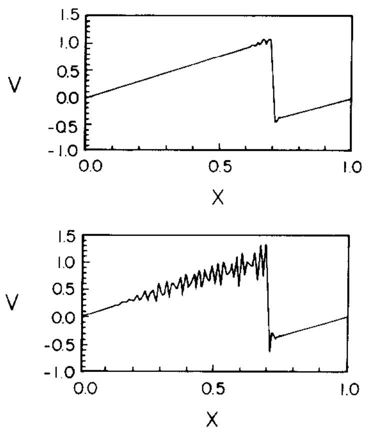

FIG. 4. Galerkin finite element solution on a fixed mesh of 100 nodes, at time $t=0.5$ for sinusoidal initial conditions; (a) $\mu=10^{-3}$, (b) $\mu=10^{-4}$.

puter time. Note that while the number of time steps for 11 points is 50% more than the 100 point fixed grid solution, the number of Jacobian evaluations is 10 times more. Since the bandwidth of MFEM formulation for one equation is 3 as opposed to 1 for the fixed grid solution, there is a higher cost for a Jacobian evaluation.

For case B, a 100 point fixed mesh GFEM is used with $\mu=10^{-3}$ and 10^{-4}, with the results shown in Figs 4 a–b. MFEM was used with 9 and 19 nodes, for $\mu=10^{-3}$, 10^{-4}, and the results are shown in Figs. 5a–c, 6a–c. The initial positions of the grid points x_i were determined by evenly dividing the interval [0, 1]. Table II shows that while a 9 point MFEM gives excellent results, it requires only a little more computer time than the GFEM.

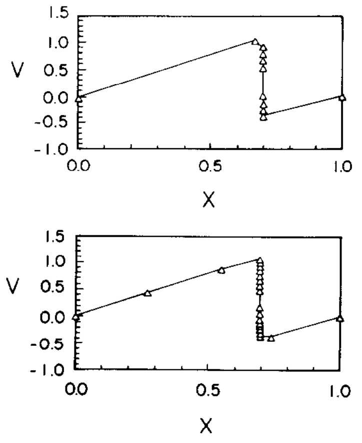

FIG. 5. Moving finite element solution, at time $t=0.5$ for sinusoidal conditions, $\mu=10^{-3}$; (a) 9 nodes, (b) 19 nodes.

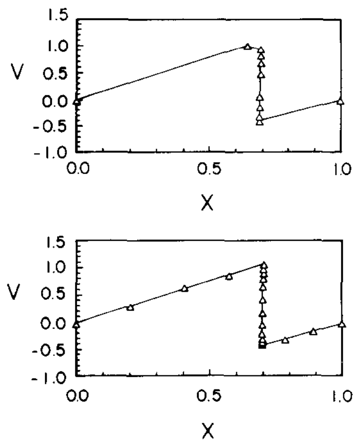

FIG. 6. Moving finite element solution, at time $t = 0.5$ for sinusoidal initial conditions, $\mu = 10^{-4}$; (a) 9 nodes, (b) 19 nodes.

The working vector space required by LSODI for the fixed mesh Galerkin formulation is $13n$, where n is the number of nodes for one PDE. For moving finite elements the storage is $38n$, but here n is much smaller. Typically MFEs used less than half the storage of the Galerkin mesh. As the Galerkin fixed mesh grows to obtain a lower residual error, the savings magnify.

In summary, we found that a straightforward implementation of the moving finite element method with LSODI lead to reductions in computer time and storage for comparably accurate solutions. Coupled with the fact that the penalty terms can be specified *a priori* the MFEM offers a viable alternative to solution of difficult practical problems.

TABLE II

Computational Statistics for the Numerical Solution of $u_t = -uu_x + \mu u_{xx}$ for Sinusoidal Initial Conditions for Time Interval $[0, 1]$[a]

	GFEM-100 points		MFEM		
	Analytical Jacobian	Difference Jacobian	9 pts	19 pts	19/5
$\mu = 10^{-3}$	165	165	178	230	
	27	27	113	177	
	16.2	17.2	17.8	53.9	
$\mu = 10^{-4}$	206	206	250	273	122
	15	15	173	194	72
	16.8	17.3	26.9	60.2	26.9

[a] Table values: time-steps, Jacobian evaluations, DEC-20 CPU sec.

6. Fewer Degrees of Freedom

The previous section has shown that MFEM, for the same required accuracy, would take less time than a Galerkin method of lines approach. The question arises if it is possible to reduce the computer time further. As noted above, much of the time in MFEM is due to the highly nonlinear Jacobian. One way to reduce the time would be to reduce the number of moving nodes. However, even in the MFEM there is a point where the solution is not accurate (see Fig. 2a and 3a). One could update parts of the Jacobian rather than the whole matrix. Updating could be done by checking where the greatest changes have occurred in nodal positions and/or amplitudes. Similarly one could lump the Jacobian and only calculate diagonal blocks in the corrector iterations of the temporal solver. The idea of only updating parts of the Jacobian would require changing LSODI, which would require specialized programming and defeat the purpose of testing implementations that require minimum additional work. Another idea would be to use an operator splitting notion where certain parts of the domain or the governing equations are integrated more often, and the whole system integrated together after many of the smaller time-steps. Also, to save storage one could go to an iterative solver in the integrator.

Since it is desirable to continue using LSODI the nonlinearity in the Jacobian is attacked directly. The moving finite elements are formulated so that only some of the node motions are directly minimizing the error, while the rest follow trajectories determined by criteria other than making the residual error orthogonal to some node motion.

One could have some of the nodes fixed and others moving, but this would require a great deal of *a priori* knowledge about the solution. Another idea would have a subset of the nodes move and require a certain proportional spacing between the remainder of the nodes which are between the moving nodes, but this would lead to discontinuous grid motions (though it would reduce the number of ODEs). From the analysis of the effect of penalty functions in the degenerate case, it can be seen that often node velocities are implicitly linked, but without regard to the residual error. The node positions continue to be smooth functions of time.

If I is a subset of nodes chosen to minimize the residual error directly, then the MFEM equations are of the form

$$(\alpha_i, R) = 0, \qquad i = 1,\dots, n, \tag{59}$$

$$(\beta_j, R) = 0, \qquad j \in I. \tag{60}$$

Penalty terms are included for the subset of nodes I minimizing the residual to maintain some control over mesh spacing. The rest of the nodes motions will be determined by, for example,

$$\dot{x}_i = \frac{\dot{x}_{i+1} + \dot{x}_{i-1}}{2}, \tag{61}$$

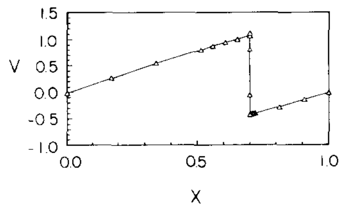

FIG. 7. Moving finite element solution, at time $t=0.5$ for sinusoidal initial conditions, $\mu=10^{-4}$, 19 nodes with 3, 7, 10, 13, and 17 used to minimize the residual.

i.e., the node velocities are directly linked by averaging adjacent node velocities. There are still $2n$ ODEs, but the β-equations for many of them have been simplified.

Burger's equation is solved using case B, again with 19 moving nodes, and $\mu=10^{-4}$. Nodes 3, 7, 10, 13, and 19 are arbitrarily picked so that for these nodes

$$(\beta_i, R)=0, \qquad i=3, 7, 10, 13, 17$$

and for the remainder

$$-\dot{x}_{i-1}+2\dot{x}_i-\dot{x}_{i+1}=0.$$

The solution shown for $t=0.5$ in Fig. 7 is almost identical in profile to Fig. 6c, though the node positions are different. The computer time, as shown in Table II under 19/5 is drastically reduced to 26.9 CPU seconds. This is about the same time as for fewer MFE nodes, but still gives the accuracy of a 19 node solution.

This result begged the question of how many grid points are needed as degrees of freedom for the residual error minimization. When 3 nodes were used as degrees of freedom the solution showed some oscillation within the superelements, like that of a fixed mesh procedure. Five nodes worked very well. Logically only one degree of freedom node is needed to follow the shock, with other node velocities determined by that node's velocity. However, more nodes are needed because of the simple averaging of the nodal velocities used, not taking into account gradient or curvature information. Therefore, some of the degrees of freedom monitor the changes in curvature, and others resolve the shock.

This idea could be used to partition nodes so that a certain set of nodes only provides a solution in a distinct zone, which may be of some use in multiple dimensions where there is recirculation.

When using a subset of nodes as independent degrees of freedom the penalty functions become much less important. In many cases the penalty functions are not needed at all. However, the penalty functions are needed whenever the slope may be constant over two whole superelements, i.e., all elements between two independently moving nodes $j-1$ and j have the same slope as the elements between the independently moving nodes j and $j+1$, which is true when there are many independently nodes or large piecewise linear segments in the solution. Note that in Wathen's approach removal of the penalty terms is accomplished by adding the fixed node equation and then moving a singular node to a desired relative position when the singular set of equations is detected at a particular time-step. These procedures are equivalent to our method of using fewer independent node positions as degrees of freedom to the problem which we feel is an appropriate way to view the numerical solution of the physical system, except we use fewer independently moving nodes throughout the total time interval.

7. Conclusions

The moving finite element method provides a viable means to reduce the computer time necessary to simulate problems with steep fronts. The MFEM has been implemented directly using a readily available integration package, and very good results have been obtained both in terms of accuracy and simulation cost. Analysis and review of the current moving finite element literature was carried out to determine what might be stumbling blocks for an easy implementation of the method. A modification to the algorithm has been proposed which will further reduce the computer time necessary to obtain an accurate solution. We are presently working to apply this method to practical chemical engineering simulation and design problems where the analysis cost of these distributed parameter systems is significant.

Acknowledgments

The authors gratefully acknowledge support from the Exxon Foundation. A. Hrymak was also the recipient of an Imperial Oil Graduate Research Fellowship. G. McRae was supported by the Dow Chemical Company and by a Presidential Young Investigator Award from the National Science Foundation under Grant CEE-83-52506. We thank Dr. R. Gelinas for bringing the unpublished work of Wathen *et al.* to our attention.

References

1. R. A. Adams, "Sobolev Spaces," Sect. 2.17–2.19, Academic Press, New York,1975.
2. R. Alexander, P. Manselli, and K. Miller, Moving finite elements for the Stefan problem in two dimensions, *Atti Accad. Naz. Lincei Mem. Cl. Sci. Fiz. Mat. Nat. Sez. Ren.* **67** (1979), 57–61.
3. J. Djomehri and K. Miller, "A moving Finite Element Code for General Systems of PDEs in 2-D," Report PAM-57, Center for Pube and Applied Mathematics, Univ. of California, Berkeley, October 1981.
4. J. Djomehri, "Moving Finite Element Solution of Systems of Partial Differential Equations in 1-Dimension," Ph. D. thesis, Department of Mathematics, Univ. of California, Berkeley, December [illegible]

[illegible]

6. R. J. Gelinas and S. K. Doss, "Fourth IMACS International Symposium on Computer Methods for Partial Differential Equations," Lehigh University, Bethlehem, Penn., June 30–July 2, 1981.
7. R. J. Gelinas and S. K. Doss, *in* "Tenth IMACS Congress," Montreal, Canada, August 8–13, 1982.

8. R. J. GELINAS, S. K. DOSS, J. P. VAJK, J. DJOMEHRI, AND K. MILLER, *in* "Tenth IMACS Congress," Montreal, Canada, August 8–13, 1982.
9. B. M. HERBST, S. W. SCHOOMBIE, AND A. R. MITCHELL, *Int. J. Numer. Methods Eng.* **18** (1982), 1321.
10. B. M. HERBST, S. W. SCHOOMBIE, AND A. R. MITCHELL, *J. Comput. Appl. Math.* **9** (1983), 337.
11. B. M. HERBST, S. W. SCHOOMBIE, D. F. GRIFFITHS, AND A. R. MITCHELL, *Int. J. Numer. Methods Eng.* **20** (1984), 1273.
12. A. C. HINDMARSH, *ACM-Signum Newslett.* **15** (1980), 10.
13. D. R. LYNCH, *J. Comput. Phys.* **47** (1982), 387.
14. K. MILLER AND R. N. MILLER, *SIAM J. Numer. Anal.* **18** (1981), 1019.
15. K. MILLER, *SIAM J. Numer. Anal.* **18** (1981), 1033.
16. K. MILLER, *in* "Adaptive Computational Methods for Partial Differential Equations," (I. Babuska, J. Chandra, and J. E. Flaherty, Eds.), pp. 165–182, SIAM, Philadelphia, Penn., 1983.
17. A. J. WATHEN AND M. J. BAINES, "On the Structure of the Moving Finite Element Equations," Numerical Analysis Report, No. 5/83, University of Reading, 1983.
18. A. J. WATHEN, "Moving Finite Elements and Oil Reservoir Modelling," Ph. D. thesis, Department of Mathematics, Univ. of Reading, May 1984.