INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS
Int. J. Numer. Meth. Fluids 2000; **33**: 789–832

The Lagrange–Galerkin method for the two-dimensional shallow water equations on adaptive grids

Francis X. Giraldo*[1]

Naval Research Laboratory, Monterey, CA, U.S.A.

SUMMARY

The weak Lagrange–Galerkin finite element method for the two-dimensional shallow water equations on adaptive unstructured grids is presented. The equations are written in conservation form and the domains are discretized using triangular elements. Lagrangian methods integrate the governing equations along the characteristic curves, thus being well suited for resolving the non-linearities introduced by the advection operator of the fluid dynamics equations. An additional fortuitous consequence of using Lagrangian methods is that the resulting spatial operator is self-adjoint, thereby justifying the use of a Galerkin formulation; this formulation has been proven to be optimal for such differential operators. The weak Lagrange–Galerkin method automatically takes into account the dilation of the control volume, thereby resulting in a conservative scheme. The use of linear triangular elements permits the construction of accurate (by virtue of the second-order spatial and temporal accuracies of the scheme) and efficient (by virtue of the less stringent Courant–Friedrich–Lewy (CFL) condition of Lagrangian methods) schemes on adaptive unstructured triangular grids. Lagrangian methods are natural candidates for use with adaptive unstructured grids because the resolution of the grid can be increased without having to decrease the time step in order to satisfy stability. An advancing front adaptive unstructured triangular mesh generator is presented. The highlight of this algorithm is that the weak Lagrange–Galerkin method is used to project the conservation variables from the old mesh onto the newly adapted mesh. In addition, two new schemes for computing the characteristic curves are presented: a composite mid-point rule and a general family of Runge–Kutta schemes. Results for the two-dimensional advection equation with and without time-dependent velocity fields are illustrated to confirm the accuracy of the particle trajectories. Results for the two-dimensional shallow water equations on a non-linear soliton wave are presented to illustrate the power and flexibility of this strategy. Copyright © 2000 John Wiley & Sons, Ltd.

KEY WORDS: characteristic-based; Eluerian–Lagrangian; finite element; semi-Lagrangian; unstructured grid

* Correspondence to: Naval Research Laboratory, 7 Grace Hopper Avenue, Stop 2, Building 704-245, Monterey, CA 93943-5502, U.S.A.
[1] E-mail: giraldo@nrlmry.navy.mil

Received 16 February 1999
Revised 12 August 1999

1. INTRODUCTION

The advection terms in the governing equations of fluid motion present formidable challenges to many spatial discretization methods, including Galerkin methods. These terms prevent the operator from being self-adjoint and as a result, Galerkin methods are no longer optimal for the spatial discretization. Researchers have tried circumventing this problem by using high-order Eulerian methods and characteristic-based Lagrangian methods. In this section a brief overview of some of the more interesting methods is given, identifying their weaknesses and assessing their strengths.

A limited number of new Eulerian methods have been introduced in recent years for solving hyperbolic partial differential equations (PDEs) (such as the shallow water equations), perhaps the best being the spectral element method [1–3]. This method offers high accuracy solutions and spectral convergence (provided the solution is smooth) but at the price of having to use small time steps (due to the explicit solvers typically used) and structured grids (due to the restriction that the elements be quadrilaterals). Recently, Giraldo [4] showed how to increase the time step by combining the Lagrange–Galerkin method with the spectral element method, while Taylor and Wingate [5] have been working on developing a triangular spectral element method.

In contrast, there exists a multitude of characteristic-based methods. Eulerian–Lagrangian methods combine the standard finite element procedure with the method of characteristics. By integrating the equation in time along the characteristics the resulting spatial operator becomes self-adjoint, which then justifies the use of Galerkin spatial discretizations. These methods can be classified into the following three groups: the direct (or strong) Lagrange–Galerkin method, the weak Lagrange–Galerkin method, and the Eulerian–Lagrangian localized adjoint method. In the strong Lagrange–Galerkin method, the time derivative and the advection terms are combined into the Lagrangian derivative and the resulting operator is then discretized using the standard finite element method. This is the approach used by, for example, Bercovier and Pironneau [6], Bermejo [7], Douglas and Russell [8], and Priestley [9]. In this method the basis functions are the typical Lagrange polynomials used in the standard finite element method, which are only dependent on the spatial co-ordinates. The success of this method hinges on determining the values at the feet of the characteristics. Related to this method are the semi-Lagrangian [10] and characteristic-Galerkin [11] methods (see also the Taylor–Galerkin method [12]). The semi-Lagrangian method is essentially the strong Lagrange–Galerkin method with the exception that the spatial discretization is achieved through finite differencing; this method is quite ubiquitous in the meteorology community. In this method, the values of the variables at the feet of the characteristics are obtained through interpolation. The characteristic-Galerkin method avoids interpolations by using approximations obtained from Taylor series expansions. Therefore, the computations all take place in terms of the Eulerian grid, which eliminates the difficulties of the Lagrangian grid but at the expense of having to use smaller time steps due to the stricter stability restrictions that govern all Eulerian methods.

In contrast, the weak Lagrange–Galerkin method uses basis functions that are dependent on both space and time by forcing the basis functions to disappear along the adjoint of the transport operator (the transport operator referring to a homogeneous conservation law); this results in the basis functions vanishing along the characteristics. Using the conservation form of the governing equations simplifies the resulting discretization by introducing the Reynolds' transport theorem [13], which then eliminates the boundary terms arising from the integration by parts used to obtain the adjoint of the transport operator. This is the method introduced by Benque *et al.* [14,15] and the approach used in this paper.

Finally, the Eulerian–Lagrangian localized adjoint method forces the basis functions to disappear along the adjoint of the entire differential operator (not just the transport operator). Since the Reynolds' transport theorem is not used in this formulation, all of the boundary terms remain, which makes it rather easy to enforce any type of boundary conditions. This is the method introduced by Celia *et al.* [16] and used in References [17–20]. Thus, for applications where the boundaries play an important role it would seem prudent to use this method. However, since the final objective of our work is to develop a shallow water model on the sphere, where there are no boundary conditions, the weak Lagrange–Galerkin method appears to be the perfect candidate for such a model.

Lagrange–Galerkin methods have increased in popularity in the last 10 years because they offer increased accuracy and efficiency by virtue of their independence on the Courant–Friedrich–Lewy (CFL) condition; however, almost all research has involved the strong method as opposed to the weak method presented here. Currently the only applications (of which this author is aware) using the weak method or variations thereof are the following: two-dimensional advection [14,21], two-dimensional shallow water on the plane [22], two-dimensional Navier–Stokes [15], and two-dimensional shallow water on the sphere [23].

There have been few attempts at using the weak Lagrange–Galerkin method for the shallow water equations. In Reference [22], the weak Lagrange–Galerkin method of Benque [15] is presented for estuary flows on structured quadrilateral grids. In their paper, the interpolations for the departure point values at the vertices of the elements are bilinear and the variables inside their respective Lagrangian element are then assumed continuous and linearly varying throughout the element. Assuming a linear variation within the Lagrangian element results in a highly stable scheme but a strongly diffusive one as well. In our paper, triangular grids are used and the integration can either be done piecewise exactly or numerically. Neither approach assumes a linear variation within the element but rather takes into consideration the gradient across the different Eulerian element that the Lagrangian element spans.

In Reference [23] an exactly integrated weak Lagrange–Galerkin method on the spherical shallow water equations is presented. The basis functions used are not the natural co-ordinates derived for linear triangles. In our paper, the natural co-ordinates are used. This means that we can not only derive explicit basis function formulas that can be used for computing derivatives or searching for inclusion of departure points, but also they can be used to construct exact integration formulas for all of the finite element integrals including those occurring at the Lagrangian elements. In addition, the use of the explicit natural co-ordinates has ramifications for higher-dimensional models. All of the algorithms presented in this paper are directly generalizable to a two-dimensional spherical or three-dimensional Cartesian shallow water model. This has been the focus of our previous [21] and future work. Therefore, the idea of using the natural co-ordinates on the triangle in two-dimensional space is directly applicable to a spherical model. For a full three-dimensional model, rather than using the linear triangular natural co-ordinates, the linear tetrahedral natural co-ordinates are used with little change to the algorithms presented in this paper.

The method presented in Reference [22] uses quadrilaterals instead of triangles, which are used in this paper. There are many advantages in using triangles over quadrilaterals. Since the triangle is the two-simplex, any two-dimensional domain can be discretized with triangles no matter how multiply-connected (i.e., non-convex) it is. In addition, it is irrelevant to the scheme whether the flow has strong shears or is highly rotational. If quadrilaterals are used, the Lagrangian elements may become highly skewed or completely distorted, whereas this is of no concern with triangles. Using quadrilateral elements also requires a non-linear iterative solver in order to compute the local element co-ordinates from the global co-ordinates obtained from the trajectory equation. This is required because the interpolations within an element must be carried out using the local co-ordinates because the local basis functions are defined only in terms of these co-ordinates and not in terms of the global co-ordinates. On triangles, because the basis functions are defined in terms of the global co-ordinate, this procedure is not required. Furthermore, in this paper we show how to apply the weak method with adaptive unstructured grids, which can only be accomplished with triangles. The idea of using adaptive unstructured grids with Lagrange–Galerkin methods is attractive because the resolution of the grid can be increased without the need to decrease the time step. In Eulerian methods, the time step must be decreased in order to satisfy the CFL condition; violating the CFL condition results in instabilities and inaccuracies. In Lagrangian methods, on the other hand, the CFL condition is less stringent and typically the time step need not be decreased. Numerical tests are presented that show the effects of increasing the time step on the accuracy of the scheme.

2. SHALLOW WATER EQUATIONS

The two-dimensional planar shallow water equations in conservation form are

$$\frac{\partial}{\partial t}\begin{bmatrix}\varphi\\ \varphi u\\ \varphi v\end{bmatrix}+\frac{\partial}{\partial x}\begin{bmatrix}\varphi u\\ \varphi u^2+\frac{1}{2}\varphi^2\\ \varphi uv\end{bmatrix}+\frac{\partial}{\partial y}\begin{bmatrix}\varphi v\\ \varphi uv\\ \varphi v^2+\frac{1}{2}\varphi^2\end{bmatrix}=\begin{bmatrix}0\\ +f\varphi v\\ -f\varphi u\end{bmatrix}$$

but note that if we move the pressure terms to the right-hand side, we get

$$\frac{\partial}{\partial t}\begin{bmatrix}\varphi\\ \varphi u\\ \varphi v\end{bmatrix}+\frac{\partial}{\partial x}\begin{bmatrix}\varphi u\\ \varphi u^2\\ \varphi uv\end{bmatrix}+\frac{\partial}{\partial y}\begin{bmatrix}\varphi v\\ \varphi uv\\ \varphi v^2\end{bmatrix}=\begin{bmatrix}0\\ -\varphi(\partial\varphi/\partial x)+f\varphi v\\ -\varphi(\partial\varphi/\partial y)-f\varphi u\end{bmatrix} \tag{1}$$

This system can now be written more compactly as

$$\frac{\partial \mathbf{f}}{\partial t}+\nabla\cdot(\mathbf{fu})=\mathbf{S}(\mathbf{f}) \tag{2}$$

where

$$\mathbf{f} = \begin{bmatrix} \varphi \\ \varphi u \\ \varphi v \end{bmatrix}, \qquad \mathbf{u} = \begin{bmatrix} u \\ v \end{bmatrix}, \quad \text{and} \quad \mathbf{S(f)} = \begin{bmatrix} 0 \\ -\varphi(\partial\varphi/\partial x) + f\varphi v \\ -\varphi(\partial\varphi/\partial y) - f\varphi u \end{bmatrix} \tag{3}$$

The equations are solved for the three conservation variables φ, φu, and φv. Clearly, we could also include other forcing functions within **S(f)**. Before proceeding to the discretization of the weak Lagrange–Galerkin method, let us first look at the discretization obtained by an Eulerian method. This will serve both for contrast and also because we require one Eulerian time step prior to using the weak Lagrange–Galerkin method because it requires two known time steps.

3. EULER–GALERKIN METHOD

Beginning with Equation (2) we can define an Eulerian finite element method by taking the weak form

$$\int_\Omega \psi \left[\frac{\partial \mathbf{f}}{\partial t} + \nabla \cdot (\mathbf{fu}) - \mathbf{S(f)} \right] \mathrm{d}\Omega = 0$$

and integrating by parts (Green's theorem) such that

$$\nabla \cdot (\psi \mathbf{fu}) = \psi \nabla \cdot (\mathbf{fu}) + (\mathbf{fu}) \cdot \nabla \psi$$

to arrive at

$$\int_\Omega \psi \frac{\partial \mathbf{f}}{\partial t} \, \mathrm{d}\Omega + \int_\Gamma \mathbf{n} \cdot \mathbf{u} \psi \mathbf{f} \, \mathrm{d}\Gamma - \int_\Omega \nabla \psi \cdot (\mathbf{fu}) \, \mathrm{d}\Omega - \int_\Omega \mathbf{S(f)} \, \mathrm{d}\Omega = 0$$

In the case of no-flux boundary conditions, the second integral vanishes. In other words

$$\int_\Gamma \mathbf{n} \cdot \mathbf{u} \psi \mathbf{f} \, \mathrm{d}\Gamma = 0$$

and we are left with

$$\int_\Omega \psi \frac{\partial \mathbf{f}}{\partial t} \, \mathrm{d}\Omega = \int_\Omega \nabla \psi \cdot (\mathbf{fu}) \, \mathrm{d}\Omega + \int_\Omega \mathbf{S(f)} \, \mathrm{d}\Omega$$

The resulting system of ordinary differential equations (ODEs) can now be written

$$\frac{d\mathbf{U}}{dt} = H(\mathbf{U})$$

which, after integrating in time by a general family of Runge–Kutta schemes, yields

$$\mathbf{U}^{k+1} = \mathbf{U}^{n} + \Delta t \beta H(\mathbf{U})^{k-1}$$

where

$$\beta = \frac{1}{M-k+1}, \quad k = 1, \ldots, M \quad \text{and} \quad H(\mathbf{U})^{0} = H(\mathbf{U})^{n}$$

This scheme is required for the first time step because, as you will see, the weak Lagrange–Galerkin method although a two-time level scheme, requires two known times in order to be able to compute the particle trajectories accurately.

4. WEAK LAGRANGE–GALERKIN METHOD

4.1. Spatial discretization

The weak form of Equation (2) is

$$\int_{\Omega} \psi \left[\frac{\partial \mathbf{f}}{\partial t} + \nabla \cdot (\mathbf{f}\mathbf{u}) - \mathbf{S}(\mathbf{f}) \right] d\Omega = 0$$

and using the calculus identities

$$\frac{\partial}{\partial t}(\psi \mathbf{f}) = \psi \frac{\partial \mathbf{f}}{\partial t} + \mathbf{f} \frac{\partial \psi}{\partial t} \quad \text{and} \quad \nabla \cdot (\psi \mathbf{f}\mathbf{u}) = \psi \nabla \cdot (\mathbf{f}\mathbf{u}) + (\mathbf{f}\mathbf{u}) \cdot \nabla \psi$$

and integrating by parts results in

$$\int_{\Omega} \left[\frac{\partial}{\partial t}(\psi \mathbf{f}) + \nabla \cdot (\psi \mathbf{f}\mathbf{u}) \right] - \mathbf{f} \left[\frac{\partial \psi}{\partial t} + \mathbf{u} \cdot \nabla \psi \right] - [\psi \mathbf{S}(\mathbf{f})] \, d\Omega = 0 \tag{4}$$

The first bracketed term of Equation (4), using Reynolds' transport theorem [13], can be written as

$$\frac{d}{dt} \int_{\Omega} (\psi \mathbf{f}) \, d\Omega = \int_{\Omega} \left[\frac{\partial}{\partial t}(\psi \mathbf{f}) + \nabla \cdot (\psi \mathbf{f}\mathbf{u}) \right] d\Omega \tag{5}$$

and the second bracketed term is actually the characteristic equation

$$\frac{d\psi}{dt} \equiv \frac{\partial \psi}{\partial t} + \mathbf{u} \cdot \nabla \psi = 0 \tag{6}$$

where

$$\frac{d\mathbf{x}}{dt} = \mathbf{u} \tag{7}$$

is used to predict the particle trajectories along which the basis functions vanish. The characteristic equation is equal to zero because we are constraining the basis functions to be constant along the particle trajectories. Therefore, in essence, the basis functions ψ are functions of both space and time. This arises from only considering changes of the conservation variables along the trajectory lines (see References [15,22]). In addition, in Appendix A, we have included an example to show that the basis functions satisfy the characteristic equation.

Substituting Equations (5) and (6) into Equation (4) yields the weak Lagrange–Galerkin system

$$\frac{d}{dt} \int_{\Omega} (\psi \mathbf{f}) \, d\Omega = \int_{\Omega} \psi \mathbf{S}(\mathbf{f}) \, d\Omega \tag{8}$$

Note that the advection operator has disappeared from the equations; however, the correct particle trajectories are accounted for by the trajectory equation (7). In addition, consider that the divergence of the velocity has also disappeared or rather has been absorbed by Reynolds' transport theorem.

Remark 1
The implicit absence of the divergence term is deceiving because this term, which controls the dilation (contraction or expansion) of a fluid volume, is accounted for by virtue of the trajectory equation. Therefore, if the fluid volume dilates then it is exactly satisfied by the dilation of the Lagrangian element through the projection of the element grid point trajectories. This key point is the difference between the weak and strong Lagrange–Galerkin methods (see Reference [21] for details) and is the reason for which the weak method conserves better.

Figure 1 shows a schematic of the dilation of a fluid volume. The grid points E_1, E_2, and E_3 represent the vertices of the Eulerian element at the arrival time t^{n+1}. The trajectory equation is then solved in order to determine the particle trajectories of these three grid points. Let L_1, L_2, and L_3 represent the departure points of the Eulerian grid points. In other words, the points L correspond to where the points E resided at time t^n. Note that the areas of the triangles E and L are not necessarily equal, and this is where the dilation of the control volume is accounted for. In other words, since the trajectory equation contributes the advection portion of the equations this operator does not need to be included in the equations. In addition, since the change in areas of the control volumes obtained by the trajectory equation contributes the divergence normally given by the velocity, then this term also does not need to appear in the equations.

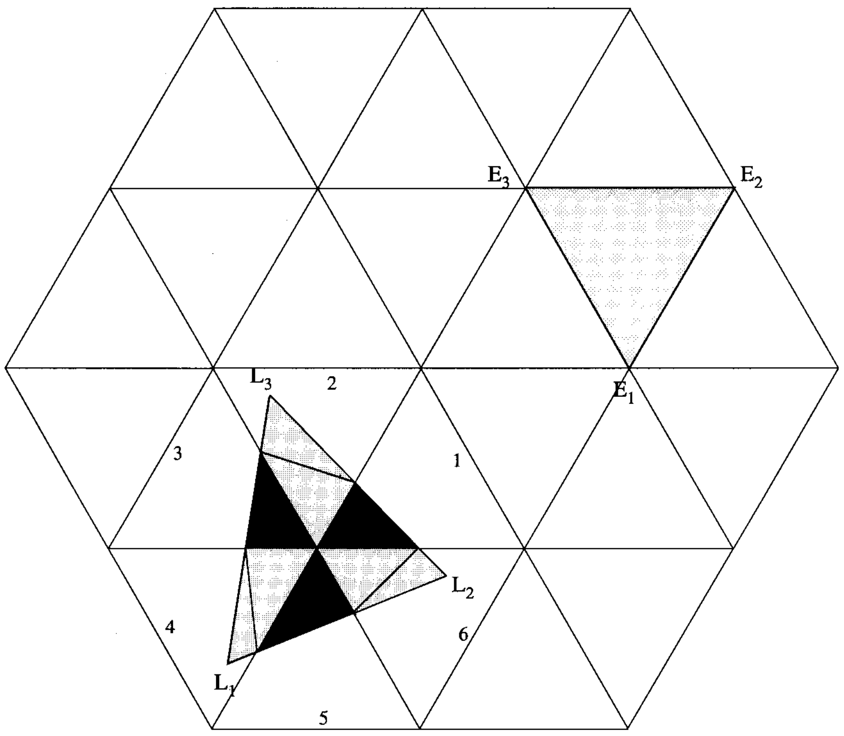


Figure 1. The dilation of the Eulerian element (E) area to the Lagrangian element (L) area traced by the particle trajectories. The numbers 1–6 refer to the six Eulerian elements surrounding the Lagrangian element.

4.2. Time discretization

Integrating Equation (8) in time by the θ algorithm yields

$$\int_{\Omega^{n+1}} (\psi \mathbf{f})\, \mathrm{d}\Omega^{n+1} = \int_{\Omega^{n}} (\psi \mathbf{f})\, \mathrm{d}\Omega^{n} + \Delta t \left[\theta \int_{\Omega^{n+1}} \psi \mathbf{S}(\mathbf{f})\, \mathrm{d}\Omega^{n+1} + (1-\theta) \int_{\Omega^{n}} \psi \mathbf{S}(\mathbf{f})\, \mathrm{d}\Omega^{n} \right] \quad (9)$$

which represents a two-time level scheme and gives the explicit rectangle rule, the trapezoid rule, and the implicit rectangle rule for $\theta = 0$, $\frac{1}{2}$, and 1 respectively. In this paper, $\theta = \frac{1}{2}$ is used throughout because it yields a second-order accurate scheme and is unconditionally stable with respect to advection. (Lagrangian methods are not unconditionally stable for advection with a forcing function, though; in this case, the Courant number must be chosen in order to satisfy ODE stability conditions, which are much less restrictive than their PDE counterparts.) The other two schemes are both first-order and the $\theta = 0$ is not very stable while the $\theta = 1$ is too diffusive (see Reference [24] for details).

The trajectory equation (7) is required for closure. Clearly, Equations (9) and (7) define a two-time level scheme requiring the variables at times t^n and t^{n+1}. However, in the following section, it is shown that in order to solve the trajectory equation (7) accurately, the velocity field at previous time levels is required. The accurate solution of the trajectory equation is perhaps the most important part of Lagrangian methods and is in fact often overlooked. In the next sections, we compare two new methods for calculating the particle trajectories.

4.2.1. Composite mid-point rule. In most Lagrangian methods, the particle trajectories are computed by the mid-point rule

$$\mathbf{x}^{n+1} - \mathbf{x}^n = \Delta t \mathbf{u}(\mathbf{x}^{n+1/2})$$

which requires the following extrapolation of the velocity field

$$\mathbf{u}^{n+1/2} = \frac{3}{2}\mathbf{u}^n - \frac{1}{2}\mathbf{u}^{n-1}$$

In addition, since we need to know $\mathbf{x}^{n+1/2}$, we have to assume that the trajectory from $\mathbf{x}^{n+1} \rightarrow \mathbf{x}^n$ is linear; in other words

$$\mathbf{x}^{n+1/2} = \frac{\mathbf{x}^{n+1} + \mathbf{x}^n}{2} \tag{10}$$

which then yields the iterative equation

$$\mathbf{x}^{n+1/2} = \mathbf{x}^{n+1} - \frac{\Delta t}{2}\mathbf{u}(\mathbf{x}^{n+1/2})$$

We begin with the arrival point $\mathbf{x}^{n+1}$ as the first approximation $\mathbf{x}^{n+1/2}$. Continuing in this fashion results in convergence in less than five iterations. This scheme is second-order accurate in space and time and is used quite extensively in the semi-Lagrangian community. (The semi-Lagrangian method when used in conjunction with the finite element is also known as the *strong* Lagrange–Galerkin method.)

However, the weakness of this scheme is that upon obtaining the correct mid-point trajectories, the departure points are then computed assuming a linear variation in the trajectories as shown in Equation (10). One simple way of improving this scheme is to construct a composite mid-point rule. The idea is to sub-divide the trajectory into multiple trajectories. Assuming we have M sub-divisions of the trajectory, we begin from time t^{n+1} and construct the following scheme:

$$\mathbf{x}^{n+1-\beta} = \mathbf{x}^{n+1-\alpha} - \frac{\Delta t}{2}\mathbf{u}(\mathbf{x}^{n+1-\beta})$$

where

$$\alpha = \frac{i-1}{M}, \quad \beta = \frac{i-(1/2)}{M}, \quad \gamma = \frac{i}{M} \quad \text{for } i = 1, \ldots, M$$

and the mid-point departure points are now given by

$$\mathbf{x}^{n+1-\gamma} = \frac{\mathbf{x}^{n+1-\alpha} + \mathbf{x}^{n+1-\beta}}{2}$$

This approach defines M series of second-order accurate steps in both space and time. Note that the velocity field must be extrapolated in order to get its value at time $t^{n+1-\beta}$. This is achieved by the following second-order extrapolation:

$$\mathbf{u}(t^{n+1-\beta}) = (2-\beta)\mathbf{u}(t^{n}) - (1-\beta)\mathbf{u}(t^{n-1}) \tag{11}$$

which then requires an interpolation to get the values $\mathbf{u}(\mathbf{x}^{n+1-\beta})$.

4.2.2. Runge–Kutta scheme. A higher-order approximation can be obtained by applying the Runge–Kutta scheme to the trajectory equation (7), yielding

$$\mathbf{x}^{k+1} = \mathbf{x}^{n+1} - \Delta t \beta \mathbf{u}(\mathbf{x})^{k-1}$$

where

$$\beta = \frac{1}{M-k+1}, \quad k = 1, \ldots, M \quad \text{and} \quad \mathbf{u}(\mathbf{x}) = \begin{cases} \mathbf{u}(\mathbf{x})^{n+1} & \text{for } k = 1 \\ \mathbf{u}(\mathbf{x})^{n+1-\beta} & \text{for } k > 1 \end{cases}$$

This requires knowing $\mathbf{u}(\mathbf{x}^{n+1-\beta})$, which can be extrapolated from the velocity fields at the previous time steps. We could use the extrapolation given in Equation (11) but this would formally give only second-order approximations for the velocity. Therefore, the most consistent approach is to use an extrapolation of equal order to that of the Runge–Kutta scheme. At the first Lagrange–Galerkin step (t^n) we already know the velocity at times t^{n-1} and t^{n-2} (because the first time step t^{n-1} is carried out with the Eulerian method and at t^{n-2} the intial conditions are used). Therefore, at this time step we can only use a maximum of a second-order scheme; for this reason we use a second-order extrapolation in conjunction with a second-order Runge–Kutta scheme ($M = 2$). However, at the next time step (t^{n+1}) we know the velocity at t^n, t^{n-1}, and t^{n-2}. We can do this one more time until we know the velocity field at sufficient time levels in order to obtain a fourth-order scheme. The extrapolations of the velocity field at time $t^{n+1-\beta}$ then can be obtained from a Taylor series expansion giving

$$\mathbf{u}(t^{n+1-\beta}) = (1+a+b+c)\mathbf{u}(t^{n}) + a\mathbf{u}(t^{n-1}) + b\mathbf{u}(t^{n-2}) + c\mathbf{u}(t^{n-3}) \tag{12}$$

where

$$\begin{array}{lllll}
a=\delta & b=0 & c=0 & \text{for} & M=2\\
a=2\delta+\delta^2 & b=-\frac{1}{2}\delta-\frac{1}{2}\delta^2 & c=0 & \text{for} & M=3\\
a=3\delta+\frac{5}{2}\delta^2+\frac{1}{2}\delta^3 & b=-\frac{3}{2}\delta-2\delta^2-\frac{1}{2}\delta^3 & c=\frac{1}{3}\delta+\frac{1}{2}\delta^2+\frac{1}{6}\delta^3 & \text{for} & M=4
\end{array}$$

and

$$\delta=1-\beta$$

which means that after four time steps, the scheme will be running at fourth-order accuracy throughout the remainder of the time integration. Note that this scheme does not require a first guess starting point as in the previous scheme because this is not an iterative scheme but rather a multi-step scheme. Therefore, we begin from the arrival point, as in the previous scheme, and step through towards the mid-point. The advantage of this scheme over the composite mid-point rule is that the Runge–Kutta scheme never assumes a linear trajectory, and is in fact inherently non-linear. The superiority of this scheme over the composite mid-point rule becomes evident when very large time steps are taken. Because of its non-linear structure, and its high-order approximation the Runge–Kutta scheme is better equipped to compute more accurate trajectories than the mid-point rule especially when large changes of the velocity field with respect to time occur.

4.3. Element equations

Returning to Equation (9) and substituting in the conservation variables from Equation (3) gives

$$\begin{aligned}
\int_{\Omega^{n+1}} \psi \begin{bmatrix} \varphi \\ \varphi u \\ \varphi v \end{bmatrix}^{n+1} \mathrm{d}\Omega^{n+1} = \int_{\Omega^{n}} \psi \begin{bmatrix} \varphi \\ \varphi u \\ \varphi v \end{bmatrix}^{n} \mathrm{d}\Omega^{n} \\
+\Delta t\theta \int_{\Omega^{n+1}} \psi \begin{bmatrix} 0 \\ -\varphi(\partial\varphi/\partial x)+f\varphi v \\ -\varphi(\partial\varphi/\partial y)-f\varphi u \end{bmatrix}^{n+1} \mathrm{d}\Omega^{n+1} \\
+\Delta t(1-\theta) \int_{\Omega^{n}} \psi \begin{bmatrix} 0 \\ -\varphi(\partial\varphi/\partial x)+f\varphi v \\ -\varphi(\partial\varphi/\partial y)-f\varphi u \end{bmatrix}^{n} \mathrm{d}\Omega^{n} \qquad (13)
\end{aligned}$$

which is the equation to be solved within each element. The mass and momentum equations can be decoupled as follows:

the mass equation is given as

$$\int_{\Omega^{n+1}} \psi(\varphi)\, d\Omega^{n+1} = \int_{\Omega^n} \psi(\varphi)\, d\Omega^n$$

which is no longer a function of velocity except through the trajectory equation, the momentum equations are

$$\int_{\Omega^{n+1}} \psi(\varphi u)\, d\Omega^{n+1} - \Delta t\theta \int_{\Omega^{n+1}} \psi(f\varphi v)\, d\Omega^{n+1}$$
$$= \int_{\Omega^n} \psi(\varphi u)\, d\Omega^n + \Delta t(1-\theta) \int_{\Omega^n} \psi\left(-\varphi \frac{\partial \varphi}{\partial x} + f\varphi v\right) d\Omega^n + \Delta t\theta \int_{\Omega^{n+1}} \psi\left(-\varphi \frac{\partial \varphi}{\partial x}\right) d\Omega^{n+1}$$

and

$$\int_{\Omega^{n+1}} \psi(\varphi v)\, d\Omega^{n+1} + \Delta t\theta \int_{\Omega^{n+1}} \psi(f\varphi u)\, d\Omega^{n+1}$$
$$= \int_{\Omega^n} \psi(\varphi v)\, d\Omega^n + \Delta t(1-\theta) \int_{\Omega^n} \psi\left(-\varphi \frac{\partial \varphi}{\partial y} - f\varphi u\right) d\Omega^n + \Delta t\theta \int_{\Omega^{n+1}} \psi\left(-\varphi \frac{\partial \varphi}{\partial y}\right) d\Omega^{n+1}$$

where they are coupled through the Coriolis terms. Since the mass equation can be solved for first, the gravity terms in the momentum equations ($\varphi(\partial\varphi/\partial x)$ and $\varphi(\partial\varphi/\partial y)$) at time t^{n+1} can be moved to the right-hand side as known quantities. These derivatives can then be obtained by differentiating the basis functions as is done in the typical finite element method. However, these same gravity terms at time t^n need to be determined in some other way because we need these values at the quadrature points. Section 4.5 describes how these derivatives are obtained. Approximating the conservation variables within each element Ω by the relation

$$f^{(e)} = \sum_{j=1}^{3} \psi_j f_j \tag{14}$$

gives the following matrix form for the mass:

$$\int_{\Omega^{n+1}} [\psi_i \psi_j]\, d\Omega^{n+1} [(\varphi)_j]^{n+1} = \int_{\Omega^n} \psi_i [(\varphi)]^n\, d\Omega^n$$

and for the two momentum equations

$$\int_{\Omega^{n+1}} \begin{bmatrix} \psi_i\psi_j & -\Delta t\theta\psi_i\psi_j\psi_k f_k \\ +\Delta t\theta\psi_i\psi_j\psi_k f_k & \psi_i\psi_j \end{bmatrix} \mathrm{d}\Omega^{n+1} \begin{bmatrix} (\varphi u)_j \\ (\varphi v)_j \end{bmatrix}^{n+1}$$

$$= \int_{\Omega^n} \begin{bmatrix} \psi_i(\varphi u) + \Delta t(1-\theta)\psi_i\left(-\varphi\dfrac{\partial\varphi}{\partial x} + f\varphi v\right) \\ \psi_i(\varphi v) + \Delta t(1-\theta)\psi_i\left(-\varphi\dfrac{\partial\varphi}{\partial y} - f\varphi u\right) \end{bmatrix}^n \mathrm{d}\Omega^n$$

$$+ \int_{\Omega^{n+1}} \begin{bmatrix} -\Delta t\theta\psi_i\psi_j\varphi_j\dfrac{\partial\psi_h}{\partial x}\varphi_k \\ -\Delta t\theta\psi_i\psi_j\varphi_j\dfrac{\partial\psi_h}{\partial y}\varphi_k \end{bmatrix}^{n+1} \mathrm{d}\Omega^{n+1} \tag{15}$$

which represents the element system. In Section 4.5, we describe how the integrals at time t^n are obtained.

4.4. Global equations

Adding all of the contributions from the elements to each of the global grid points results in the following linear system of global equations for the mass

$$A_{i,j}\varphi_j^{n+1} = b_i^{\varphi} \tag{16}$$

and the momentum equations

$$\begin{bmatrix} A_{i,j} & -B_{i,j} \\ B_{i,j} & A_{i,j} \end{bmatrix} \begin{bmatrix} (\varphi u)_j \\ (\varphi v)_j \end{bmatrix}^{n+1} = \begin{bmatrix} b_i^u \\ b_i^v \end{bmatrix}, \quad i, j = 1, \ldots, N \tag{17}$$

where

$$A_{i,j} = \int_{\Omega^{n+1}} \psi_i\psi_j \, \mathrm{d}\Omega^{n+1}$$

$$B_{i,j} = \Delta t\theta \int_{\Omega^{n+1}} \psi_i\psi_j\psi_k f_k \, \mathrm{d}\Omega^{n+1}$$

and b_i^{φ}, b_i^u, and b_i^v are the right-hand side vectors for the mass and u–v momentum respectively.

The matrices A and B are both symmetric positive definite and so the mass matrix is also symmetric positive definite but the momentum matrix is not. The mass matrix poses no difficulty and can be solved quite rapidly using conjugate gradient methods. However, the momentum matrix is skewed symmetric and for this type of matrix there is no one best solver. The momentum matrix is sparse, however, and iterative methods may be the best methods of solution. The good news, however, is that although we began with a system of non-linear equations, discretizing the equations by the weak Lagrange–Galerkin method results in a system of decoupled linear equations.

Because the momentum matrix is not symmetric positive definite we cannot safely utilize many of the best known solvers, such as the conjugate gradient method. There are, however, variants of these methods that can be used, such as the bi-conjugate gradient method. Because the system is quite sparse, direct methods are inefficient. For the moment, we are employing a direct lower–upper (LU) decomposition method, which, while not the most efficient, offers a simple way of inverting the coefficient matrices.

Another possibility for solving the momentum equations more efficiently is to simplify the equations in order to make them symmetric positive definite. This approach is introduced in Reference [23] and the strategy is to extrapolate the velocity field (using the times t^n and t^{n-1}) to get an approximation at t^{n+1}. This now allows moving the B matrices in Equation (17) to the right-hand side. Each conservation variable can now be solved explicitly. However, doing this greatly restricts the maximum time step that can be used. In our paper, we invert the skewed symmetric system with a direct solver but are currently exploring more efficient iterative methods, such as the multi-grid and bi-conjugate gradient methods.

4.5. Basic functions

The conservation variables belong to the following spaces:

$$\varphi \in H^1(\Omega), \qquad (\varphi u, \varphi v) \in H_0^1(\Omega)$$

and their test functions, likewise, are defined as

$$\psi_i^{\varphi} \in H^1(\Omega), \qquad \psi_i^{\varphi u, \varphi v} \in H_0^1(\Omega)$$

in other words, they belong to the set of square integrable functions whose first derivatives are also square integrable and are the linear triangular basis functions. The Sobolev space $H_0^1(\Omega)$ is defined as

$$H_0^1(\Omega) = \{\psi \in H_0^1(\Omega) | \psi(\Gamma) = 0\}$$

where Γ denotes the boundary of the domain Ω. These functions are written as follows:

$$\psi_i = \frac{a_i x + b_i y + c_i}{\det \triangle_{i,j,k}} \tag{18}$$

and have the exact integration rule

$$\int_{\Omega} \psi_i^{\alpha} \psi_j^{\beta} \psi_k^{\gamma} \, d\Omega = \det \triangle_{i,j,k} \frac{\alpha! \beta! \gamma!}{(\alpha + \beta + \gamma + 2)!} \tag{19}$$

where

$$\det \triangle_{i,j,k} = (\mathbf{x}_j - \mathbf{x}_i) \times (\mathbf{x}_k - \mathbf{x}_i)$$

and

$$a_i = y_j - y_k, \qquad b_i = x_k - x_j, \qquad c_i = x_j y_k - x_k y_j$$

The indices $i, j, k, = 1, \ldots, 3$ are cyclic in the following order:

$$\begin{tikzcd} & \boldsymbol{k} \arrow[dl] & \\ \boldsymbol{i} \arrow[rr] & & \boldsymbol{j} \arrow[ul] \end{tikzcd}$$

The C^0 basis functions given in Equation (18) are also called *natural* or *area* co-ordinates because they represent the area ratios between the triangle $\triangle_{i,j,k}$ and the sub-triangles formed by any point p in the interior of the element and the three vertices i, j, and k. For this reason, using these types of functions extend to higher dimensions because we can use similar triangles on the sphere as in References [21,25], or we can use the volume co-ordinates that are the linear tetrahedral functions in R^3.

4.5.1. Derivatives. The approximation of any function f within an element is given by Equation (14) and its derivative in the s-direction follows from Equation (14) and can be written as

$$\frac{\partial f^{(e)}}{\partial s} = \sum_{j=1}^{3} \frac{\partial \psi_j}{\partial s} f_j \tag{20}$$

where s can be either the x or y co-ordinate. Recall that in Equation (15) we need to know the values of the gravity terms $\varphi(\partial\varphi/\partial s)$ both at times t^{n+1} and t^n. Clearly, at time t^{n+1} we can use Equation (20) because these integrals occur in the Eulerian elements (see Figure 1). However, at time t^n, integrals now occur in the Lagrangian elements and if these elements span more than one Eulerian element, then using (20) would result in a constant derivative approximation across elements, which is clearly not consistent with C^0 basis functions. Therefore, the best way to obtain these values is the same way in which any other value at the quadrature points is obtained; i.e., by using the vertex (or node) values of the Eulerian elements to interpolate (using the local linear basis functions) the values at the quadrature points. For the conservation variables φ, φu, φv, these values are known at all the grid points; however, the derivatives are not. One way of obtaining these grid point derivatives is to consider the following: the derivative within an element can be written as done previously using the function values themselves at the grid points as given in Equation (20). However, if the derivatives at the grid points themselves were known, then we could also write the derivatives within an element as

$$\frac{\partial f^{(e)}}{\partial s} = \sum_{j=1}^{3} \psi_j \frac{\partial f_j}{\partial s} \tag{21}$$

and equating Equations (20) and (21) in a weak sense, yields

$$\int_\Omega \psi_i \psi_j \, d\Omega \frac{\partial f_j}{\partial s} = \int_\Omega \psi_i \frac{\partial \psi_j}{\partial s} f_j \, d\Omega \tag{22}$$

where the element contributions are summed in order to form a global system, which when inverted yields the derivatives at the grid points. This derivative matrix is symmetric positive definite and can be inverted easily; however, it can also be diagonalized for the sake of efficiency. This is the approach used in our paper. In Reference [25] this derivative computation approach was shown to be second-order accurate and better than a centered finite differencing approximation.

4.5.2. Integrals. At the Eulerian grid points (those at time t^{n+1}) the integration is done exactly using Equation (19) but it still remains to be decided how to handle the Lagrangian elements (those at time t^n). One way is to write the approximation within the Lagrangian element and equate the values using the Eulerian element, which it intersects. Let I be the region $L \cap E$. Thus, we can write any grid point shared by both elements as

$$\mathbf{x} = \sum_{j=1}^{3} \psi_j^E \mathbf{x}_j^E = \sum_{j=1}^{3} \psi_j^I \mathbf{x}_j^I$$

and expanding these relations results in the following matrix system:

$$\begin{bmatrix} x_1^I & x_2^I & x_3^I \\ y_1^I & y_2^I & y_3^I \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} \psi_1^I \\ \psi_2^I \\ \psi_3^I \end{bmatrix} = \begin{bmatrix} x_1^E\psi_1^E + x_2^E\psi_2^E + x_3^E\psi_3^E \\ y_1^E\psi_1^E + y_2^E\psi_2^E + y_3^E\psi_3^E \\ 1 \end{bmatrix}$$

which after inverting yields

$$\psi_i^I = \frac{a_i^I\psi_1^E + b_i^I\psi_2^E + c_i^I\psi_3^E + d_i^I}{\det \triangle_{i,j,k}^I} \tag{23}$$

where

$$a_i^I = x_1^E\eta_i^I + y_1^E\xi_i^I, \qquad b_i^I = x_2^E\eta_i^I + y_2^E\xi_i^I, \qquad c_i^I = x_3^E\eta_i^I + y_3^E\xi_i^I, \qquad d_i^I = x_j^I y_k^I - x_k^I y_j^I$$

$$\xi_i^I = x_k^I - x_j^I, \qquad \eta_i^I = y_j^I - y_k^I$$

and

$$\det \triangle_{i,j,k}^I = (\mathbf{x}_j^I - \mathbf{x}_i^I) \times (\mathbf{x}_k^I - \mathbf{x}_i^I)$$

In Figure 1 we see that the Lagrangian element $L_{1,2,3}$ spans six Eulerian elements. Note that element 1 and the Lagrangian element overlap in only one triangular intersection region (dark triangle), whereas element 2 overlaps with the Lagrangian element in two triangular intersection regions (light triangles). The basis functions of these intersection regions are written as functions of the surrounding Eulerian basis functions. Since we have exact integration rules for any integral involving the Eulerian basis functions, we now also have exact integration rules for the intersection regions. This exact integration approach was introduced in Reference [9] and is quite an elegant method. Unfortunately, in order to integrate the elements exactly we must first decompose the Lagrangian element such that the pieces lie exactly within Eulerian elements. This requires an unstructured mesh generation strategy, which essentially becomes a very tedious exercise in intersection and inclusion tests. Therefore, in the absence of efficient intersection and inclusion tests it is best to use some other simpler means.

Another approach for obtaining the integrals at time t^n is simply to interpolate the vertex of the Lagrangian element (L_1, L_2, L_3) and then construct linear basis functions within this element. The obvious shortcoming of this approach is that we now lose the variation within the element by assuming a linear approximation. Using this simpler approach would render the method quite inutile. Therefore, the final possibility is to take sufficient sampling (or quadrature) points in order to capture the variation within the element. In other words, we seek the minimum number of quadrature points that can extract the maximum amount of information from all six surrounding elements.

Many quadrature rules have been tested and the quintic (seven-point) Gauss quadrature rule appears to be the minimum that should be used in order to get results comparable with the exact integration approach. In the numerical integration approach, the vertices of the Eulerian triangular element (t^{n+1}) are mapped along the characteristics to where they resided at the previous time step (t^n). The feet of these characteristics now forms the Lagrangian element $L_{1,2,3}$. Within this Lagrangian element the variables at the quadrature points are interpolated linearly using the basis functions (area co-ordinates) of the Eulerian element, which claims the quadrature points. In other words, the quadrature value arising from the Eulerian element 1 (Figure 1) is obtained by linearly interpolating the vertex values of element 1 at the quadrature point. Similarly, the quadrature point lying in element 2 gets its value from the interpolation of the values from element 2, and so on. In this fashion, the integration takes into account the variation (higher than linear) of the function values inside the Lagrangian element. For this reason, using a low-order quadrature rule will result in large inaccuracies while higher orders will always result in increased accuracy (until a limit is reached). For the problems studied in this paper, the minimum number of sampling points appeared to be around seven, which results in our choice of a quintic quadrature rule.

Using numerical integration results in almost identical results to those of exact integration. The only potential pitfalls are in conservation and stability. The exact integration approach is exactly conservative provided that a direct solver is used to invert the matrix. This is rarely feasible, and when iterative methods are used exact conservation can no longer be guaranteed. In practice, however, the lack of exact conservation is insignificant and has not been shown to adversely affect the results. For the problems studied thus far, conservation has been preserved even for the numerical integration approach with an iterative solver.

The question of stability is a bit more elusive to pin down. Priestley [9] showed that using quadrature rules for the numerical integration of the Lagrangian elements results in instabilities, in theory. However, in practice these instabilities have yet to present themselves either in Priestley's [9] or our current work. The exact integration approach does not suffer the likelihood of such instabilities, however. A comparison of the exact and numerical integration approaches are discussed in Section 6.

4.6. Searching algorithms

The crux of any Lagrangian scheme is the accurate calculation of the particle trajectories, in other words the correct solution of Equation (7). Once the correct trajectories are computed it then remains to interpolate either the quadrature points in the numerical integration approach or the points comprising the intersection regions in the exact integration approach. For structured or quasi-structured grids *ad hoc* searching algorithms can be constructed but for unstructured or general grids quadtree algorithms are the best choice.

The idea behind the quadtree algorithm is to find the nearest neighbor in the background (Eulerian) grid to the departure point being searched. Let *quad_tree*[1: *ntree*, 1: 7] be an integer array that stores this quadtree. This array stores the following information:

- *quad_tree*[*i*, 1–4] stores the four children of this quad.
- *quad_tree*[*i*, 5] stores the position of this quad with respect of its parent.
- *quad_tree*[*i*, 6] stores the location of its parent.
- *quad_tree*[*i*, 7] stores the number of nodes contained within this quad.

However, another integer array is required, which stores the list of elements claiming each grid point. Upon finding this nearest neighbor we then search through the list of elements that claim this grid point and check for inclusion using the local area co-ordinates (basis functions) of the element. If we let the departure point be denoted by p, then if the following conditions are met:

$$\det \triangle_{p,j,k} > 0, \qquad \det \triangle_{i,p,k} > 0, \quad \text{and} \quad \det \triangle_{i,j,p} > 0$$

then the point p is said to be contained within the element with vertices i, j, k.

There are usually no more than six elements claiming each node even for distorted unstructured grids. For highly distorted grids, however, the departure point may not necessarily lie within one of the elements claiming the nearest neighbor. A good example where this can occur is in non-convex domains. In this case, during the sweep through the element list claiming the nearest neighbor, the minimum distance between the departure point and the element nodes is stored. The element node yielding the minimum distance is considered to be the new nearest neighbor. If no inclusion is found, then the new nearest neighbor is used and the process is repeated. Therefore, in the worst case scenario, only two nearest neighbor loops are required. This can have adverse affects on the efficiency of the scheme if this case should arise often; fortunately, this situation has not yet presented itself.

5. ADAPTIVE UNSTRUCTURED MESH GENERATOR

A practical condition for creating optimal grids based on numerical solutions is to require that the error be equally distributed across all elements Ω comprising the domain D. Let

$$\varphi_I(x, y) = \sum_{m=1}^{N} \psi_m(x, y)\varphi(x_m, y_m)$$

be the interpolation of $\varphi(x, y)$ in Ω, where N is the number of interpolation points. The error in using the interpolant can be written via a Taylor series expansion about the point (a, b) as

$$\begin{aligned}\varphi(x, y) - \varphi_I(x, y) = &\sum_{i+j=k+1} \varphi^{(i,j)}(x, y)\frac{(x-a)^i}{i!}\frac{(y-b)^j}{j!}\\ &- \sum_{m=1}^{N}\sum_{i+j=k+1} \varphi^{(i,j)}(x, y)\frac{(x-a)^i}{i!}\frac{(y-b)^j}{j!}\psi_m(x, y)\end{aligned}$$

where $\varphi^{(i,j)}$ denotes the derivatives of φ with respect to x and y. Taking the ∞ norm and using norm properties to simplify gives

$$\left\|\varphi(x, y) - \varphi_I(x, y)\right\|_\infty \leq C \max h^{k+1} \sum_{i+j=k+1} \left|\varphi^{(i,j)}(x, y)\right| \tag{24}$$

In this relation k is the order of the interpolation functions, h is the local grid spacing, and C is a constant. A value of $k = 1$ corresponds to linear triangles. Therefore, for an optimal grid using linear triangular finite elements we require that

$$C \max h^2 \sum_{i+j=2} \left|\varphi^{(i,j)}(x, y)\right| = \text{costant}, \quad \forall\Omega\in D$$

The choice of error indicator and indicator variable are very much problem-dependent. The error indicator used in this paper uses the derivative of the mass variable. The strategy behind this mesh generator can be characterized as a global reconstruction (or remeshing), as opposed to a refinement method that only changes the grid locally. It is an advancing front method and is termed a global reconstruction because whenever the grid must be altered it is constructed beginning with the boundaries and circling towards the center of the domain (see Reference [26] for further details on this mesh generator).

The accurate interpolation of values from one grid to a newly refined grid is not at all simple since both grids (the old and newly adapted meshes) may be completely unstructured and the domain is not necessarily convex. Therefore, local methods must be used whereas global constructs like splines cannot.

For steady state problems, the grid points on the new mesh can be interpolated using the basis functions and elements from the old mesh. This is possible because we only require an initial guess for the problem and while the initial guess can affect the convergence of the solution, it does not affect the accuracy. In transient problems, on the other hand, the new grid point values should not only be as accurate as possible but the variables should also be conserved. By using the idea of the weak Lagrange–Galerkin method, we can obtain these conserving integrals, which yield the values at the new grid points [9]. In other words, we solve the following system:

$$\int_{\Omega_{\text{new}}} \psi_i \psi_i \, d\Omega_{\text{new}} \varphi_j^{\text{new}} = \int_{\Omega_{\text{old}}} \psi_i \varphi^{\text{old}} \, d\Omega_{\text{old}}$$

where the right-hand side is known. This approach globally conserves the variable anywhere from one to three orders of magnitude better than a linear interpolator using the local basis functions. In fact, the change in the conservation variables from an old mesh projected onto a newly adapted mesh is around the order of 1×10^{-8} percent, which is obviously conservative to within machine precision.

6. NUMERICAL EXPERIMENTS

For the numerical experiments, the following terms are used to compare the performance of the schemes: the L_2 error norm

$$\|f\|_{L_2} = \frac{\int_{\Omega} (f_{\text{exact}} - f)^2 \, d\Omega}{\int_{\Omega} f_{\text{exact}}^2 \, d\Omega}$$

where f represents any of the conservation variables in Equations (3), the trajectory norm [25]

$$T = \frac{\int_{\Omega} (\mathbf{x}_{\text{D}} - \mathbf{x}_{\text{D}}^{\text{exact}})^2 \, d\Omega}{\int_{\Omega} (\mathbf{x}_{\text{A}} - \mathbf{x}_{\text{D}}^{\text{exact}})^2 \, d\Omega}$$

where subscripts A and D denote the arrival and departure points, and the following two additional measures

$$M = \frac{\int_{\Omega} \varphi \, d\Omega}{\int_{\Omega} \varphi_{\text{exact}} \, d\Omega}, \qquad E = \frac{\int_{\Omega} \varphi(u^2 + v^2) + \varphi^2 \, d\Omega}{\int_{\Omega} \varphi_{\text{exact}}(u_{\text{exact}}^2 + v_{\text{exact}}^2) + \varphi_{\text{exact}}^2 \, d\Omega}$$

The L_2 error norm compares the root-mean-square percent error of the numerical and exact solutions, T measures the accuracy of the trajectories, M measures the conservation property of the mass, and E measures the conservation of the total available energy. The ideal scheme should yield L_2 error and trajectory norms of zero, and mass and energy measures of one. In addition, the following ratio is used throughout the following sections:

$$\sigma = \frac{\Delta t}{\Delta t_{\mathrm{RK}}}$$

The variable represents the ratio between the Lagrange–Galerkin time step and the maximum allowable time step for the Eulerian fourth-order Runge–Kutta presented in Section 3. Since the fourth-order Runge–Kutta method allows times steps up to Courant numbers of 2 this ratio represents a very stringent measure of the Lagrange–Galerkin time step. The Courant number on unstructured triangular grids is best obtained in an edgewise manner; in other words, the characteristic length used is the sides of the triangle and the velocity is the velocity vector at the mid-point of the particular side, thereby yielding

$$C = \Delta t \sqrt{\frac{u^2 + v^2}{\Delta x^2 + \Delta y^2}}$$

It should also be noted that the resolution of the grid is given in terms of the number of quadrilaterals in the grid. For instance, a 40×40 grid ($N = 40$) actually contains 80×80 triangular elements. But we denote the resolution of the grid by the number of quadrilaterals as these yield the correct element spacing. For example, for a domain having a length in the x-direction of 2 (as in cases 1 and 2), the $N = 40$ grid yields an element spacing of $\Delta x = 2/40$.

6.1. Problem statement

In the following sections, the three test cases used to measure the performance of the Lagrange–Galerkin method are introduced. They are a solid body rotation with time-independent velocity field, solid body rotation with time-dependent velocity field and a westward traveling soliton wave.

6.1.1. Case 1: solid body rotation. The initial condition for this test case is given by the Gaussian wave

$$\varphi(x, y, 0) = \exp\left\{\frac{-[(x - x_0)^2 + (y - y_0)^2]}{2\lambda_0^2}\right\}$$

having the far boundary conditions

$$\varphi(x, y, t) = 0, \quad \forall (x, y) \in \Gamma$$

where

$$\lambda_0 = \frac{1}{8}, \qquad (x_0, y_0) = \left(-\frac{1}{2}, 0\right), \qquad (x, y) \in [-1, 1]$$

The velocity field is constant for all times and is given by

$$u = +y \quad \text{and} \quad v = -x$$

which defines a clockwise rotation about the center of the domain. There are no Coriolis effects and since only the mass is allowed to vary, this problem simplifies to the passive advection of the quantity φ. The Gaussian wave rotates along this circular path with neither distortion nor dissipation. The exact solution is given by

$$\varphi(x, y, t) = \exp\left\{\frac{-[(\tilde{x})^2 + (\tilde{y})^2]}{2\lambda_0^2}\right\}$$

where

$$\tilde{x} = x - x_0 \cos t - y_0 \sin t \quad \text{and} \quad \tilde{y} = y + x_0 \sin t - y_0 \cos t$$

Results are given for one full revolution of the initial wave. The period for one revolution of the wave is 2π, which means that one revolution corresponds to $t = 2\pi$. For a structured 40×40 element grid ($N = 40$) $\Delta t_{\mathrm{RK}} = 2\pi/80$, which is the value used to determine the time step ratio σ for the Lagrange–Galerkin method.

6.1.2. Case 2: solid body rotation with time-dependent velocity. This test case was introduced by Chukapalli [23]. The only difference between this test case and the previous one is that the velocity field is now given as

$$u = +y(1 + \cos t) \quad \text{and} \quad v = -x(1 + \cos t) \tag{25}$$

Therefore, the problem has the same period of revolution as in case 1 and it admits the same type of solution except that now the velocity field is a function of time. Note that this velocity field is uniformly rotational about the center of the domain. Therefore, any fluid particle starting out at a specific radial distance from the center of the domain will always remain on the circle defined by the initial radius. For this reason it is best to obtain the analytic trajectories by mapping from Cartesian to polar co-ordinates using

$$x = r \cos \theta \quad \text{and} \quad y = r \sin \theta \tag{26}$$

Using the u component we note that

$$\frac{\mathrm{d}x}{\mathrm{d}t} \equiv u = +y(1 + \cos t) \tag{27}$$

and substituting (26) into the left-hand side of (27) yields

$$\frac{\mathrm{d}}{\mathrm{d}t}(r\cos\theta)=\cos\theta\frac{\mathrm{d}r}{\mathrm{d}t}-r\sin\theta\frac{\mathrm{d}\theta}{\mathrm{d}t}$$

But note that from

$$r=\sqrt{x^2+y^2}$$

we can show that

$$\frac{\mathrm{d}r}{\mathrm{d}t}=\frac{x}{r}\frac{\mathrm{d}x}{\mathrm{d}t}+\frac{y}{r}\frac{\mathrm{d}y}{\mathrm{d}t}$$

Substituting the u and v components of velocity from Equation (25) yields

$$\frac{\mathrm{d}x}{\mathrm{d}t}=\frac{x}{r}y(1+\cos t)-\frac{y}{r}x(1+\cos t)=0$$

Therefore, Equation (27) becomes

$$-r\sin\theta\frac{\mathrm{d}\theta}{\mathrm{d}t}=+y(1+\cos t)$$

and canceling like terms yields

$$\frac{\mathrm{d}\theta}{\mathrm{d}t}=-(1+\cos t) \tag{28}$$

which describes the trajectories in terms of the angular component as a function of time (we would have obtained the same expression if we had used the v component instead). Since we know that the exact solution is of the form

$$\varphi(x,y,t)=\exp\left\{\frac{-[(\tilde{x}-x_0)^2+(\tilde{y}-y_0)^2]}{2\lambda_0^2}\right\}$$

we now also know the exact solution in terms of θ because the Cartesian co-ordinates are given in terms of the angular co-ordinates as

$$\tilde{x}=r\cos\theta(t)\quad\text{and}\quad\tilde{y}=y\sin\theta(t)$$

Integrating Equation (28) we get

$$\theta(t)=\theta_0-(t+\sin t)$$

where

$$\theta_0 = \arctan \frac{y_0}{x_0}$$

This test problem also has a period of $t = 2\pi$ but the velocity field initially is very large and slows down continuously until finally reaching zero velocity at one half revolution ($t = \pi$). The velocity field then begins to increase at this point and by virtue of the behavior of the cosine function, the velocity field is symmetric with respect to the time $t = \pi$. For a structured 40×40 element grid ($N = 40$), $\Delta t_{\mathrm{RK}} = 2\pi/160$ for this case. This maximum time step is smaller than in case 1 owing to the fact that the velocity field is twice that of case 1 near $t = 0$ and $t = 2\pi$, and therefore a time step half as large is required for stability.

6.1.3. Case 3: Rossby soliton waves. This problem describes an equatorially trapped Rossby soliton wave [27]. The soliton wave starts off in the center of the domain. I then moves westward along the equator without changing shape. The exact solution is given by

$$\varphi(x, y, t) = \varphi^{(0)} + \varphi^{(1)}$$

$$u(x, y, t) = u^{(0)} + u^{(1)}$$

$$v(x, y, t) = v^{(0)} + v^{(1)}$$

where the superscripts (0) and (1) denote the zeroth- and first-order asymptotic solutions of the shallow water equations respectively. They are given by

$$\varphi^{(0)} = \eta\left(\frac{-9 + 6y^2}{4}\right) \mathrm{e}^{-y^2/2}$$

$$u^{(0)} = \frac{\partial\eta}{\partial\xi}(2y)\, \mathrm{e}^{-y^2/2}$$

$$v^{(0)} = \eta\left(\frac{3 + 6y^2}{4}\right) \mathrm{e}^{-y^2/2}$$

$$\varphi^{(1)} = c^{(1)}\eta \frac{9}{16}(-5 + 2y^2)\, \mathrm{e}^{-y^2/2} + \eta^2\Phi^{(1)}(y)$$

$$u^{(1)} = c^{(1)}\eta \frac{9}{16}(3 + 2y^2)\, \mathrm{e}^{-y^2/2} + \eta^2 U^{(1)}(y)$$

$$v^{(1)} = \frac{\partial\eta}{\partial\xi}\eta V^{(1)}(y)$$

where $\eta(\xi, t) = A\ \mathrm{sech}^2 B\xi$, $\xi = x - ct$, $A = 0.771B^2$, $B = 0.394$, and $c = c^{(0)} + c^{(1)}$, where $c^{(0)} = -\frac{1}{3}$ and $c^{(1)} = -0.395B^2$. The variable η is the solution to the equation

$$\frac{\partial \eta}{\partial \tau} + \alpha_n \eta \frac{\partial \eta}{\partial \xi} + \beta_n \frac{\partial^3 \eta}{\partial \xi^3} = 0$$

which is the famous Korteweg–de Vries equation, which yields soliton wave solutions. The shallow water equations can be simplified into this equation using the method of multiple scales presented in Reference [28]. Finally, the remaining terms are given by

$$\begin{bmatrix} \Phi^{(1)}(y) \\ U^{(1)}(y) \\ V^{(1)}(y) \end{bmatrix} = \mathrm{e}^{-y^2/2} \sum_{n=0}^{\infty} \begin{bmatrix} \varphi_n \\ u_n \\ v_n \end{bmatrix} H_n(y)$$

where $H_n(y)$ are the Hermite polynomials and φ_n, u_n, v_n are the Hermite series coefficients given in Reference [27]. The boundary conditions used are

$$\mathbf{n} \cdot \mathbf{u} = 0, \quad \forall (x, y) \in \Gamma$$

where $\mathbf{n}$ is the outward pointing normal vector and the Coriolis parameter is given by

$$f(y) = y$$

where

$$x \in [-24, +24] \quad \text{and} \quad y \in [-8, +8]$$

(see References [1,2] for further details of this test case). The equations are integrated up to a non-dimensional time of $t = 10$. For a structured 64×32 element grid $\Delta t_{\mathrm{RK}} = \frac{1}{16}$.

6.2. Results

6.2.1. Structured grids

6.2.1.1. Case 1. This case was run using structured grids as shown in Figure 2. This plot shows the grid and contours of the mass for a 40×40 element grid ($N = 40$). Figure 3 shows the L_2 norm as a function of the number of elements N in each direction. All results are given using 80 time steps ($\Delta t = 2\pi/80$) to complete one full revolution. Therefore, for the lowest resolution grid ($N = 20$) σ is around $\frac{1}{4}$, while for the highest resolution grid ($N = 100$) σ is close to 3. Figure 3 shows that the error decreases quadratically with N.

Figure 4 shows the effects of σ on the solution of an $N = 40$ grid for the following three trajectory calculation methods: Runge–Kutta (RK), composite mid-point rule (CM), and exact trajectories (EX). Specifically, Figure 4(a) shows the effects on the L_2 norm of the mass while Figure 4(b) shows the trajectory norm. From Figure 4(a) we see that the RK method is more accurate than the CM method. Clearly, using exact trajectories yields far more accurate results but they are hardly ever known *a priori* except in transport models where the velocity fields are known at all times. Figure 4(b) shows that the trajectory error increases linearly with

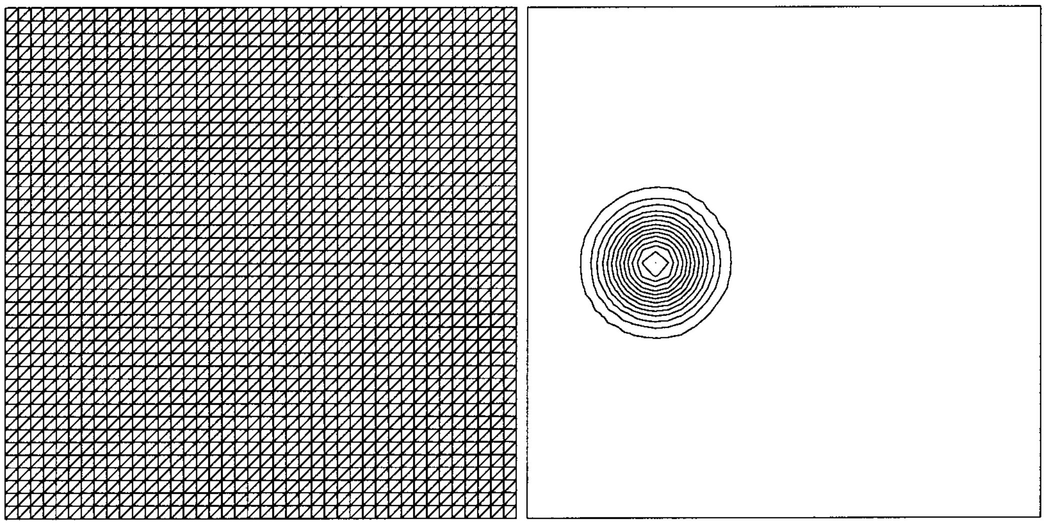


Figure 2. The $N = 40$ structured grid and mass contours for cases 1 and 2 at $t = 2\pi$.

the time step. However, Figure 4(a) shows that there is a trade-off between time step size and the number of interpolations. In other words, for small time steps very accurate trajectories are obtained but many interpolations are required, while large time steps yield less accurate trajectories but require fewer interpolations.

This test case was also used to compare the piecewise exact and numerical integration approaches for obtaining the integrals at the feet of the characteristics (the integrals involving the Lagrangian elements, i.e., those at time t^n). Numerical experiments were performed on the $N = 40$ grid and the exact and numerical approaches yielded error norms within 0.1 percent of each other with similar conservation measures M and E; however, the exact integration approach took twice as long to complete one full revolution. The inefficiency of the algorithm is in the decomposition of the Lagrangian element into the intersection regions between the Lagrangian and Eulerian elements, which are the regions having the basis functions given in Equation (23). An advancing front mesh generator and a point insertion Delaunay triangulation were both compared and although the Delaunay method was much faster it still could not compete with the numerical integration approach. On these grounds, the numerical integration approach is used exclusively for the remainder of the paper.

6.2.1.2. Case 2. This test case is only used to test the accuracy of the trajectory calculation methods. In case 1 we were able to test the trajectory calculations for a steady velocity field. Case 2 is used to test the trajectory calculations for a transient velocity field. A comparison of the CM and RK schemes is shown in Figure 5 for a structured $N = 40$ grid. Figure 5(a) shows the effects of varying the time step ratio σ on the L_2 norm of the mass while Figure 5(b) shows the effects on the trajectory norm. It is evident from this figure that the RK method is superior to the CM method but now only slightly so. It is quite surprising that the difference between

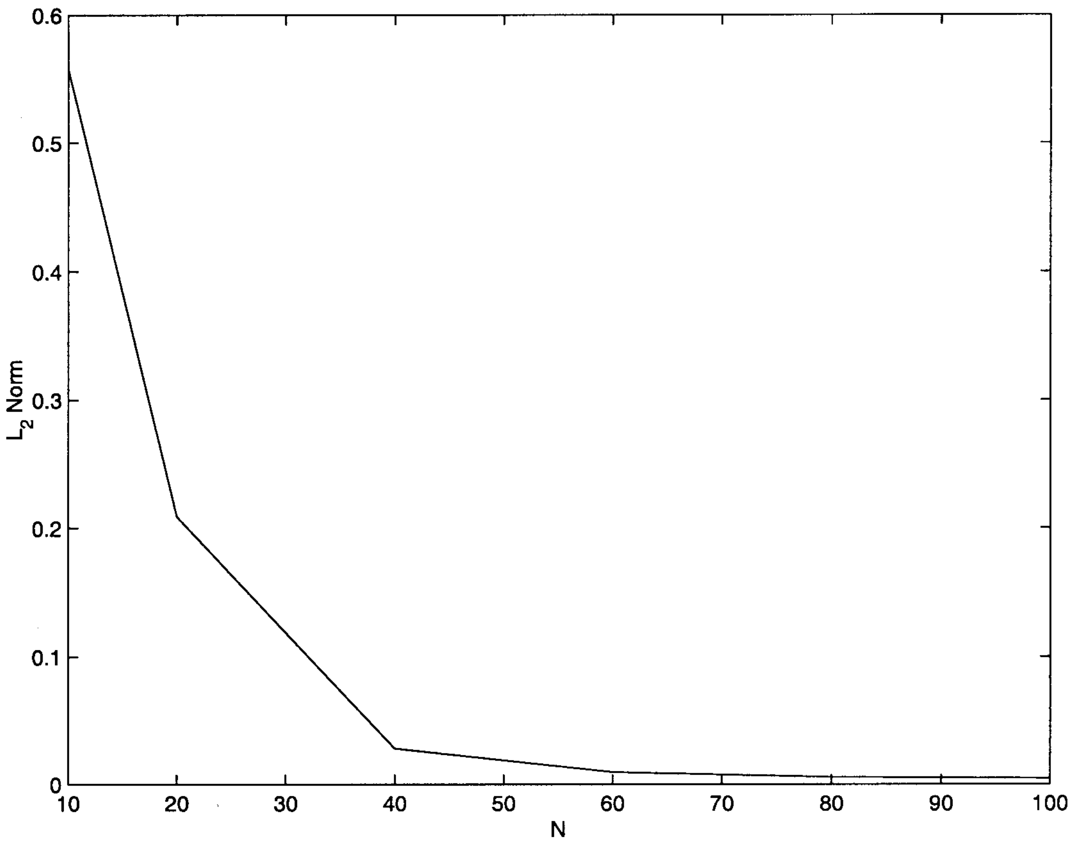
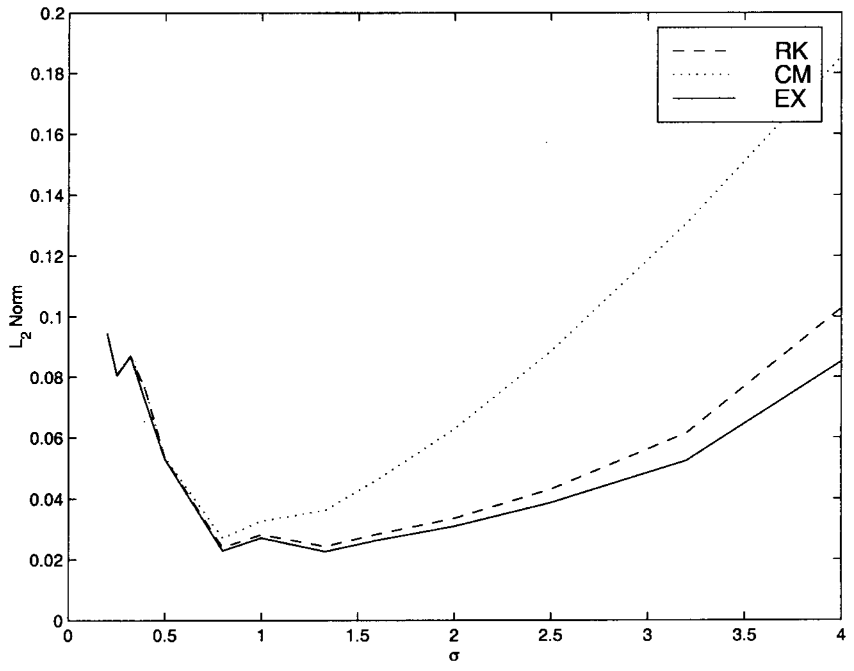


Figure 3. The mass L_2 norm as a function of the number of structured elements N for case 1 at $t = 2\pi$.

the RK and CM methods is so slight especially since Figure 5(b) shows that the RK method yields far more accurate trajectory norms than the CM method. Therefore, for such a strongly varying velocity field as used in this test case, most of the error to the mass comes from the extrapolation of the velocity field and not from the trajectory calculation. Figure 5(a) shows that if the exact trajectories, or values close to them, are used then a very good error norm for the mass can be obtained. Clearly, there is room for improvement and although we may not expect to obtain the exact trajectories, we should be able to come quite close.

6.2.1.3. Case 3. Figure 6 shows the grid and contours for a structured 64×32 ($N = 64$) element grid using 100 iterations to get to $t = 10$. Figure 6 shows the (a) grid, (b) mass, (c) u momentum, and (d) v momentum. Since the flow is unidirectional along the x co-ordinate, the v momentum is actually quite small and very close to zero. Qualitatively, we would like the solution to be symmetric with respect to the y-axis. Figure 7 shows the L_2 norm for the three primitive variables as a function of N. For this test case, N refers to the number of elements in the x-direction. The number of elements in the y-direction is $N/2$. Therefore, for an $N = 80$

a)



b)

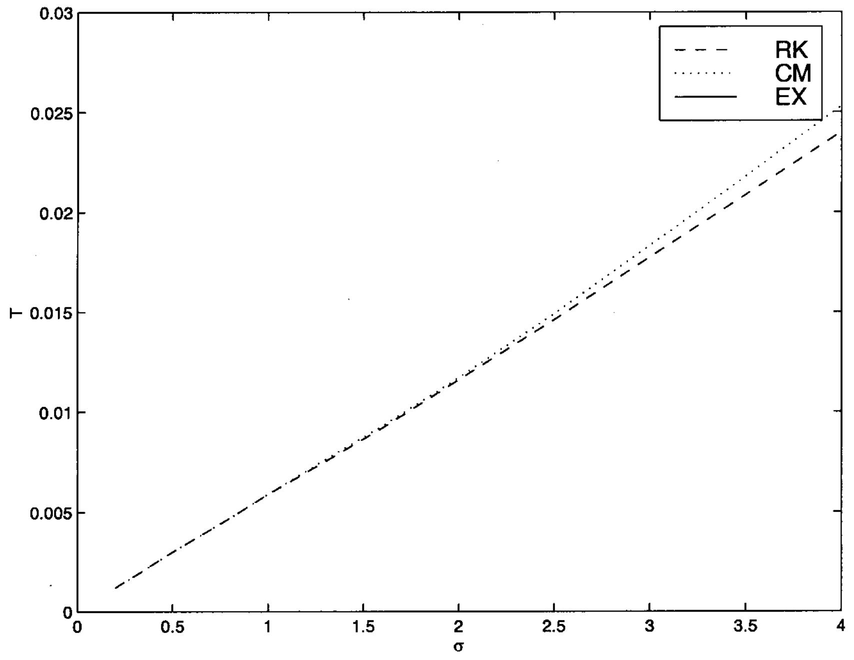
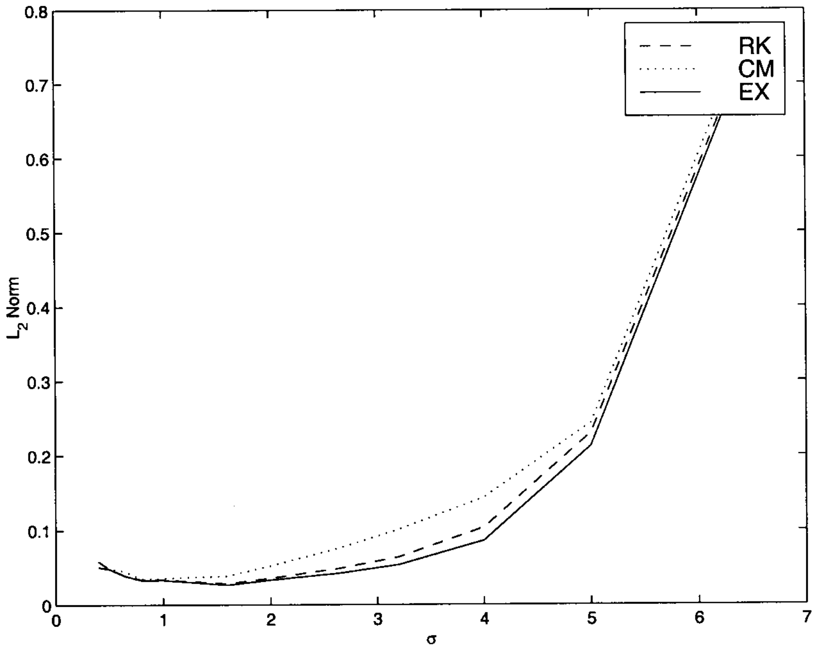


Figure 4. The effect of the time step ratio σ on the (a) mass L_2 norm and (b) trajectory norm T for the structured $N = 40$ grid for case 1 at $t = 2\pi$ using trajectories obtained by: the RK method, the CM rule, and EX.



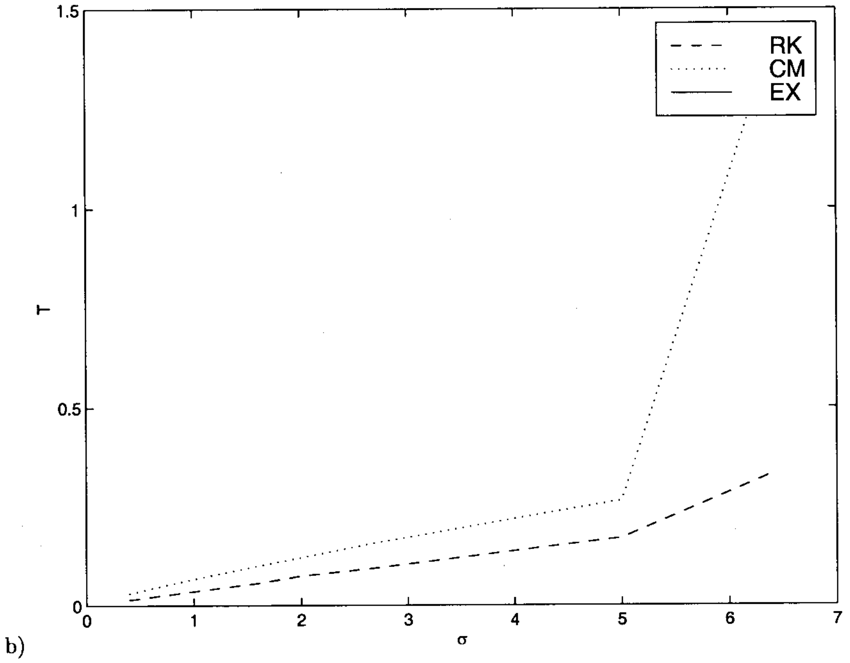


Figure 5. The effect of the time step ratio σ on the (a) mass L_2 norm and (b) trajectory norm T for the structured $N = 40$ grid for case 2 at $t = 2\pi$ using trajectories obtained by: the RK method, the CM rule, and EX.

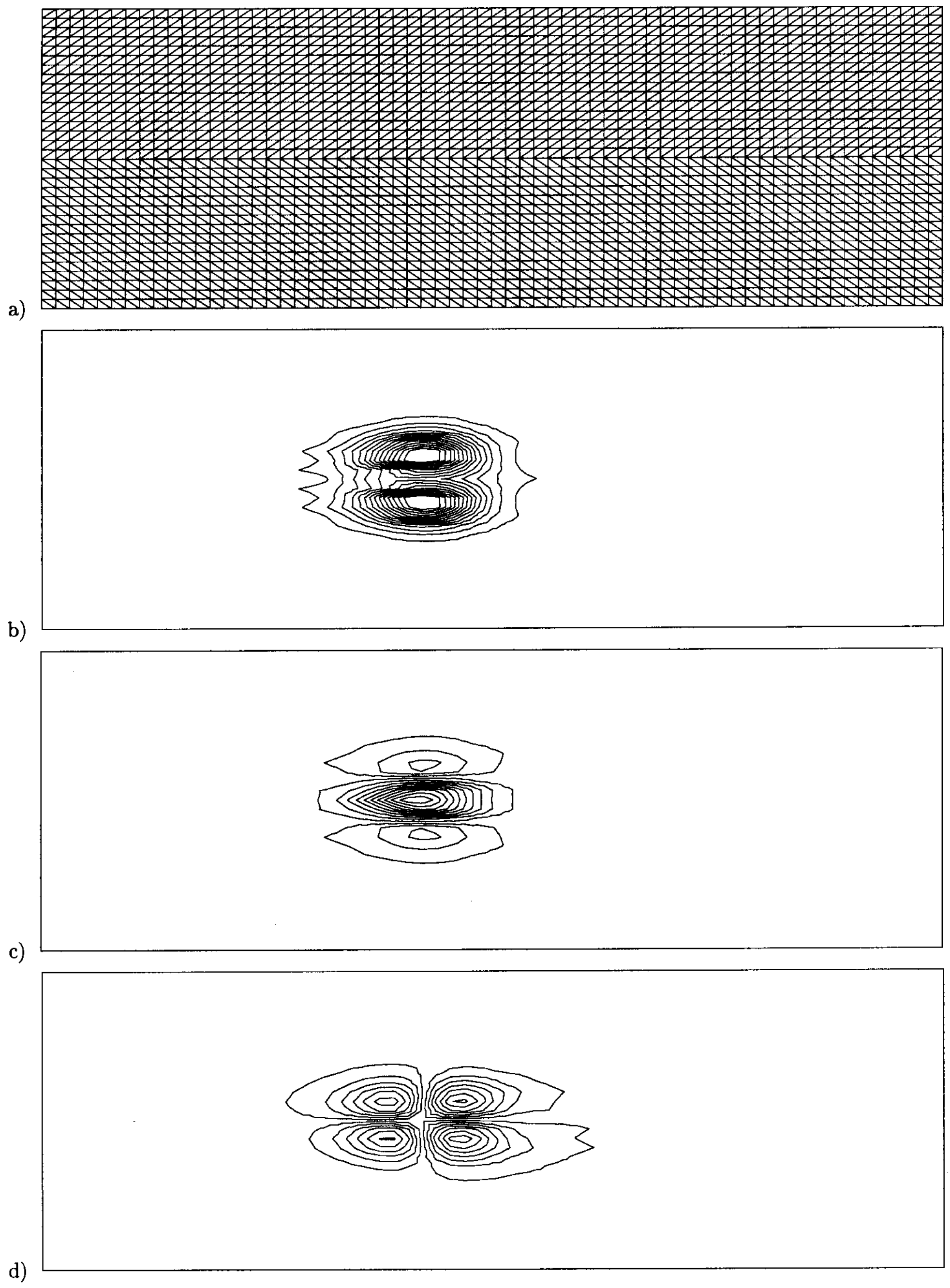


Figure 6. The structured $N = 64$ (a) grid and contours for the (b) mass, (c) u momentum, and (d) v momentum for case 3 at $t = 10$.

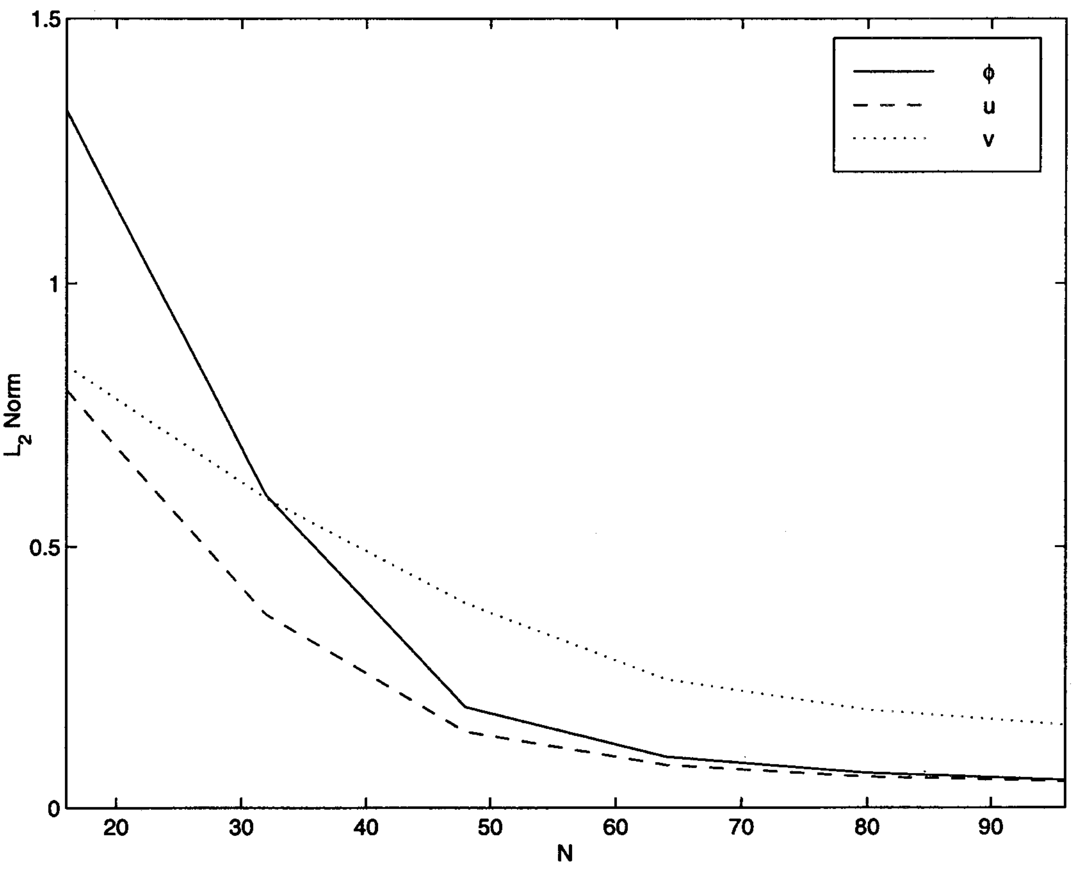


Figure 7. The L_2 norms as a function of the number of structured elements N for case 3 at $t = 10$.

grid, $N_x = 80$ and $N_y = 40$, and so on. In Figure 7, the mass and u momentum are converging at a quadratic rate, while the v momentum is converging at a slightly lower rate.

Figure 8 shows the effects of the time step ratio σ on the L_2 norm using an $N = 64$ grid. The error norms remain the same for almost all values of σ, except around 5, where the solution accuracy begins to diminish. Therefore, as was seen in cases 1 and 2, the Lagrange–Galerkin method has a clear optimal time step. It would be very useful to be able to determine what this optimal time step should be; however, there is definitely a maximum time step that should not be exceeded. Increasing the time step beyond this maximum allowable time step greatly diminishes the numerical accuracy of the solution.

6.2.2. Unstructured grids

6.2.2.1. Case 1. This case was run using unstructured grids like those shown in Figure 9. This figure illustrates the grid and contours of the mass for a comparable grid to the structured

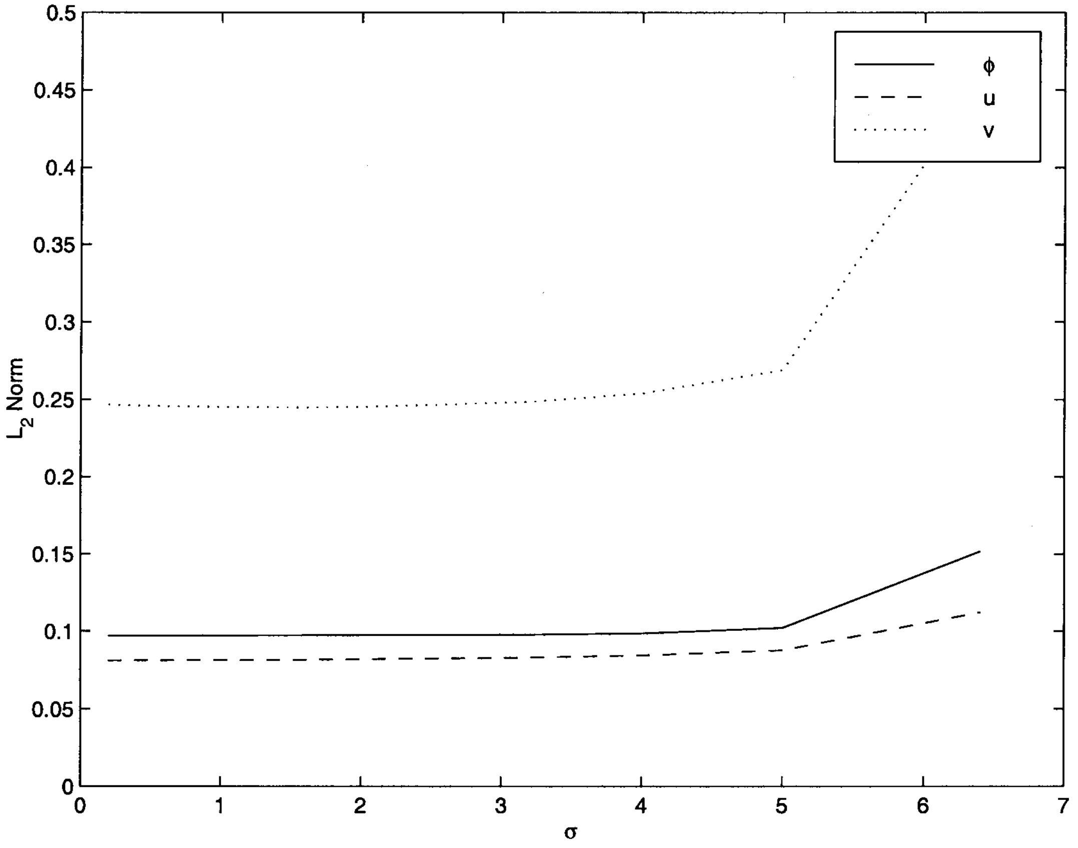


Figure 8. The effect of the step ratio σ on the L_2 norms for the structured $N = 64$ grid for case 3 at $t = 10$.

$N = 40$ grid. Figure 10 shows the L_2 norm as a function of N. All results are obtained using 80 time steps to complete one full revolution. Therefore, for the lowest resolution grid ($N = 10$) σ is also around $\frac{1}{4}$, and for the highest resolution grid ($N = 100$) σ is around 3. Figure 10 shows that the error decreases quadratically with N even for these unstructured grids. In fact, by comparing Figures 3 and 10 we can see that both the structured and unstructured grids yield about the same order of accuracy. This case shows that the weak Lagrange–Galerkin method works equally well on both structured and unstructured grids.

6.2.2.2. Case 3. The results presented for this case are obtained using 100 iterations to reach the non-dimensional time of $t = 10$; however, the grids are now unstructured. These grids are not randomly unstructured but rather the very flexible mesh generation scheme presented here is used to construct *a priori* grids. *A priori* grids refer to the construction of a grid by anticipating the behavior of the solution. In this case, since the soliton wave moves along a

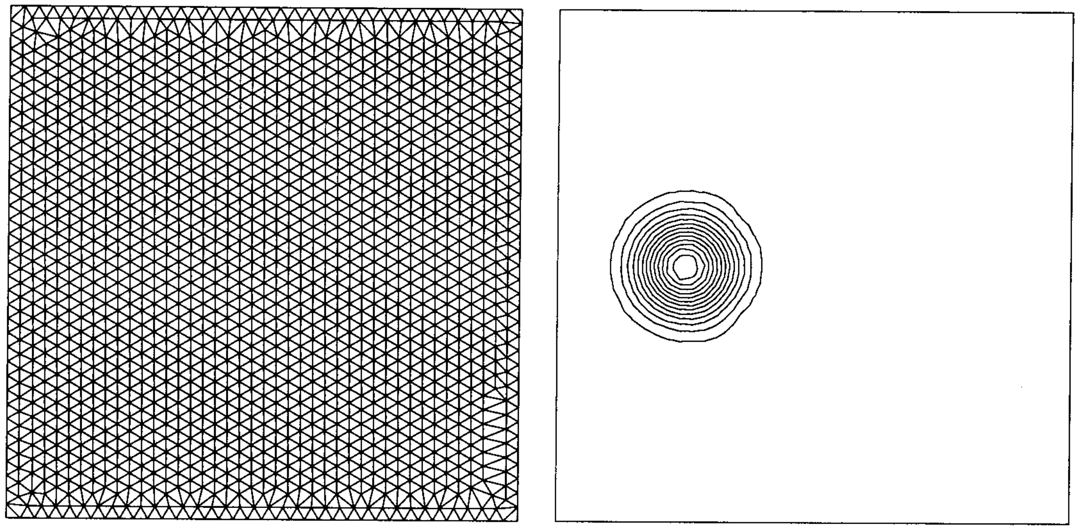


Figure 9. The $N = 40$ unstructured grid and mass contours for case 1 at $t = 2\pi$.

very constrained region along the x-direction, it makes sense to introduce many elements where the wave is destined to pass and few elements where the wave will not. Figure 11 shows a sample of an *a priori* grid comparable with the structured $N = 64$ grid. Figure 11 shows the (a) grid, (b) mass, (c) u momentum, and (d) v momentum. Figure 12 shows the L_2 norm as a function of N. The plot shows a fast rate of convergence especially for the mass (φ) and u momentum (u). This test case shows qualitatively and quantitatively that the weak Lagrange–Galerkin method yields very good solutions to the full non-linear shallow water equations on unstructured grids. In fact, the unstructured grids achieved the same order of accuracy as the structured grids as is evidenced by comparing Figures 7 and 12.

6.2.3. *Adaptive grids*

6.2.3.1. Case 1. As a final test, this case was run using an adaptive unstructured mesh generation strategy. The initial grid is comparable with the structured $N = 40$ grid and is allowed to adapt only after each quarter revolution; however, the number of points allowed are restricted to give a grid comparable with the $N = 40$ structured grid. As in the previous $N = 40$ cases, 80 iterations were taken to reach one full revolution of the Gaussian wave. Figure 13 shows the grid and contours at (a) $t = \pi/2$, (b) $t = \pi$, (c) $t = 3\pi/2$, and (d) $t = 2\pi$. Note that the mesh generator captured the Gaussian wave extremely well, which then assists the Lagrange–Galerkin method in resolving the wave quite sharply. In fact, we get a 50 percent increase in accuracy by using this strategy over the structured and unstructured grids.

6.2.3.2. Case 3. Figure 14 shows the adaptive grids and the mass contours for the soliton wave at times (a) $t = 2.5$, (b) $t = 5$, (c) $t = 7.5$, and (d) $t = 10$. Once again 100 iterations were used to integrate the solution. The adaptive grids used are comparable with the $N = 64$ structured

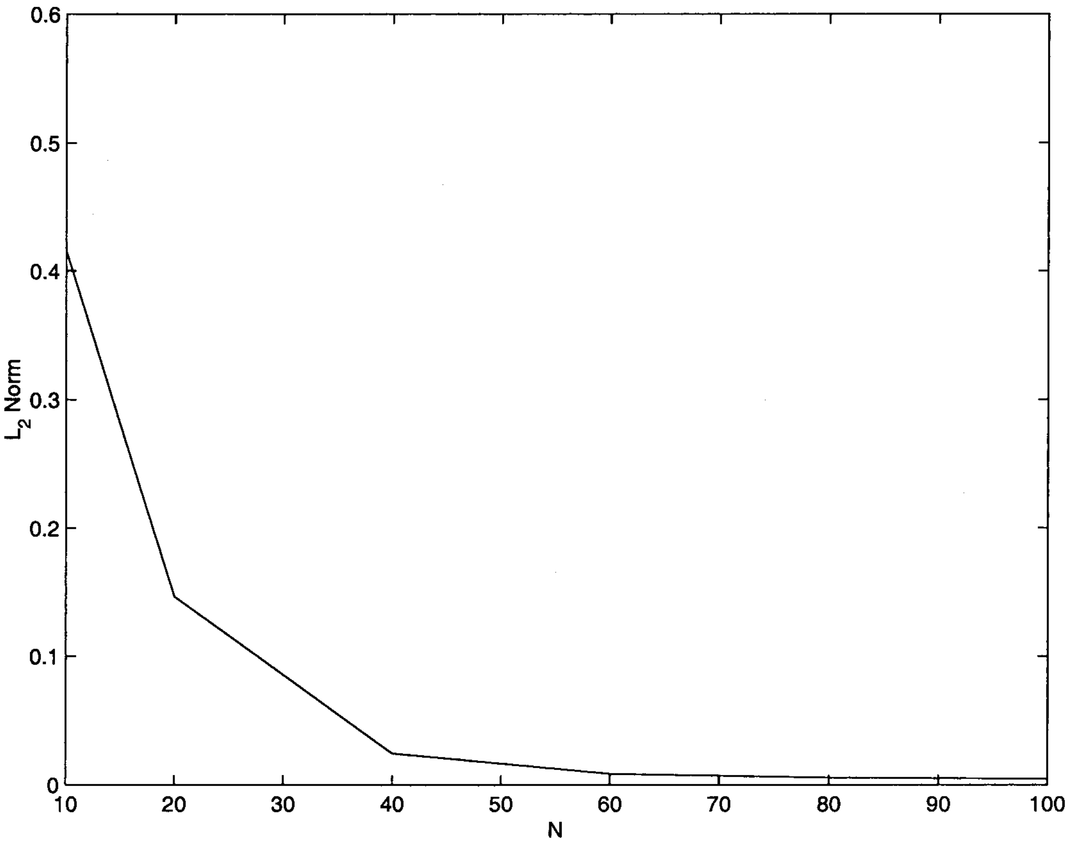


Figure 10. The mass L_2 norm as a function of the number of unstructured elements N for case 1 at $t = 2\pi$.

grid and were allowed to adapt only at the time given above. The solution has been successfully tracked by both the Lagrange–Galerkin method and the adaptive mesh generator. Figure 15 shows the (a) grid, (b) mass, (c) u momentum, and (d) v momentum at time $t = 10$. Figure 15(a) clearly shows two distinct wave structures, one on top of the other. Figure 15(b) shows the mass solution and there are clearly two distinct wave structures represented here as well. Finally, Figure 15(c) and (d) show the u and v momentum respectively. The u momentum looks very much symmetric with respect to the y-axis and the v momentum is more or less symmetric, however, these values are extremely small, as the flow is essentially unidirectional along the x-axis. This test case truly shows the power and flexibility of using the weak Lagrange–Galerkin method in conjunction with adaptive unstructured grids. In fact, the accuracy of this solution over the structured and unstructured grids is around 25 percent more accurate.

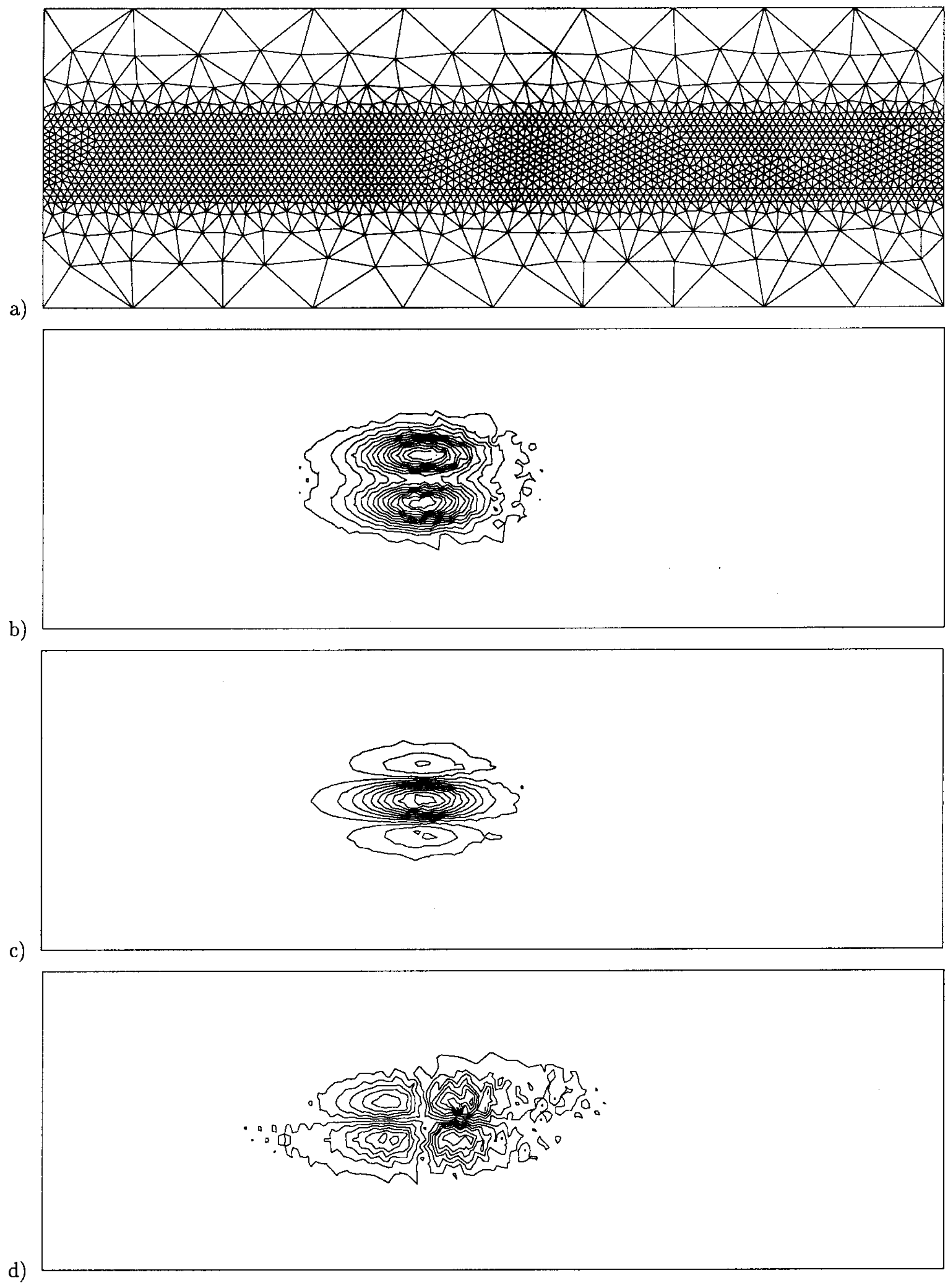


Figure 11. The unstructured $N = 64$ (a) grid and contours for the (b) mass, (c) u momentum, and (d) v momentum for case 3 at $t = 10$.

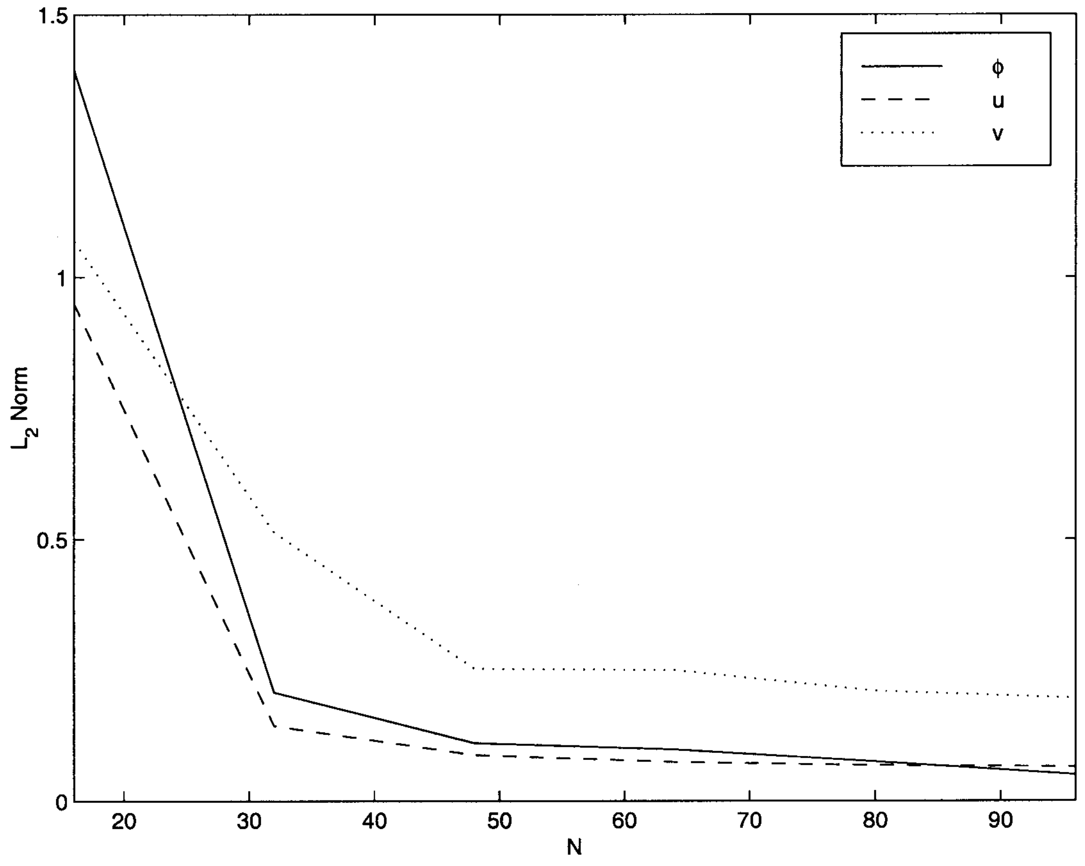


Figure 12. The L_2 norms as a function of the number of unstructured elements N for case 3 at $t=10$.

The reason why Lagrangian methods are so desirable for use with adaptive mesh generators is that the mesh generator constructs very small elements in certain regions, which then alters the maximum local Courant number. If Eulerian methods were used, the global time step would have to be changed in order to satisfy the CFL condition at these small elements. This is true not just for explicit methods (due to instabilities) but for implicit methods (due to inaccuracies) as well since the goal of using adaptive grids is to achieve high accuracy as well as stability; Lagrangian methods offer both.

The other attraction of using adaptive grids is that we do not have to concern ourselves with trying to predict how the solution will behave. Instead, we feed the solution algorithm a crude mesh and then let the Lagrange–Galerkin method and the adaptive mesh generator capture the interesting phenomena of a given problem. However, the initial grid must be sufficiently fine such that it can capture the small-scale features of the flow. In other words, assume we have a high resolution adapted grid at time t^n. In order to maintain the same level of accuracy

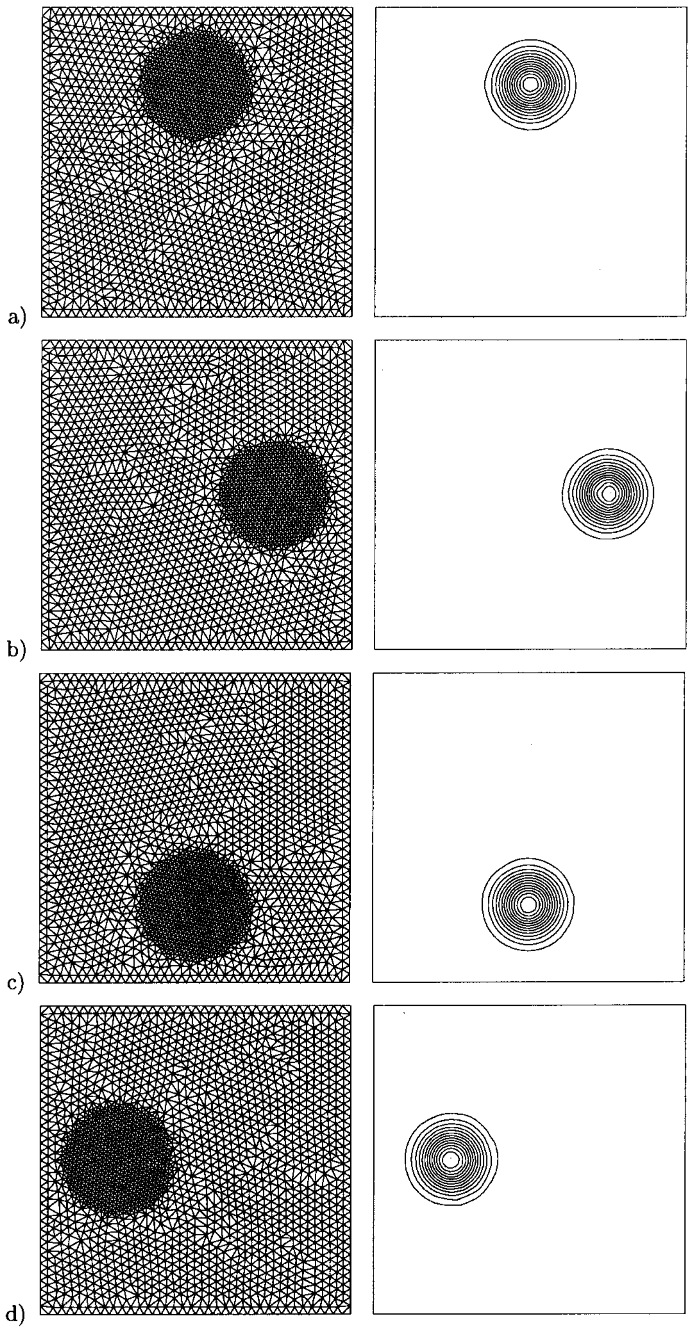


Figure 13. The adaptive unstructured $N = 40$ grid and mass contours at (a) $t = \pi/2$, (b) $t = \pi$, (c) $t = 3\pi/2$, and (d) $t = 2\pi$ for case 1.

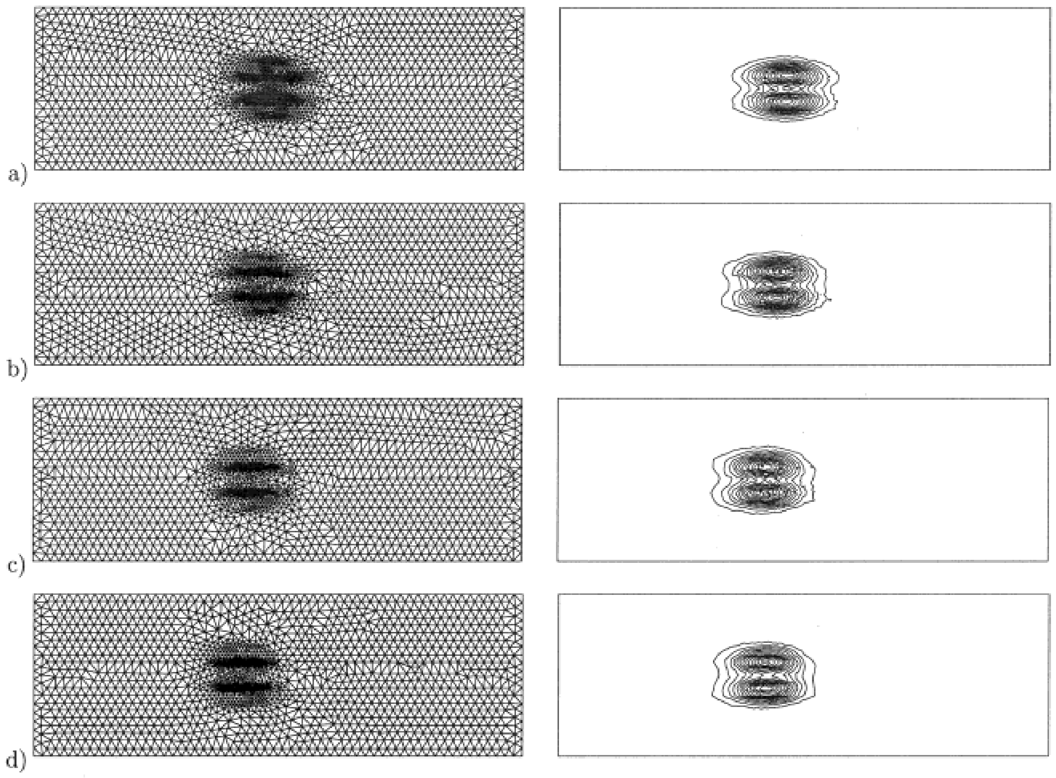


Figure 14. The adaptive unstructured $N = 64$ grid and mass contours at (a) $t = 2.5$, (b) $t = 5$, (c) $t = 7.5$ and (d) $t = 10$ for case 3.

at time t^{n+1} we require a fine mesh where the wave will move to because if we have a crude mesh there, then no matter how good the Lagrange–Galerkin method is it will not be able to maintain the same level of accuracy. For situations where the velocity field is known at all times, such as in air pollution transport, we can use adaptive grids at both ends of the characteristics (times t^n and t^{n+1}) in order to get very precise solutions. This idea can also be used to track certain tracers, fumes, or plumes.

6.3. Computational cost

Figure 16 summarizes the computational cost of using the Lagrange–Galerkin method for the $N = 64$ structured grid for case 3. These results were obtained on a DEC Alpha 8400 (EV5 CPU chip) running only one processor at a clock speed of 300 MHz. The computational cost is divided into three categories: *solve*, *search*, and *fem*. *Solve* represents the percent of the total time to invert the mass and momentum matrices. This includes the LU decompositions and the back substitution of these matrices. *Search* involves the percent of time required for the

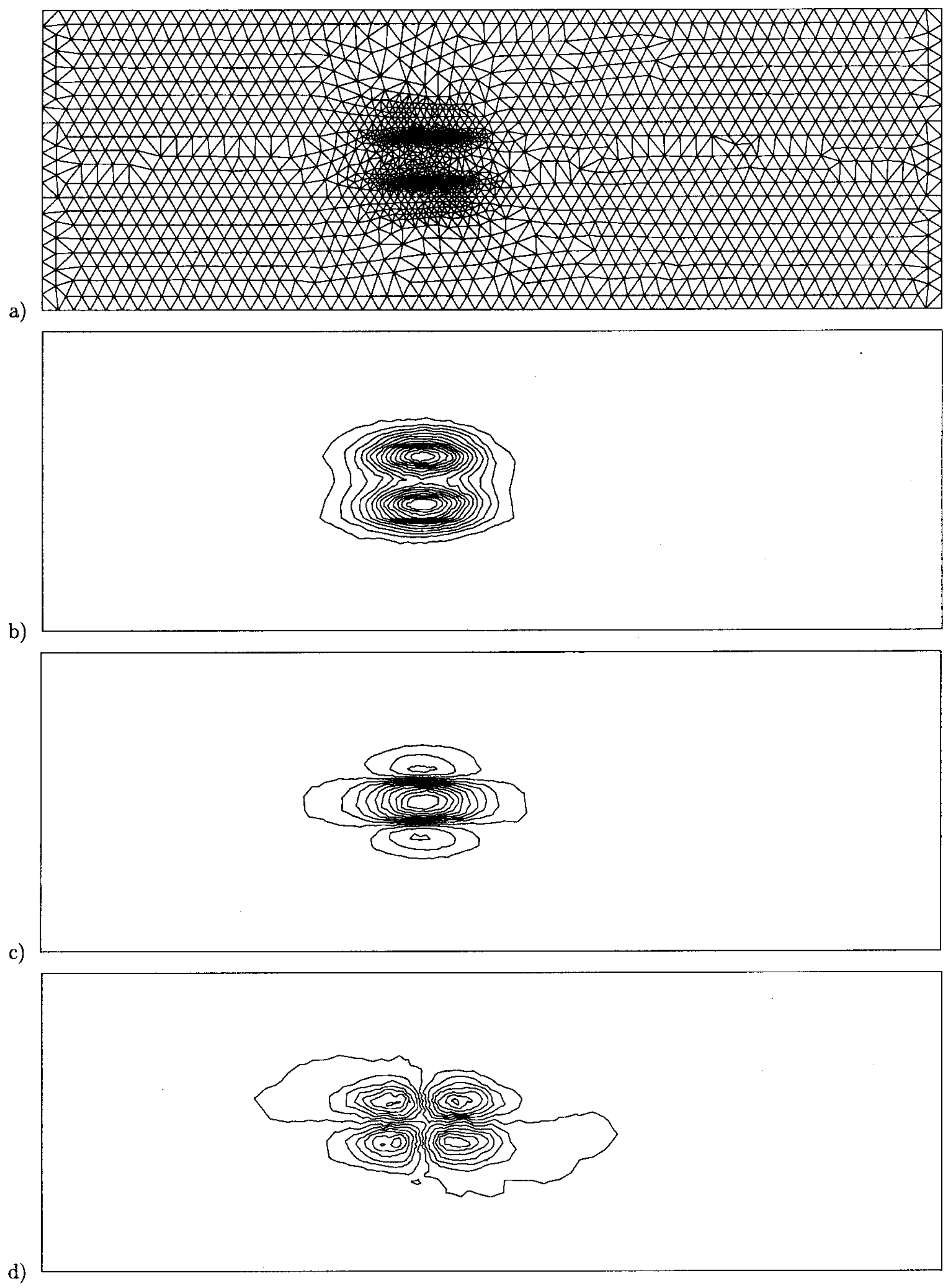


Figure 15. The adaptive unstructured $N=64$ (a) grid and contours for the (b) mass, (c) u momentum, and (d) v momentum for case 3 at $t=10$.

departure points calculations, quadtree searches, and testing for inclusions. *Fem* denotes the percent involving any operation typically associated with the finite element method. This includes the construction of the left- and right-hand side element equations as well as the summation of these element contributions, which result in the global system of equations. The percentage breakdown shown in Figure 16 is based on a 1170-s computation.

The interesting thing about Figure 16 is that neither the searching algorithms nor the finite element operations take any time whatsoever relative to the amount of time required to invert the global matrices. What this means is that if the algorithm must be made more efficient, it will only be accomplished by either constructing a more efficient matrix solver or by picking a better one.

Many other possibilities for increasing the efficiency of the current algorithm can be explored: multi-grid-type methods have proven to be the most efficient for solving linear equations, the LU solver can be streamlined by using a node renumbering scheme, which would then allow the storage of the matrix in a tightly banded form, and finally a parallel implementation of the existing algorithms would remove any of the current bottlenecks. Currently we are exploring multi-grid methods and parallel implementations using the message passing interface (MPI).

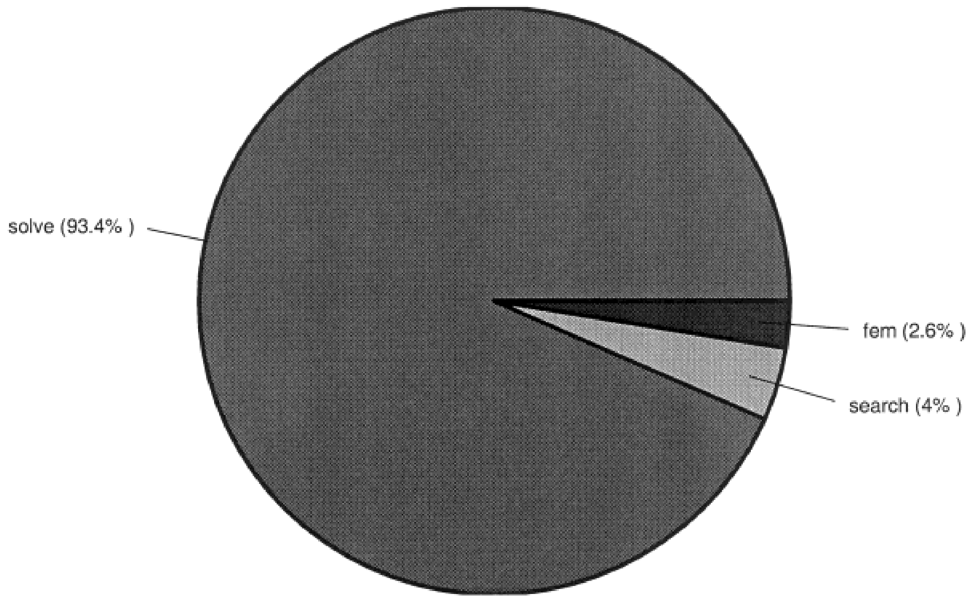


Figure 16. The percentage breakdown of the computational costs incurred by the various operations of the Lagrange–Galerkin method for the structured $N = 64$ grid for case 3 at $t = 10$. The total CPU time consisted of 1170 s.

7. CONCLUSIONS

The weak Lagrange–Galerkin finite element method on adaptive unstructured grids is presented. In order to apply the weak Lagrange–Galerkin method to the shallow water equations they must first be written in conservation form. The combination of the conservative weak Lagrange–Galerkin method along with the governing equations having been written in conservation form results in a highly conservative scheme. In fact, the mass and energy were exactly conserved (up to 99.8 percent) for all the grids and for all of the test cases regardless of the size of the time step. The weak Lagrange–Galerkin method produces a very efficient scheme for solving the shallow water equations that are not just applicable to the two-dimensional planar shallow water equations but are also directly extendible to the two-dimensional spherical and three-dimensional Cartesian shallow water equations. This generalization is possible through the use of the natural co-ordinates as the finite element basis functions.

Although the shallow water equations are a coupled system of non-linear first-order PDEs, after discretizing by the weak Lagrange–Galerkin method they yield a decoupled system of linear first-order equations. The decoupling occurs between the mass and the momentum equations; the n momentum equations (for R^n space) remain weakly coupled through the Coriolis term. In this paper we have described in detail the application of the weak Lagrange–Galerkin method to the shallow water equations. In additions, we have shown how to integrate all of the equations exactly, not just those at the arrival points (the Eulerian elements at time t^{n+1}) but also the departure points (the Lagrangian elements at time t^n). However, it was found that the numerical integration approach yielded results equal or close to the exact integration approach while costing a factor of two less. But since this is such and elegant method, further work to increase its efficiency needs to be explored.

Perhaps one of the most important aspects of a Lagrangian method is the correct capturing of the fluid particle trajectories. In this paper, we have introduced two new methods for calculating these trajectories: the Runge–Kutta (RK) and the composite mid-point rule (CM). Both methods are shown to be extremely accurate not just for steady state velocity fields but for transient ones as well. The RK method performed better than the CM method but further improvements can always be made.

Linear triangular elements are used in this study, which give three advantages over quadrilaterals: exact or numerical integration can be used to integrate all of the finite element integrals; regardless of shear or rotational aspects of the flow, triangles will not become non-convex polygons whereas this danger exists and is very probable with quadrilaterals; and finally adaptive unstructured grids can be used. The adaptive unstructured grid generator presented is an advancing front method, which uses the idea of the weak Lagrange–Galerkin method to project the values at the grid points from the old mesh onto the newly adapted mesh. This idea results in a difference in conservation of mass and energy from one mesh to the other on the order of 1×10^{-8} percent.

Three test cases were run in order to validate the algorithms: a solid body rotation of a Gaussian with a steady state velocity field, a solid body rotation of a Gaussian with a transient velocity field, and a westward traveling soliton wave. The first two test cases are given primarily to show that the convergence rate of the weak Lagrange–Galerkin method is quadratic (second-order) and to compare the two methods for calculating the fluid particle trajectories. The soliton case tests the ability of the scheme to handle a non-linear phenomenon and it performed extremely well, not only for the structured and unstructured grids but also for the adaptive unstructured grids.

The results obtained for the three problems demonstrate that the weak Lagrange–Galerkin finite element method with adaptive unstructured triangles offer a viable method for solving the shallow water equations on the plane. This paper describes the approach in detail while future papers will focus on the application of this method for various types of problem. The development of a three-dimensional weak Lagrange–Galerkin shallow water model is currently underway.

ACKNOWLEDGMENTS

The author is grateful to his sponsor, the Office of Naval Research, for supporting this work through program PE-0602435N. He would also like to thank Giri Chukapalli of the San Diego Supercomputer Center for discussions on the weak Lagrange–Galerkin method. In addition, he would like to thank the two anonymous reviewers for their invaluable comments.

APPENDIX A

In order to show that the basis functions vanish along the characteristics, let us assume for simplicity that we can write the basis functions as a linear function of time. Therefore, between times t^n and t^{n+1} we can write the basis functions as

$$\psi_i(\mathbf{x}, t) = \left(\frac{t - t^n}{\Delta t}\right)\psi_i^{n+1}(\mathbf{x}) + \left(\frac{t^{n+1} - t}{\Delta t}\right)\psi_i^n(\mathbf{x}) \tag{29}$$

recalling that the basis functions at t^{n+1} (i.e., at the Eulerian elements, see Figure 1) are

$$\psi_i^{n+1}(\mathbf{x}) = \frac{a_i x + b_i y + c_i}{\det \triangle_{i,j,k}} \tag{30}$$

where

$$\det \triangle_{i,j,k} = (\mathbf{x}_j - \mathbf{x}_i) \times (\mathbf{x}_k - \mathbf{x}_i)$$

and

$$a_i = y_j - y_k, \qquad b_i = x_k - x_j, \qquad c_i = x_j y_k - x_k y_j$$

Assuming a constant velocity field, from the trajectories we can write

$$x_i^n = x_i - u\Delta t \quad \text{and} \quad y_i^n = y_i - v\Delta t$$

where

$$x_i^{n+1} \equiv x_i \quad \text{and} \quad y_i^{n+1} \equiv y_i$$

Plugging in the $\mathbf{x}^n$ values into Equation (30) and simplifying gives

$$\psi_i^n(\mathbf{x}) = \psi_i^{n+1}(\mathbf{x}) + \Delta t \frac{a_i u + b_i v}{\det \triangle_{i,j,k}} \tag{31}$$

Substituting Equations (30) and (31) into Equation (29) and grouping like terms yields

$$\psi_i(\mathbf{x}, t) = \psi_i^{n+1}(\mathbf{x}) + (t^{n+1} - t)\left(\frac{a_i u + b_i v}{\det \triangle_{i,j,k}}\right)$$

Differentiating yields

$$\frac{\partial \psi_i}{\partial t} = -\left(\frac{a_i u + b_i v}{\det \triangle_{i,j,k}}\right)$$

$$\frac{\partial \psi_i}{\partial x} = -\left(\frac{a_i}{\det \triangle_{i,j,k}}\right)$$

and

$$\frac{\partial \psi_i}{\partial y} = -\left(\frac{b_i}{\det \triangle_{i,j,k}}\right)$$

Substituting into the characteristic equation

$$\frac{\partial \psi_i}{\partial t} + u \frac{\partial \psi_i}{\partial x} + v \frac{\partial \psi_i}{\partial y} = 0$$

we can see that the basis functions do indeed vanish along the characteristics. A similar analysis can be carried out for the general case where u and v are no longer constant; however, it is considerably more complicated.

REFERENCES

1. Iskandarani M, Haidvogel DB, Boyd JP. A staggered spectral element model with application to the oceanic shallow water equations. *International Journal for Numerical Methods in Fluids* 1995; **20**: 393–414.
2. Ma H. A spectral element basin model for the shallow water equations. *Journal of Computational Physics* 1993; **109**: 133–149.
3. Taylor M, Tribbia J, Iskandarani M. The spectral element method for the shallow water equations on the sphere. *Journal of Computational Physics* 1997; **130**: 92–108.
4. Giraldo FX. The Lagrange–Galerkin spectral element method on unstructured quadrilateral grids. *Journal of Computational Physics* 1986; **147**: 114–146.

5. Taylor M, Wingate B. The Fekete collocation points for triangular spectral elements. *SIAM Journal on Numerical Analysis* (to appear).
6. Bercovier M, Pironneau O. Characteristics and the finite element method. In *Proceedings of the Fourth International Symposium on Finite Element Methods in Flow Problems*, Kawai T (ed.). North-Holland Publishing: New York, 1982; 67.
7. Bermejo R. A Galerkin-characteristic algorithm for transport–diffusion equation. *SIAM Journal on Numerical Analysis* 1995; **32**: 425–454.
8. Douglas J, Russell TF. Numerical methods for convection-dominated diffusion problems based on combining the method of characteristics with finite element or finite difference procedures. *SIAM Journal on Numerical Analysis* 1982; **19**: 871–885.
9. Priestley A. Exact projections and the Lagrange–Galerkin a realistic alternative to quadrature. *Journal of Computational Physics* 1994; **112**: 316–333.
10. Staniforth A, Côté J. Semi-Lagrangian integration schemes for atmospheric models—a review. *Monthly Weather Review* 1991; **119**: 2206–2223.
11. Zienkiewicz OC, Ortiz P. A split-characteristic based finite element model for the shallow water equations. *International Journal for Numerical Methods in Fluids* 1995; **20**: 1061–1080.
12. Donéa J. A Taylor–Galerkin method for convective transport problems. *International Journal for Numerical Methods in Engineering* 1984; **20**: 101–118.
13. Currie IG. *Fundamental Mechanics of Fluids*. Mc-Graw-Hill: San Francisco, 1974; 9.
14. Benque JP, Ronat J. Quelques difficultes de modules numerique en hydralique. In *Computing Method in Applied Sciences and Engineering V*, Glowinski R, Lions JL (eds). North Holland Publishing: New York, 1982; 471.
15. Benque JP, Labadie G, Ronat J. A new finite element method for Navier–Stokes equations coupled with a temperature equation. In *Proceedings of the Fourth International Symposium on Finite Element Method in Flow Problems*, Kawai T (ed.). North-Holland Publishing: New York, 1982; 295.
16. Celia MA, Russell TF, Hera I, Ewing RE. An Eulerian–Lagrangian localized adjoint method for the advection–diffusion equation. *Advances in Water Resources* 1990; **13**: 187–206.
17. Healy RW, Russell TF. Solution of the advection–dispersion equation in two dimensions by a finite volume Eulerian–Lagrangian localized adjoint method. *Advances in Water Resources* 1998; **21**: 11–26.
18. Wang H, Ewing RE, Celia MA. Eulerian–Lagrangian localized adjoint methods for reactive transport with biodegradation. *Numerical Methods for Partial Differential Equations* 1995; **11**: 229–254.
19. Wang H, Ewing RE, Qin G, Lyons SL, Al-Lawatia M, Man S. A family of Eulerian–Lagrangian localized adjoint methods for multi-dimensional advection-reaction equations. *Journal of Computational Physics* 1999; **152**: 120–163.
20. Binning P, Celia MA. A finite volume Eulerian–Lagrangian localized adjoint method for solution of the contaminant transport equation two-dimensional multiphase flow systems. *Water Resources Research* 1996; **32**: 103–114.
21. Giraldo FX. Lagrange–Galerkin methods on spherical geodesic grids. *Journal of Computational Physics* 1997; **136**: 197–213.
22. Petera J, Nassehi V. A new two-dimensional finite element model for the shallow water equations using a Lagrangian framework constructed along fluid particle trajectories. *International Journal for Numerical Methods in Fluids* 1996; **39**: 4159–4182.
23. Chukapalli G. Weather and climate numerical algorithms: a unified approach to an efficient, parallel implementation. PhD thesis, University of Toronto, 1997.
24. Giraldo FX, Neta B. Stability analysis of Eulerian and semi-Lagrangian finite element formulation of the advection–diffusion equation. *Computers and Mathematics with Applications* 1999; **38**: 97–112.
25. Giraldo FX. Trajectory calculations for spherical geodesic grids in Cartesian space. *Monthly Weather Review* 1999; **127**: 1651–1662.
26. Giraldo FX. A space marching adaptive remeshing techniques applied to the 3D Euler equations for supersonic flow. PhD thesis, University of Virginia, 1995.
27. Boyd JP. Equatorial solitary waves. Part 3: westward-travelling modons. *Journal of Physical Oceanography* 1985; **15**: 46–54.
28. Boyd JP. Equatorial solitary waves. Part 1: Rossby solitons. *Journal of Physical Oceanography* 1980; **10**: 1699–1717.