ELSEVIER

Comput. Methods Appl. Mech. Engrg. 191 (2002) 5345–5363

Computer methods in applied mechanics and engineering

www.elsevier.com/locate/cma

An efficient characteristic Galerkin scheme for the advection equation in 3-D

M.R. Kaazempur-Mofrad, C.R. Ethier [*]

Department of Mechanical and Industrial Engineering, University of Toronto, 5 King's College Road, Toronto, ON, Canada M5S 3G8

Received 25 May 2001; received in revised form 20 February 2002

Abstract

Conventional Galerkin-based finite element algorithms perform poorly when modeling advective transport. Here we develop and test a characteristic Galerkin scheme to solve the unsteady three-dimensional advective transport equation. The algorithm uses tracking of the Gaussian quadrature points to project the information from the Eulerian background grid to the Lagrangian grid. Numerical experiments were carried out to investigate the performance of this scheme. Despite speculations to the contrary in the literature, this scheme does not suffer from instability problems. Further, it is conservative, and shows good phase characteristics with slight numerical diffusion. We conclude that the characteristic Galerkin method is a viable and efficient scheme for solving advection problems in three dimensions.
© 2002 Elsevier Science B.V. All rights reserved.

Keywords: Method of characteristics; Finite element method; Advection; Three-dimensional; Unstructured grid

1. Introduction

Consider the linear advection equation

$$\frac{\partial c}{\partial t} + \vec{V} \cdot \nabla c = 0, \tag{1}$$

where $c(\vec{x}, t)$ is an advected scalar and $\vec{V}(\vec{x}, t)$ is a known velocity field. The initial condition is

$$c(\vec{x}, t = 0) = c_0(\vec{x}) \tag{2}$$

and boundary data are defined on the inflow portion of the computational domain.

It is well known that the conventional Galerkin finite element method of spatial discretization, along with classical time discretization methods, fail to produce satisfactory solutions to Eq. (1). Spurious node-to-node oscillations are generated, particularly in regions of high gradients in the solution. These

[*] Corresponding author. Tel.: +1-416-978-6728; fax: +1-416-978-7753.
E-mail address: ethier@mie.utoronto.ca (C.R. Ethier).

oscillations are related to the characteristic lines playing a dominant role in the solution of such hyperbolic equations. Since time and space are linked through the characteristics, care must be exercised to properly "balance" the temporal and spatial discretization approaches [31].

In general, oscillations may be eliminated by spatial mesh refinement in zones of high gradients, accompanied by temporal refinement in time-dependent problems. However, for large problems in 3-D, this is usually not feasible. With the practical relevance of advective processes to many industrial, environmental, atmospheric and biomedical problems (for example see [8,10,17,29,30,38,40,41]), there is considerable motivation for developing alternatives to the standard Galerkin method. Such alternate approaches should preclude oscillations regardless of mesh or time step refinement. Further, certain computational attributes are desired for such schemes, namely stability, accuracy, conservation, shape preservation and computational efficiency. No single scheme devised to date can satisfy all these requirements, and thus compromises must be made.

Several numerical techniques have been proposed in the literature to deal with advection. These schemes can be divided into three broad groups: Eulerian, Lagrangian, and Eulerian–Lagrangian. These three classes of schemes are briefly reviewed below in the context of finite element methodology.

Eulerian schemes: Most numerical techniques devised for advection are Eulerian in nature. The Eulerian methods are "local" in the sense that the spatial derivative in the advection equation is approximated based on the information at the neighboring nodal points. The most popular Eulerian schemes for advection are: the streamline upwind Petrov–Galerkin [7,22,28], Galerkin/least-squares [21,37], and Taylor–Galerkin [14–16] methods. All these methods take into account the hyperbolicity of the equation and introduce some kind of stabilization terms. A major drawback of all Eulerian methods is that, for either stability (in the case of explicit time stepping) or accuracy (in the case of implicit time marching) reasons, the time step size (local Courant number) is limited through a Courant–Friedrichs–Levy (CFL) restriction [12].

Lagrangian schemes: In the Lagrangian schemes (see for example [1]) the computational mesh moves along the characteristics (fluid trajectories). Theoretically, the Lagrangian methods are well suited for advective transport problems. But in practice, stretching and shearing of the original fluid particles distort the mesh after a few time steps. Thus, these methods are rarely used.

Eulerian–Lagrangian schemes: In Eulerian–Lagrangian or characteristic methods, advection is treated by using a Lagrangian tracking algorithm along characteristic lines while keeping the convenience of a fixed computational grid. Points from within the Eulerian grid are tracked backward (or forward) along the characteristics over the time step, thereby forming a Lagrangian grid. Numerical information from the previous time level is projected from the background Eulerian grid onto the Lagrangian grid. A significant advantage of these methods is that, owing to the Lagrangian nature of the advection step, the CFL restriction is relaxed. Moreover, because the spatial and temporal discretizations are combined as a result of the Lagrangian tracking, the temporal discretization error is reduced markedly. In effect, the characteristic methods perform the temporal discretization on the total derivative by tracking fictitious fluid particles during each time step. Thus, the time truncation error is proportional to the total derivative of the solution with respect to time, which is typically smaller than the local time derivatives. Hence, characteristic schemes have the potential to be more accurate than their Eulerian counterparts.

Several versions of the characteristic method have been proposed in the literature (see for example [6,23,34,39]). The most closely related formulation to our proposed scheme is the Eulerian–Lagrangian localized adjoint method (ELLAM) [9], which, unlike other characteristic methods, retains the Eulerian form of the transport equation and defines the test (weighting) functions to satisfy the adjoint operator of the transport equation [9,33]. A finite-volume extension of ELLAM (abbreviated as FV-ELLAM) has been developed in one-dimensional (1-D) [18], two-dimensional (2-D) [4,19], and later in three-dimensional (3-D) [5,20] as a method for solution of the advection–dispersion-reaction equation governing contaminant transport in groundwater problems.

In the finite element context, the characteristic schemes combine the classical method of characteristics with a Galerkin finite element approximation, and hence are called characteristic Galerkin schemes. Since our overall goal is to apply the advection solver in complex 3-D geometries, we have focused in our work on the finite element based techniques, since they are well proven to be capable of handling geometrically complex computational domains. A detailed description of characteristic Galerkin schemes is presented in Section 3.

2. Statement of purpose

Our objective was to develop, test, and characterize a 3-D characteristic Galerkin algorithm for treatment of the pure advection equation (Eq. (1)). This was done in the context of a fully unstructured mesh arising from a tetrahedral discretization of computational domains. As a first step, we implemented the scheme in 2-D form, using both Dirichlet and periodic boundary conditions. Previous researchers have suggested a "piecewise exact" method for projecting the information from the background Eulerian grid onto the Lagrangian grid for stable 2-D characteristic Galerkin schemes [33,36]. This method works well, but is difficult and computationally expensive to extend to 3-D. Therefore, in this work, a different approach based on tracking of Gaussian quadrature points has been employed. We then extend this scheme to 3-D with relative ease.

3. Description of the characteristic Galerkin scheme

Consider again the linear advection equation (Eq. (1)). Denote the position of a fluid element at time t, which was (or will be) at $\vec{x}$ at time s, by $\vec{X}(\vec{x},s;t)$. Then the characteristic curves of this equation, along which c remains constant, are defined by

$$\frac{\mathrm{d}\vec{\mathrm{X}}}{\mathrm{d}t}(\vec{x},s;t) = \vec{V}(\vec{X}(\vec{x},s;t),t) \tag{3}$$

with

$$\vec{X}(\vec{x},s;s) = \vec{x}. \tag{4}$$

Once the characteristics are known from Eq. (3), the solution to the advection equation is

$$c(\vec{X}(\cdot,t;t+\tau),t+\tau) = c(\cdot,t), \tag{5}$$

where τ is a time interval. When discretized in time with time step Δt, $t^n = n\Delta t$, the characteristics can be used to define

$$\vec{x} = \vec{X}(\vec{y},t^{n+1};t^n) \tag{6}$$

and

$$\vec{y} = \vec{X}(\vec{x},t^n;t^{n+1}). \tag{7}$$

Here $\vec{x}$ and $\vec{y}$ denote the departure point (at the "foot" of the characteristic line) at time t^n, and the arrival point (at the "head" of the characteristic line) at time t^{n+1}, respectively (see Fig. 1).

There are two ways to proceed with the characteristic Galerkin method. They are referred to as "weak" and "direct" methods [3,32,36], and are described below.

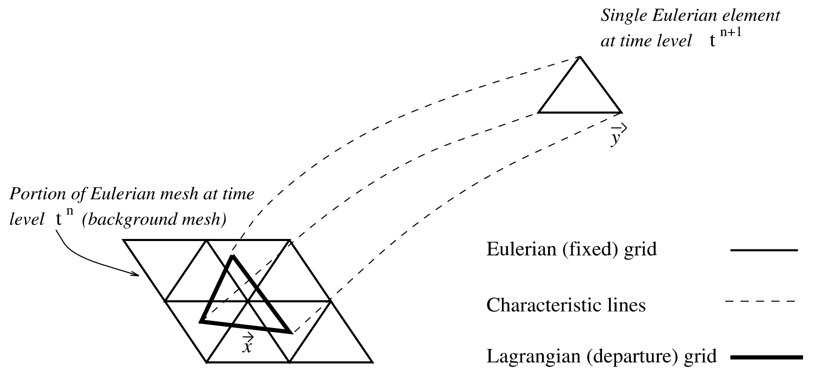

Fig. 1. Schematic of Eulerian and Lagrangian grids for a characteristic Galerkin method.

3.1. Weak characteristic Galerkin scheme

Multiplication of Eq. (1) by a test function $w_i(\vec{x}, t)$ and integration over the spatial domain Ω, and over the time interval between t^n and t^{n+1}, yields the following weak formulation:

$$\int_\Omega (c^{n+1}w_i^{n+1} - c^n w_i^n)\,\mathrm{d}\Omega - \int_{t^n}^{t^{n+1}}\int_\Omega \left(\frac{\partial w_i}{\partial t} + \vec{V}\cdot\nabla w_i\right)c\,\mathrm{d}\Omega\,\mathrm{d}t - \int_{t^n}^{t^{n+1}}\int_\Omega c w_i \nabla\cdot\vec{V}\,\mathrm{d}\Omega\,\mathrm{d}t + \int_{t^n}^{t^{n+1}}\int_\Gamma w_i c\vec{V}\cdot\vec{n}\,\mathrm{d}\Gamma\,\mathrm{d}t = 0, \tag{8}$$

where Γ is the domain boundary with outward-facing unit normal vector $\vec{n}$. We now restrict attention to incompressible flows ($\nabla\cdot\vec{V} = 0$) and choose the test functions to follow the physics of the problem, i.e.

$$\frac{\partial w_i}{\partial t} + \vec{V}\cdot\nabla w_i = 0 \tag{9}$$

with the condition that $w_i^{n+1} = \phi_i$, where ϕ_i are the (Eulerian) basis functions of the discrete functional space. Then, Eq. (8) leads to

$$\int_\Omega c^{n+1}w_i^{n+1}\,\mathrm{d}\Omega = \int_\Omega c^n w_i^n\,\mathrm{d}\Omega + \int_{t^n}^{t^{n+1}}\int_\Gamma w_i c\vec{V}\cdot\vec{n}\,\mathrm{d}\Gamma\,\mathrm{d}t, \tag{10}$$

where c^{n+1} is approximated by finite element basis functions and time-dependent nodal values in the form $\sum_j C_j^{n+1}\phi_j$. Ignoring the boundary term, [1] e.g. in the presence of periodic boundary conditions or in the case where $\vec{V}\cdot\vec{n} = 0$, and expanding c^{n+1} and c^n, Eq. (10) yields

$$\sum_j C_j^{n+1}\int_\Omega \phi_j(\vec{y})\phi_i(\vec{y})\,\mathrm{d}\vec{y} = \sum_j C_j^n \int_\Omega \phi_j(\vec{x})\phi_i(\vec{y})\,\mathrm{d}\vec{x}. \tag{11}$$

[1] In general, a boundary condition has to be applied when a characteristic or path line between t^n and t^{n+1} crosses the boundary. In such a case, the integration of Eq. (3) stops at time t^*, $t^n < t^* < t^{n+1}$. Application of Dirichlet boundary condition is the focus of Section 3.5 of this work.

3.2. Direct characteristic Galerkin scheme

A more direct formulation can be derived from Eq. (5). When discretized in time, Eq. (5) results in

$$c^{n+1}(\vec{y}) = c^n(\vec{x}). \tag{12}$$

The direct scheme proceeds from here in line with the Galerkin technique. Eq. (12) is multiplied by a weighting function w and is integrated over the spatial domain (at time t^{n+1}). The weighting functions w are chosen as equal to the Eulerian basis functions (at time t^{n+1}):

$$\int_\Omega c^{n+1}(\vec{y}) w \, \mathrm{d}\Omega = \int_\Omega c^n(\vec{x}) w \, \mathrm{d}\Omega. \tag{13}$$

The scalar field is now expressed as a function of finite element basis functions and time-dependent coefficients (nodal values), resulting in

$$\sum_j C_j^{n+1} \int_\Omega \phi_j(\vec{y}) \phi_i(\vec{y}) \, \mathrm{d}\vec{y} = \sum_j C_j^n \int_\Omega \phi_j(\vec{x}) \phi_i(\vec{y}) \, \mathrm{d}\vec{y}. \tag{14}$$

Comparing Eqs. (14) and (11), we note that the two alternative derivations result in the same system, except that in the weak method the right-hand side integral is performed on $\mathrm{d}\vec{x}$ whereas in the direct method this integral is done on $\mathrm{d}\vec{y}$. If J is the Jacobian of the mapping from $\vec{x}$ to $\vec{y}$, we have [11,36]

$$\frac{\mathrm{d}|J|}{\mathrm{d}t} = (\nabla \cdot \vec{V})|J|. \tag{15}$$

For incompressible flows $\nabla \cdot \vec{V} = 0$, and hence $|J| = 1$, i.e. $\mathrm{d}\vec{x} = \mathrm{d}\vec{y}$. That is, the weak and direct formulations are mathematically equivalent when applied to incompressible flows [36].

In the rest of this paper we focus on the direct formulation. The left-hand side of Eq. (14) leads to a linear system with a symmetric, positive-definite mass matrix (having all negative and real eigenvalues) which is easily inverted using standard methods, such as a preconditioned conjugate-gradient technique. This avoidance of non-symmetric contributions from the advection operator is a major advantage of the characteristic Galerkin method.

We now restrict our attention to linear triangular elements (in 2-D) and linear tetrahedral elements (in 3-D). Area coordinates (2-D) and volume coordinates (3-D) (L_k; $k = 1, \ldots, d$, where $d = 3$ for 2-D and $d = 4$ for 3-D), defined over the triangular and tetrahedral elements respectively, are employed as basis functions. Then the left-hand side of Eq. (14) becomes

$$\sum_{\text{ele}} \sum_{j=1}^{d} C_j^{n+1} \int_\Delta \phi_j(\vec{y}) \phi_i(\vec{y}) \, \mathrm{d}\vec{y} = \sum_{\text{ele}} \sum_{j=1}^{d} C_j^{n+1} \int_\Delta L_j L_i \, \mathrm{d}\vec{y} \quad i = 1, \ldots, d, \tag{16}$$

where Δ is the area/volume of the triangle/tetrahedral element in 2-D/3-D, respectively. This integral can be exactly determined by using well-known identities. The difficult task is the computation of the right-hand side of Eq. (14) (RHS projection), which is the subject of the next section.

3.3. RHS projection

RHS projection represents the L_2-projection of numerical information from the Eulerian background grid onto the Lagrangian grid. The $\phi_i(\vec{y})$ are piecewise linear over the Eulerian element $\vec{y}$, whereas $\phi_j(\vec{x})$ are piecewise linear over the Lagrangian element $\vec{x}$. Hence, exact evaluation of the RHS projection is in general not possible, and an approximate method is required.

It has been suggested that approximate L_2-projection from one grid to another may cause instability [32]. In order to preclude this instability, a "piecewise exact" RHS projection method was proposed by Priestley [36]. The piecewise exact method starts by backtracking the vertices of the Eulerian grid to locate their departure points at t^n. The interior of the element is assumed to have been transported linearly, i.e. the triangular element is assumed to backtrack to a triangle. The intersection points of the Lagrangian element with the background Eulerian grid are then located, and subregions over which $\phi(\vec{x})$ and $\phi(\vec{y})$ are both linear are constructed and used to exactly calculate the integral of the elemental RHS over each subregion. This method works well, but is difficult and computationally expensive to extend to 3-D. Thus, a different RHS projection scheme was used in this work. The Gaussian quadrature points of the Eulerian element were backtracked, and the Eulerian background elements that contain the departure points of these quadrature points were located. The scalar values at these departure points were then determined by interpolation from the Eulerian background grid at t^n. Finally, the RHS integral was approximated using Gaussian quadrature:

$$\int_{\Omega} c^n(\vec{x})\phi_i(\vec{y})\,\mathrm{d}\vec{y} = \sum_{\mathrm{ele}} \sum_{k} w_k c^n(\vec{x}_k)\phi_i(\vec{y}_k). \tag{17}$$

Here, the index k refers to the quadrature points and $c^n(\vec{x}_k)$ is the scalar value at the departure point of quadrature point k. According to our experience (see below), as long as one uses reasonable [2] quadrature formulae, this scheme is stable.

It must be remarked that both of the above projection methods are approximate. The piecewise exact method incorporates exact integration over approximate regions, since the transported element may in practice be deformed by the convective process so that the triangular element described by the three convected vertices is not exactly the one that is actually transported. The quadrature method, on the other hand, performs approximate integration using exact quadrature points, i.e. individual quadrature points are moved to their own departure point.

3.4. Searching algorithms

At each time step, and for every quadrature point, the Eulerian background element that contains the departure location of the quadrature point (the departure element) must be located in order to interpolate information onto the departure point ($c^n(\vec{x}_k)$ in Eq. (17)). This elemental searching is a potentially computationally expensive part of the characteristic Galerkin scheme, and it is therefore critical that an efficient search procedure be used. In this work, two algorithms were devised and implemented for elemental searching: (1) zone-based search, and (2) search based on nearest node. They are described below.

3.4.1. Zone-based search

The steps involved in this method are as follows:

- In a preprocessing step, the computational domain is circumscribed with a quadrilateral. This quadrilateral is then sub-divided into quadrilateral (or rectangular, in 2-D) zones (see Fig. 2) whose size is equal to the largest element (edge) size, i.e. maximum distance between two neighboring nodes, in the entire unstructured mesh. A node-to-zone mapping table is then determined and used to create a zone-to-element mapping table. Elements are included if they have at least one node in the zone.

[2] In practice we found that quadrature formulae that use the centroid or vertices of the element as their only quadrature points, as well as those with negative quadrature weights, resulted in instability for long-term simulations.

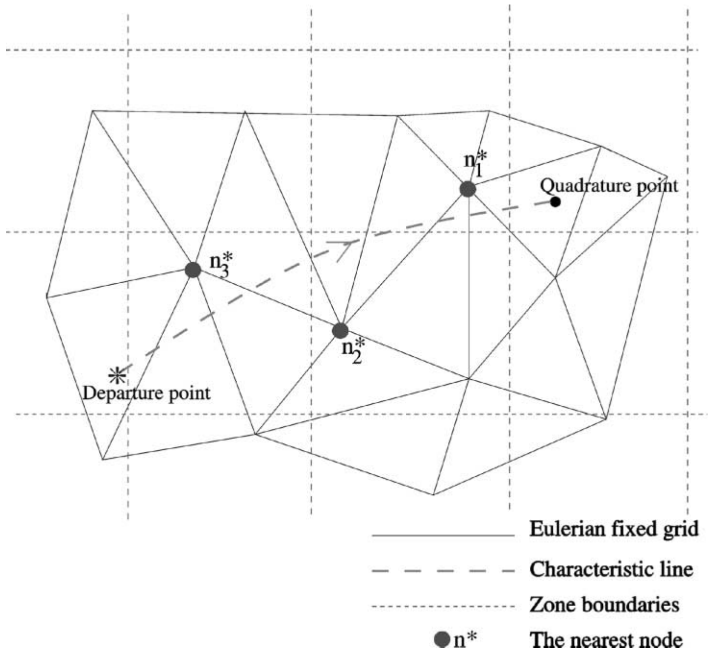

Fig. 2. Schematic of the searching methods: zone-based search and search based on the nearest node. The subscript i in n_i denotes the evolution of the near node, n, from the first guess to the actual nearest node.

- Based on its coordinates, the foot of a characteristic can be trivially assigned to a zone. Then, a search is performed among the elements associated with this zone to locate the departure element. The element search can be done fairly quickly by adopting a two-stage process: first check to see whether the departure point is within the bounding quadrilateral (or rectangle, in 2-D) of each element. If so, compute the point's barycentric coordinates. If all the barycentric coordinates lie in $[0, 1]$, the element is the departure element.

3.4.2. Search based on nearest node

This method is similar to that proposed by Pironneau [34], in which the departure element is searched for element-by-element along the characteristic line upstream of the head of the characteristic. The major difference is that Pironneau's search method is interleaved with stepwise integration of the characteristic lines, given the velocity field is known as a piecewise function. The present algorithm searches for the feet of the characteristics through the closest nodes in an element-by-element fashion, and thus is similar to Pironneau's method while being very efficient at higher Courant numbers.

The specific steps in the search method based on nearest node are as follows:

- In a preprocessing step, create a node-to-node neighbor table and a node-to-element connectivity table.
- Select an arbitrary node from the element that contains the head of the characteristic, and denote this node by n^* (n_1^* in Fig. 2).
- Using the node-to-node neighbor table, compute the distance between the departure point and the immediate neighbors of n^* as stored in the node-to-node neighbor table. As soon as a neighboring node is found that is closer to the departure point than node n^*, switch to that neighboring node, which is now denoted as the new n^* (see n_2^* in Fig. 2).

- Repeat the above step until the distances from the departure point to all neighbors of n^* are greater than the distance from the departure point to n^*.
- Using the node-to-element neighbor table, search the elements containing n^* to find the departure element.

3.4.3. Practical implementation details

In comparing the performance of the above searching methods, it was found that the second method was faster than the first for local Courant numbers below approximately 5. Therefore, in practice we used the following approach: First, depending on the local Courant number, the departure element associated with the first quadrature point of an element was found using either the zone-based search method (if $Cu > 5$) or the search method based on nearest node (if $Cu \leqslant 5$). Once the departure element of the first quadrature point was known, we then used the search method based on nearest node to locate the departure elements of the other quadrature points. In this case, we can exploit the previously found foot as a starting guess, i.e. the seed node for the node-based search was simply a node belonging to the departure element corresponding to the previously treated quadrature point.

For steady state simulations, we only needed to perform the trajectory approximation and the elemental searching once, thanks to the temporally constant velocity field. In this approach, we march Eq. (1) to convergence in pseudo-time, storing the trajectory information in a preprocessing step and using this information for each consecutive pseudo-time step.

3.5. Boundary condition treatment

Dirichlet boundary conditions must be taken into account when a characteristic crosses an inflow boundary between t^n and t^{n+1}. In this work, a variable time step method was devised to deal with such boundary conditions, as follows. For characteristics whose feet were not located in the domain, the time t^* ($t^n < t^* < t^{n+1}$) at which the characteristic intersects the boundary and the location of intersection $\vec{X}_\mathrm{b}$ were determined. Since two unknowns (t^* and $\vec{X}_\mathrm{b}$) existed, the trajectory was approximated iteratively in this procedure. The iteration was seeded by taking $\vec{X}_\mathrm{b}$ as the intersection point between the inflow boundary and an imaginary straight line drawn between the head of the characteristic and the point found outside the domain after integration of the characteristic line back to time t^n. The corresponding t^* was then estimated by assuming that the particle traveled at a constant velocity equal to the velocity at the head of the characteristic. The stopping criterion on the iteration can be chosen depending on how smooth the solution is expected to be near the boundary. In the tests reported herein, we used a stopping criterion of 10^{-2} as the tolerance in $\vec{X}_\mathrm{b}$. This typically required 4–6 iterations. Once the crossing point was found, the integration of the characteristic line was stopped at time t^*, and the Dirichlet scalar boundary data was imposed.

This algorithm works well for inflow boundaries with simple (e.g. planar) shapes. In such cases, if the initial guess for the crossing point was not on the inflow portion of the domain, the time step was decreased by a factor of 2 to increase the local accuracy of the characteristic approximation.

We have further demonstrated the viability of this boundary condition treatment through application of the present algorithm in the simulation of highly advection-dominated mass transport in an anatomically realistic human right coronary artery [26], and in physiologically relevant axisymmetric and asymmetric arterial stenosis [24,27]. In both of these problems there are inflow boundaries with specified concentrations (Dirichlet boundary conditions), impermeable walls, and outflow boundaries.

3.6. Numerical testing procedure

2-D and 3-D versions of the above characteristic Galerkin scheme were implemented in *C* using double precision variable types. The code was tested by simulating the advection of a $\cos^2$-shaped concentration profile in a solid body rotation velocity field. The steep concentration gradients that exist at the sides of the profile in the absence of physical diffusion make this test case valuable for evaluating an advection scheme [2].

In 2-D the initial data was a $\cos^2$ cone with a base radius of 0.25 centered at $(-0.5, 0)$, i.e.

$$c_0(x,y) = \begin{cases} 1 + \cos^2 2\pi r & \text{for } r \leqslant 0.25, \\ 1 & \text{otherwise,} \end{cases} \tag{18}$$

where $r = \sqrt{(x+0.5)^2 + y^2}$. The velocity field was given by $\vec{V}(x,y) = 2\pi(-y,x)$ on the domain $\Omega = [-1,1]^2$. Two types of boundary conditions were used. In some tests, periodic boundary conditions were imposed on the domain boundaries. In other tests, Dirichlet boundary conditions of $c = 1$ were applied at inflow boundaries.

In 3-D the initial data was a $\cos^2$ "ball" with a base radius of 0.25 centered at $(-0.5, 0, 0)$, i.e. of the same form as given in Eq. (18) with $r = \sqrt{(x+0.5)^2 + y^2 + z^2}$. The velocity field was given by $\vec{V}(x,y,z) = 2\pi(-y,x,0)$, on the domain $\Omega = [-1,1]^3$. Dirichlet boundary conditions of $c = 1$ were applied at inflow boundaries.

In all tests, various error norms were computed, including a Euclidean (L_2) error defined as $\sqrt{(\int_\Omega (C_{\text{computed}} - C_{\text{exact}})^2 \, d\Omega)/(\int_\Omega C^2_{\text{exact}} \, d\Omega)}$.

4. Results

4.1. A systematic testing of the present method

Numerical experiments were done in 2-D to investigate the performance of the above scheme as certain critical parameters were varied, namely the temporal and spatial resolutions. Additionally, studies were done with different trajectory approximations, and with various Gaussian quadrature integration formulae. Six meshes with different levels of refinement were used. The coarsest mesh had 10 triangles per side and is denoted as 10×10, and the finest mesh was 130×130.

4.1.1. Gaussian quadrature formulations for the RHS projection

Six different high-order Gaussian quadrature formulae were implemented for evaluating the RHS of Eq. (14). More specifically, the quadrature formulae taken from Cowper [13] consisted of a four-point scheme, a six-point scheme, a seven-point scheme, a nine-point scheme, a twelve-point scheme, and a thirteen-point scheme.

Medium (40×40) to very fine (130×130) grids were used, and a time step of $\Delta t = 0.025$ was selected. The trajectory approximation error was excluded by using the exact trajectories. It is noteworthy that for all of the above quadrature formulae, the scheme was stable without any additional numerical diffusion.

Fig. 3 shows the Euclidean norm of error at the end of one revolution of the cone with the above Gaussian quadrature formulae. As the figure demonstrates, a distinct improvement was observed in the accuracy of the overall solution by increasing the number of quadrature points from 6 to 7. However, no appreciable improvements were discovered by further increasing the number of quadrature points, at least for the mesh spacing used in the present tests. This trend was also consistently observed in the other norms of error. As evident in the figure, the spatial resolution did not appear to have a significant effect on this trend.

Based on the above results, we adopted the seven-point Gaussian quadrature formula for the rest of the 2-D portion of the study.

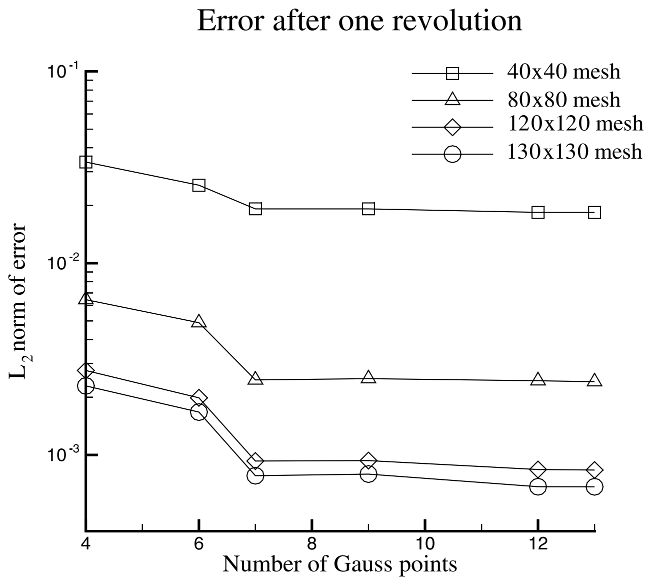

Fig. 3. L_2-norm of error at the end of one full revolution, with various Gaussian quadrature formulae on different meshes for the 2-D test problem. Fourty time steps were used for one revolution, with exact characteristic specification as described in text.

4.1.2. Different trajectory integration schemes

Accurate approximation of the characteristic lines is crucial to the overall accuracy of the characteristic Galerkin method. Several integration methods for Eq. (3) were examined in this study, namely the implicit second-order mid-point method, and the explicit first-, second-, third-, and fourth-order Runge–Kutta methods. Errors in the numerical solution for each of the above predicted trajectories were compared to errors in the solution for the scheme using exact trajectories, available in the case of the rigid body rotation velocity field. A time step of $\Delta t = 0.025$ was used in all runs, corresponding to average Courant numbers in the range 1.5–5.

The implicit second-order mid-point method approximates the trajectory Eq. (3) as

$$\Delta\vec{X}(\vec{x}) = \vec{V}((\vec{X}(\vec{x},t;t^n) - \tfrac{1}{2}\Delta\vec{X}(\vec{x})), t^{n+1/2})\Delta t. \tag{19}$$

Eq. (19) is solved iteratively for the displacement $\Delta\vec{X}$, by

$$\Delta\vec{X}^{(k+1)}(\vec{x}) = \vec{V}((\vec{X}^{(k)}(\vec{x},t;t^n) - \tfrac{1}{2}\Delta\vec{X}^{(k)}(\vec{x})), t^{n+1/2})\Delta t, \tag{20}$$

where $\vec{V}$ is evaluated between mesh points using spatial interpolation. For testing purposes, the iterations were continued until the trajectory changed by less than 10^{-5}. However, in practice it is not recommended to repeat the iteration procedure more than a few times due to efficiency considerations. The Runge–Kutta methods use standard Runge–Kutta integration of Eq. (3).

Fig. 4 shows the L_2-norm of error at the end of one full revolution of the cosine bell on different grids, using the above methods. As expected, the fourth-order Runge–Kutta method was most accurate. The implicit second-order mid-point method, which is widely used in weather forecasting codes, exhibits slightly better accuracy than RK2.

When using the RK4 method, the total CPU cost per grid point per time step was only 0.23% and 0.06% more than that for, respectively, the cheapest method (first-order Euler) and for the commonly used second-order mid-point method (with iterations). Considering the accuracy of the RK4 method and its modest cost, it was considered to be optimal for the present algorithm.

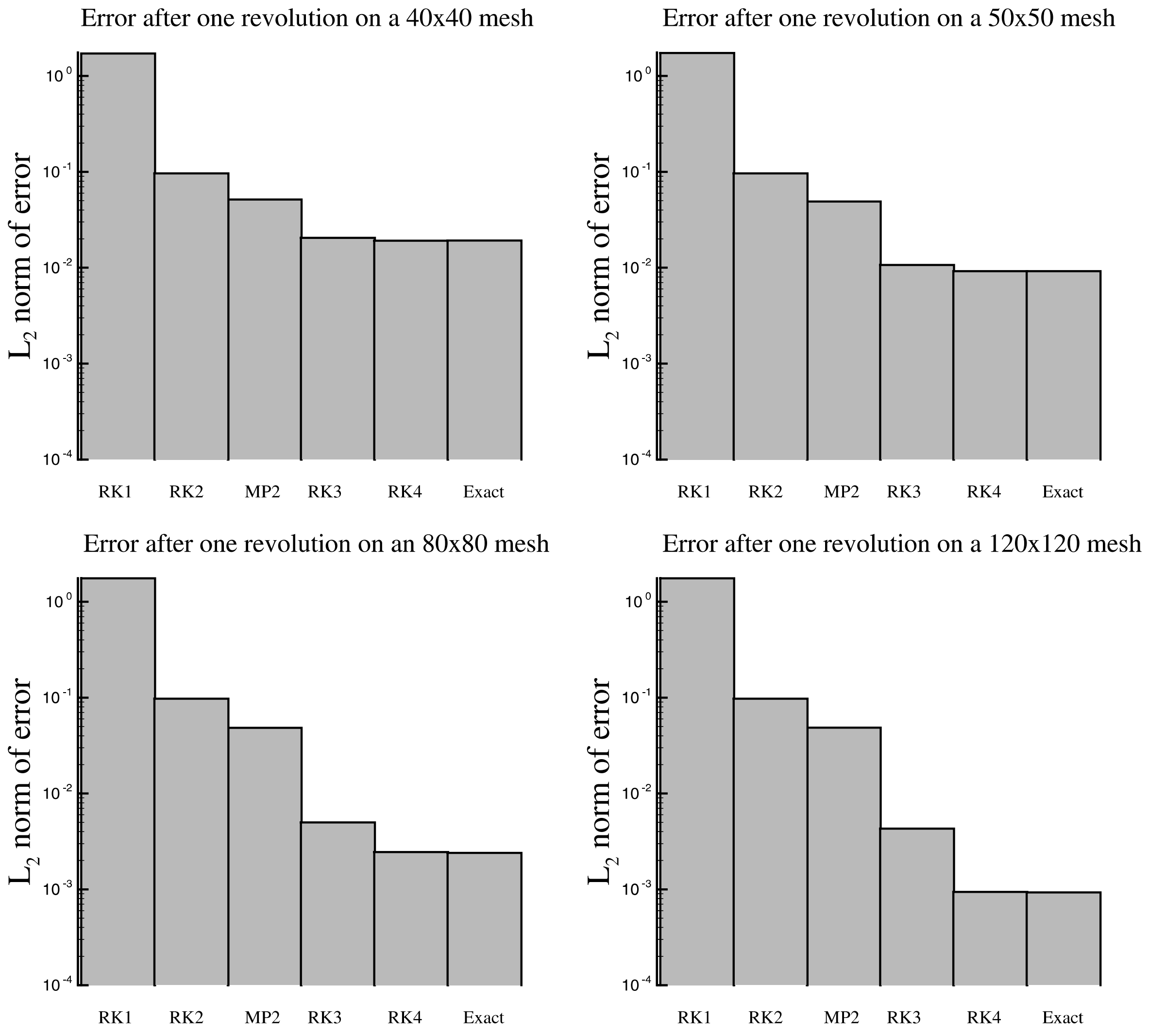

Fig. 4. L_2-norm of error at the end of one full revolution on different meshes for different trajectory approximation methods for the 2-D test problem. In all runs, a time step $\Delta t = 0.025$ and 7 Gauss points were used. Legend: RK1–RK4 represent first through fourth-order Runge–Kutta schemes and MP2 represents the second-order mid-point method.

It must be noted that with smaller time step sizes (for instance, $\Delta t = 0.0125$, corresponding to average Courant numbers of 0.75–2.5) all methods worked well; however for practical situations, we are primarily interested in larger time steps, i.e. larger Courant numbers. As the time step size was increased, the superiority of the fourth-order Runge–Kutta method was more pronounced.

As a result of this investigation, the fourth-order Runge–Kutta method was adopted for later studies.

4.1.3. *Effect of Courant number*

The stability limitations of the method of characteristics are usually due to errors in the approximation of the trajectories, specifically in the location of the departure points at the feet of the characteristic lines.

This effectively places a limit on the Courant number. This limit is, nonetheless, much less restrictive than that imposed by Eulerian schemes.

To investigate the effect of Courant number on the overall accuracy of the scheme, several test cases were done with various temporal resolutions ranging from $\Delta t = 0.003125$ to 0.1, and with several spatial mesh densities. Simulations were stopped at $t = 1$, when the cone completed one full revolution around the origin. The trajectories were approximated by the fourth-order Runge–Kutta method.

L_2-norm errors calculated at $t = 1$ for various temporal and spatial resolutions are consistent with the error bound $E < ((h^2/u\Delta t) + h + u\Delta t)$ suggested by Pironneau [35], where u is a characteristic velocity magnitude (Fig. 5). For the smallest time steps ($\Delta t < 0.02$) the error behavior is dominated by the $h^2/u\Delta t$ term, for intermediate time steps ($0.02 < \Delta t < 0.08$) the error is dominated by the spatial resolution h, and for larger time steps ($\Delta t > 0.08$) the error is dominated by $u\Delta t$.

Pironneau's error bound relationship can be rewritten as

$$E = h\left(\frac{c_1}{Cu} + c_2 + c_3 Cu\right) \tag{21}$$

from which it is predicted that error normalized by h should be a function of Cu only. This prediction is confirmed in Fig. 6, where we plot the L_2-norm of the error vs. the Courant number (calculated at the center of the cone). As the figure demonstrates the data approximately collapse onto a single curve. However, the fit given by Eq. (21) is not perfect, with data scattering somewhat at the higher end of the Courant number range. It is not clear why this is the case. Note that the deviation from the expected curve at large time steps is more pronounced for coarse meshes, suggesting that the scatter could be due to mesh resolution effects (i.e. the ratio of cone size to element size).

We conclude that for this scheme a Courant number in the range of 1.5–8 is the most efficient for this problem, since it yields acceptable error while minimizing the number of required time steps. The Courant number at which the minimum error is obtained is approximately $Cu \approx 3$. This limit is much less restrictive than that imposed by Eulerian schemes where either stability or accuracy reasons limit the Courant number to about 1.

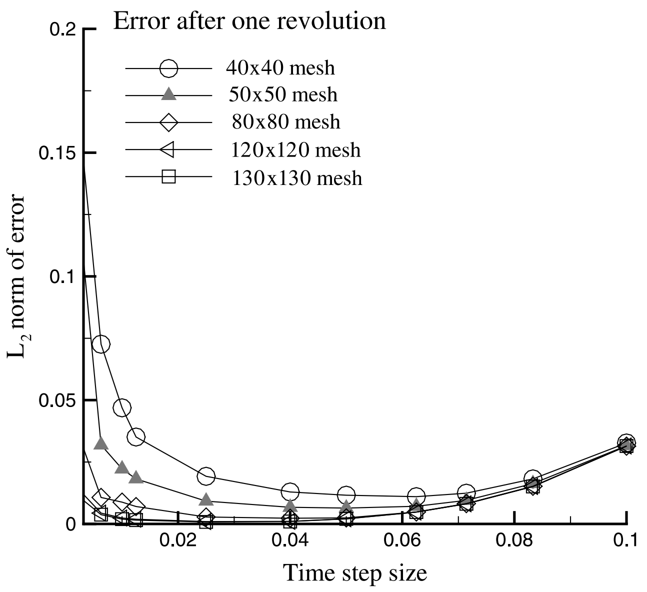

Fig. 5. L_2-norm error for various temporal and spatial resolutions after one complete revolution of the cosine bell (2-D tests with periodic boundary conditions). The coarsest mesh has 40 triangles per side (denoted as 40×40), and the finest mesh is 130×130.

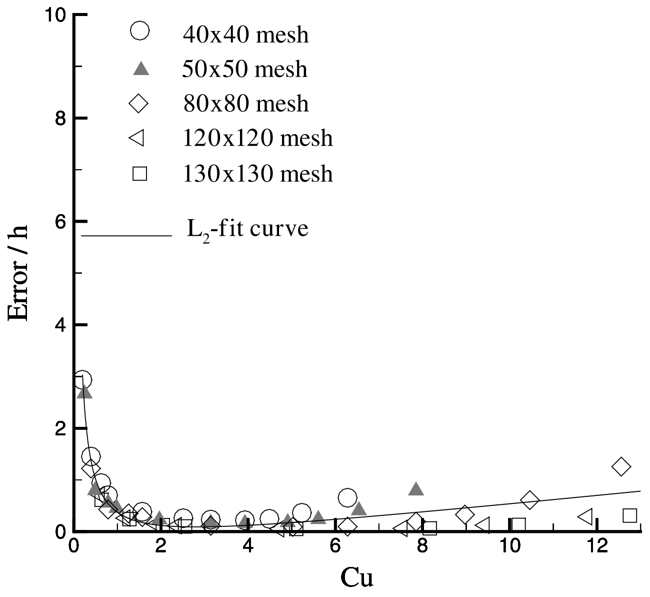

Fig. 6. L_2-norm of error, $\|E\|_{L_2}$, divided by grid size, h, vs. Courant number for various temporal and spatial resolutions after one complete revolution of the cosine bell (2-D tests with periodic boundary conditions). The data is least squares fit by $E/h = c_1/Cu + c_2 + c_3Cu$ with coefficients $c_1 = 0.68 \pm 0.03$ (mean ± SD), $c_2 = -0.39 \pm 0.06$ and $c_3 = 0.09 \pm 0.01$.

4.2. Typical two-dimensional results

Figs. 7 and 8 show representative results for a grid with 80 triangles on a side, and a time step of $\Delta t = 0.025$, corresponding to an average Courant number of $Cu_{\text{avg}} \approx 3.5$ on the domain. For this time step

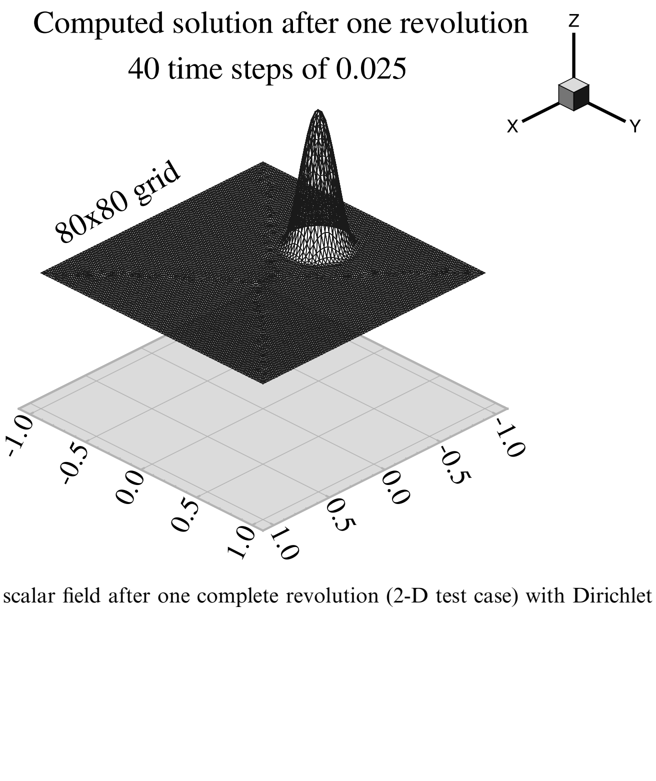

Fig. 7. The computed scalar field after one complete revolution (2-D test case) with Dirichlet boundary conditions.

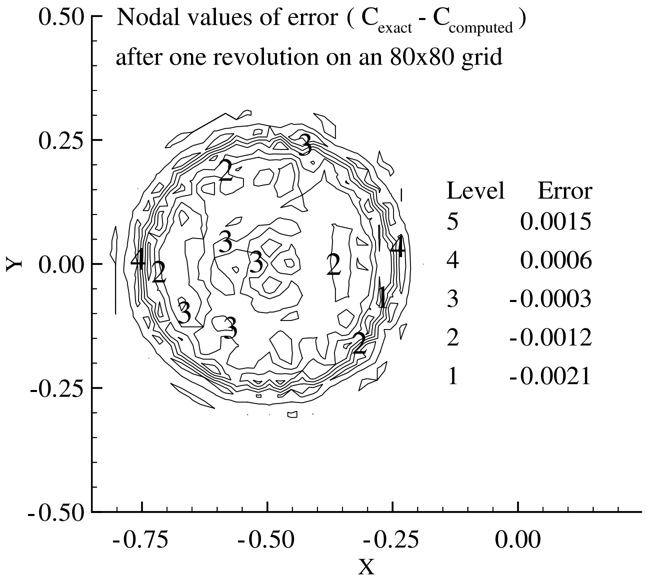

Fig. 8. The global error field after one complete revolution (2-D test case) with Dirichlet boundary conditions.

size, 40 time steps are required for the cosine bell to complete one full rotation around the domain. The CPU time for this configuration was about 0.03 s per grid point per time step on a Sun Ultra-1 workstation, and the maximum memory required was about 0.5 kb per grid point. As seen in Figs. 7 and 8, the numerical scheme replicates the exact scalar field very well, preserving the shape of the bell. The error field $C_{\mathrm{exact}} - C_{\mathrm{computed}}$ is almost symmetric, indicating that the scheme has good phase characteristics. The oscillations corresponding to the regions of steep gradients in the solution are localized and confined to those regions. The maximum local error observed in these regions is about −0.21% at the margin of the bell. This means that the numerical cone has widened by a very small amount, which reveals that the scheme suffers from some numerical diffusion.

The error in the peak, defined as $(C_{\mathrm{computed,max}} - C_{\mathrm{exact,max}})/C_{\mathrm{exact,max}}$, was about 0.015% at the end of the simulation, which is very small and reconfirms that numerical diffusion is very low. At the end of the simulation, the scheme lost about 0.001% mass (see Table 1). Considering the errors owing to roundoff and those introduced while inverting the mass matrix, one can claim that this method has good mass conservation performance. Numerical damping will cause loss of energy (see Table 1) even if mass is conserved. Here, the numerical scheme suffered from less than 0.01% energy loss at the end of this advection process.

This simulation was carried out for up to 100 complete revolutions of the cone, and good results were obtained. For example, the error field obtained with the present characteristic Galerkin scheme after 100 revolutions of the cosine bell over an 80×80 mesh is compared in Fig. 9 with the corresponding error field obtained with the characteristic Galerkin method using piecewise exact integration (Priestley [36]). Priestley's results were obtained from a simulation using exact trajectories, i.e. excluding the error due to approximation of the characteristic lines, and at the same Courant number as in our case. From this figure

Table 1
Global conservation measures after one revolution

Formula	Global conservation laws	Computed/exact
$\int_\Omega c\,\mathrm{d}\Omega$	Mass conservation	0.999990
$\int_\Omega c^2\,\mathrm{d}\Omega$	Energy conservation	0.999900

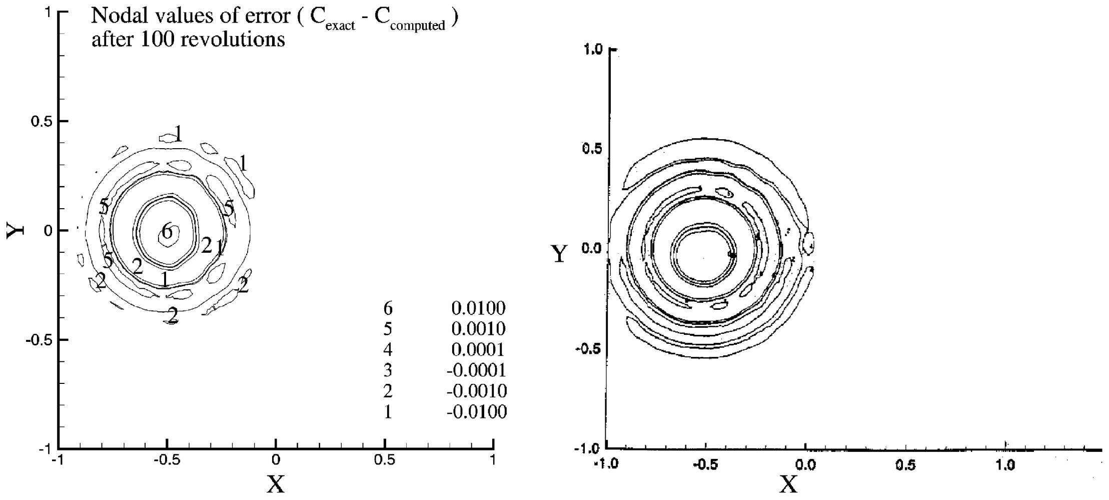

Fig. 9. Local values of error after 100 revolutions over an 80×80 grid with time steps of $\Delta t = 0.025$ obtained with seven-point Gaussian quadrature and RK4 trajectory approximation (left) and with piecewise exact integration and exact trajectories (right). Right panel is taken from [36]. Both simulations were performed on an 80×80 grid. The contour levels are the same in both plots.

Table 2
A comparison of errors exhibited by the present quadrature-based method and those by the piecewise-exact method

	Piecewise exact projection	Quadrature projection
Maximum error	0.0025	0.0014
Error in peak	0.005	0.002
L_2-norm of error	0.0018	0.0024

it can be seen that despite predictions to the contrary in the literature, the present scheme does not suffer from instability problems.

It is evident in Fig. 9 that the error field in the present quadrature-based method is more localized than that obtained with the piecewise exact integration characteristic Galerkin scheme. The methods are further compared in Table 2 which gives a quantitative measure of the error exhibited by these two methods at the end of a one-revolution simulation. The maximum error after one revolution with the present characteristic Galerkin scheme was about half of that reported with piecewise integration characteristic Galerkin scheme. The error in the peak after one full revolution in the present scheme was slightly less than half of that in the piecewise integration scheme, while the L_2-norm of error with the present scheme was 50% more than that with the piecewise exact integration characteristic Galerkin scheme [36]. Taking all factors into account, we conclude that the piecewise exact integration scheme and the present scheme demonstrate very similar error performance. However, the piecewise exact method is more costly than the quadrature projection method. More importantly, as previously noted, it is not easily extended to 3-D.

4.3. Typical three-dimensional results

Representative results are shown in Figs. 10 and 11 for a $64 \times 64 \times 64$ grid with an average grid size of $h = 0.03$, and a time step of $\Delta t = 0.025$, which required 40 time steps for one complete rotation of the "cosine ball" around the domain. The average Courant number on the domain was about 2.5. The

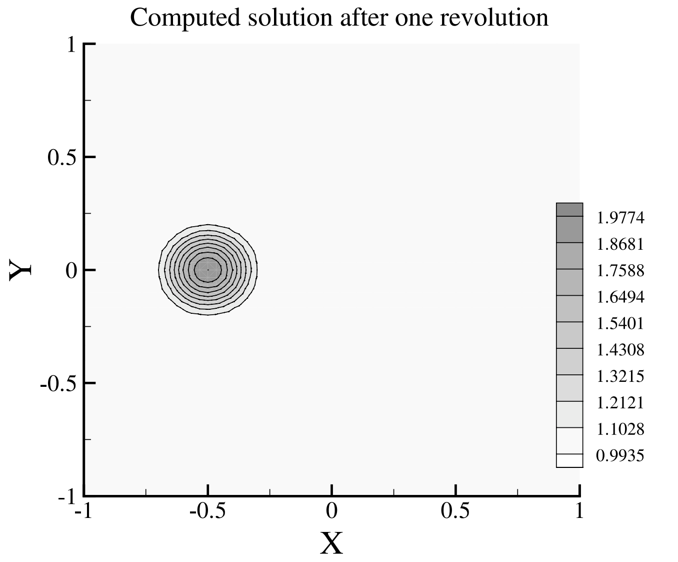

Fig. 10. Horizontal slice (on $Z = 0$ plane) of the computed scalar field after one complete revolution (3-D test case) on the $64 \times 64 \times 64$ mesh in the presence of Dirichlet boundary conditions. The time step size was $\Delta t = 0.025$.

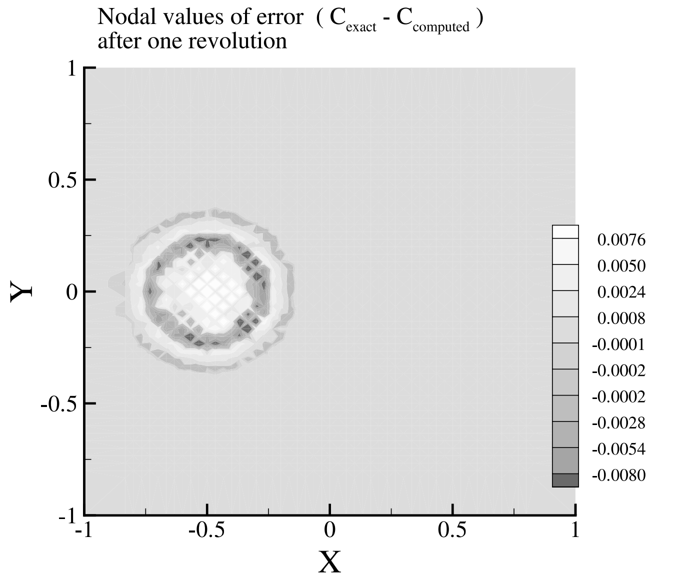

Fig. 11. Horizontal slice (on $Z = 0$ plane) of the error field after one complete revolution (3-D test case) on the $64 \times 64 \times 64$ mesh in the presence of Dirichlet boundary conditions. The time step size was $\Delta t = 0.025$.

trajectories were integrated using the fourth-order Runge–Kutta method. An eleven-point Gaussian quadrature was found to be appropriate for the RHS projection in 3-D. The CPU time for this configuration was about 0.05 s per grid point per time step, and the maximum memory required was about 1.6 kb per grid point.

The 3-D results were qualitatively similar to the 2-D results. Quantitatively, after one revolution the maximum local error in the $Z = 0$ plane was less than 0.8%, the error in the peak was about 0.8%, the lost mass was about 0.001%, and the energy loss was about 0.004%. These results indicate that the performance of the scheme is maintained in three dimensions.

5. Summary and conclusions

A 3-D characteristic Galerkin scheme for solving the linear advection equation was developed and tested. The scheme uses Gaussian quadrature for projecting the information from the Lagrangian grid onto the Eulerian grid. A variable time step technique was devised and implemented to treat the characteristic lines that cross the domain inlet boundaries before finding their foot at the previous time level. Two efficient algorithms were devised for searching within the unstructured finite element mesh to find the foot of the characteristic. In order to characterize the scheme, numerical tests were performed on a 2-D implementation of the process. Testing the 2-D and 3-D versions of the scheme against benchmark solutions showed encouraging results. The scheme exhibited good phase, accuracy and stability behaviour at large time steps.

The method is efficient if implemented wisely. Using Gaussian quadrature points for L_2-projection of the numerical information from the Lagrangian (departure) grid at the previous time step onto the Eulerian grid at the current time step makes the scheme easily and efficiently extendible to three dimensions.

The quadrature projection method that we use in this work is similar to the technique proposed in the context of the FV-ELLAM method [5,19,20], using structured 2-D and 3-D finite-volume cells. The FV-ELLAM methods define the test (weighting) functions as unity over the volume during the advection time step to satisfy the homogenous form of the adjoint of the advection equation for all $\vec{X}$ and t. One difference between our approach and the FV-ELLAM schemes, relates to the treatment of boundary conditions. Traditionally, characteristic-based algorithms are applied to open boundary problems (e.g. in groundwater applications) or to problems with periodic boundary conditions (like meteorological problems on a sphere). Because of uncertainty as to how to handle integration points that backtrack into inflow boundaries, the authors of the FV-ELLAM method use forward tracking [19,20]. As demonstrated in this work, the characteristic-based schemes can be easily applied in problems with closed boundaries by stopping the characteristic lines where they cross the inlet boundaries. In addition, our technique has been proven to be effective for simulations in complex geometries using fully unstructured grids [24,26,27], and is therefore a viable approach for treatment of advective-dominated transport problems.

Elemental searching (i.e. searching for an element that contains a certain point, namely the foot of a characteristic) is another feature that has made the characteristic-based methods unpopular with unstructured grids. This work shows that if the searching is done wisely, this procedure can be carried out efficiently even with large Courant numbers.

Although the scheme is in theory stable for all Courant numbers, in practice we found that a Courant number of about 3 is the best compromise between cost and accuracy. This limit is, of course, much less restrictive than that imposed by Eulerian schemes. However, the error performance of the scheme (Fig. 6) has several limitations. Most critically, care must be exercised when modeling an unsteady, real-world problem, where there typically exists a spectrum of velocity magnitudes and grid sizes over the computational domain. In such a case, one must obtain, as a preprocessing step, an estimate of how the ratio of $\vec{V}/h$ varies on the domain and choose a suitable time step Δt so that the Courant number lies within an acceptable range for the elements in the computational domain.

Overall, the present characteristic Galerkin algorithm is a viable method for the solution of advection problems using 3-D unstructured grids. In a subsequent paper [25], we show how this scheme can be incorporated into a more general solver for the convection–diffusion equation in 3-D. We have further demonstrated the viability of this boundary condition treatment through application of the present algorithm in the simulation of highly advection-dominated mass transport in an anatomically realistic human right coronary artery [26], and in physiologically relevant axisymmetric and asymmetric arterial stenosis [24,27].

Acknowledgements

M. Kaazempur-Mofrad gratefully acknowledges fruitful discussions with Drs. Andrew Priestley, Bill Morton and Olivier Pironneau. This work was supported by NSERC Collaborative Project grant CPG0163599 and an NSERC Steacie Fellowship (CRE).

References

[1] J.M. Augenbaum, A Lagrangian method for shallow water equations based on Voronoi mesh—one dimensional results, J. Comp. Phys. 53 (2) (1984) 240–265.
[2] A.M. Baptista, A.A. Adams, P. Gresho, Benchmarks for the transport equation: the convection–diffusion forum and beyond, in: Lynch, Davies (Eds.), Quantitative Skill Assessment for Coastal Ocean Models, AGU Coastal and Estuarine Studies, vol. 47, 1995, pp. 241–268.
[3] J. Benque, G. Labadie, J. Ronat, A new finite element method for the Navier–Stokes equations coupled with a temperature equation, in: Proceedings of the 4th International Conference on Finite Element Methods in Flow Problems, 1982, pp. 295–301.
[4] P. Binning, M. Celia, A finite-volume Eulerian–Lagrangian localized method for solution of the contaminant transport equations in two-dimensional multiphase flow systems, Water Resour. Res. 32 (1996) 103–114.
[5] P. Binning, M. Celia, A three-dimensional forward particle tracking Eulerian–Lagrangian localized adjoint method for solution of the contaminant transport equation, in: Bentley et al. (Eds.), Computational Methods in Water Resources XIII, 2000, pp. 611–618.
[6] K. Boukir, Y. Maday, B. Metivet, A high order characteristics method for incompressible Navier–Stokes equations, Comp. Meth. Appl. Mech. Engrg. 116 (1994) 211–218.
[7] A.N. Brooks, T.J.R. Hughes, Stream-line upwind/Petrov–Galerkin formulation for convection dominated flows with particular emphasis on the incompressible Navier–Stokes equations, Comp. Meth. Appl. Mech. Engrg. 32 (1982) 199–259.
[8] G.C. Buscaglia, A finite element analysis of rubber coextursion using a power-law model, Int. J. Num. Meth. Engrg. 36 (1993) 2143–2156.
[9] M. Celia, Eulerian–Lagrangian localized adjoint method for contaminant transport simulations, in: Peters et al. (Eds.), Computational Methods in Water Resources X, 1994, pp. 207–216.
[10] M.A. Celia, J.S. Kindred, I. Herrera, Contaminant transport and biodegradation: I. A numerical model for reactive transport in porous media, Water Resour. Res. 25 (1989) 1141–1148.
[11] A.J. Chorin, J.E. Mardsen, A Mathematical Introduction to Fluid Mechanics, Springer, NY, 1973.
[12] R. Courant, K. Friedrichs, H. Lewy, On partial differential equations of mathematical physics, IBM J. 11 (1967) 215–234.
[13] G.R. Cowper, Gaussian quadrature formulas for triangles, Int. J. Num. Meth. Engrg. 7 (3) (1972) 405–408.
[14] J. Donea, A Taylor–Galerkin method for convective transport problems, Int. J. Num. Meth. Engrg. 20 (1984) 101–119.
[15] J. Donea, S. Giuliani, H. Laval, L. Quartapelle, Time accurate solution of advection–diffusion problems by finite elements, Comp. Meth. Appl. Mech. Engrg. 45 (1985) 123–145.
[16] J. Donea, L. Quartapelle, V. Selmin, An analysis of time discretization in the finite element solution of hyperbolic problems, J. Comput. Phys. 70 (1987) 463–499.
[17] G.A.A.V. Haagh, F.N. Van de Vosse, Simulation of three-dimensional polymer mould filling process using a pseudo-concentration method, Int. J. Num. Meth. Fluids 28 (1998) 1355–1369.
[18] R. Healy, T. Russell, A finite-volume Eulerian–Lagrangian localized adjoint method for solution of the advection–diffusion equation, Water Resour. Res. 29 (1993) 2399–2413.
[19] R. Healy, T. Russell, Solution of the advection–dispersion equation in two-dimensions by a finite-volume Eulerian–Lagrangian localized adjoint method, Adv. Water Resour. 21 (1998) 11–26.
[20] C. Heberton, T. Russell, L. Konikow, G. Hornberger, Three-dimensional finite-volume ELLAM implementation, in: Bentley et al. (Eds.), Computational Methods in Water Resources XIII, 2000, pp. 603–610.
[21] T.J.R. Hughes, I.P. Franca, G.M. Hulbert, A new finite element formulation for computational fluid dynamics: VIII. The Galerkin/least-squares method for advective–diffusive equations, Comp. Meth. Appl. Mech. Engrg. 73 (1989) 173–189.

[22] T.J.R. Hughes, I.P. Franca, M. Mallet, A new finite element formulation for computational fluid dynamics: VI. Convergence analysis of the generalized SUPG formulation for linear time-dependent multidimensional advection–diffusion systems, Comp. Meth. Appl. Mech. Engrg. 63 (1987) 97–112.

[23] C. Johnson, A new approach to algorithms for convection problems which are based on exact transport + projection, Comp. Meth. Appl. Mech. Engrg. 100 (1992) 45–62.

[24] M.R. Kaazempur-Mofrad. A characteristic/finite element algorithm for 3-D unsteady advection-dominated transport phenomena using unstructured grids, Ph.D. thesis, Department of Mechanical and Industrial Engineering, Univesity of Toronto, 1999.

[25] M.R. Kaazempur-Mofrad, P.D. Minev, C. Ross Ethier. A characteristic/finite element algorithm for 3-D unsteady advection-dominated transport phenomena using unstructured grids. Comp. Meth. Appl. Mech Engrg, submitted for publication.

[26] M.R. Kaazempur-Mofrad, C.R. Ethier, Mass transport in an anatomically realistic human right coronary artery, Ann. Biomed. Engrg. 29 (2001) 121–127.

[27] M.R. Kaazempur-Mofrad, J.G. Myers, C.R. Ethier. Mass transport in axisymmetric and asymmetric arterial stenoses, in preparation.

[28] D.W. Kelly, S. Nakazawa, O.C. Zienkiewicz, J.C. Heinrich, A note of upwinding and anisotropic balancing dissipation in finite element approximation to convective diffusion problems, Int. J. Num. Meth. Engrg. 15 (1980) 1705–1711.

[29] D.N. Ku, D.P. Giddens, K.Z. Christopher, S. Glagov, Pulsatile flow and atherosclerosis in human cartoid bifurcation, Atherosclerosis 5 (1985) 293–302.

[30] D.E. Metcalfe, G.J. Farquhar, Modeling gas migration through unsaturated soils from waste disposal sites, Water Air Soil Pollut. 32 (1987) 247–259.

[31] K.W. Morton, A. Parrott, Generalized Galerkin methods for hyperbolic equations, J. Comp. Phys. 36 (1980) 249–270.

[32] K.W. Morton, A. Priestley, E. Suli, Stability of the Lagrange–Galerkin method with non-exact integration, Math. Model. Numer. Anal. 22 (1988) 625–653.

[33] A. Oliveira, A. Baptista, A comparison of integration and interpolation Eulerian–Lagrangian methods, Int. J. Numer. Meth. Fluids 116 (1995) 183–204.

[34] O. Pironneau, On the transport–diffusion algorithm and its applications to the Navier–Stokes equations, Numer. Math. 38 (1982) 309–332.

[35] O. Pironneau, Finite Element Methods for Fluids, John Wiley and Sons Ltd., Chichester, UK, 1989.

[36] A. Priestley, Exact projections and the Lagrange–Galerkin method: a realistic alternative to quadrature, J. Comp. Phys. 112 (1994) 316–333.

[37] F. Shakib, T.J.R. Hughes, A new finite element formulation for computational fluid dynamics: IX. Fourier analysis of space–time Galerkin/least-squares algorithms, Comp. Meth. Appl. Mech. Engrg. 87 (1991) 35–58.

[38] B.E. Sleep, J.F. Sykes, Computational simulation of groundwater contamination by organic compounds, Water Resour. Res. 29 (1993) 1697–1708.

[39] P.K. Smolarkiewicz, J.A. Pudykiewicz, A class of semi-Lagrangian approximations for fluids, J. Atmos. Sci. 49 (1987) 2082–2096.

[40] A. Staniforth, J. Cote, Semi-Lagrangian integration schemes for atmospheric models—a review, Mon. Weather Rev. 119 (1991) 2206–2223.

[41] A.S. Usmani, J.T. Cross, R.W. Lewis, A finite element model for the simulation of mould filling in metal casting and the associated heat transfer, Int. J. Num. Meth. Engrg. 35 (1992) 787–806.