Available online at www.sciencedirect.com
SCIENCE DIRECT®
Comput. Methods Appl. Mech. Engrg. 192 (2003) 1281–1298
Computer methods in applied mechanics and engineering
www.elsevier.com/locate/cma

A characteristic/finite element algorithm for time-dependent 3-D advection-dominated transport using unstructured grids

M.R. Kaazempur-Mofrad [a], P.D. Minev [b], C.R. Ethier [a,*]

[a] *Department of Mechanical and Industrial Engineering, University of Toronto, 5 King's College Road, Toronto, Ont., Canada M5S 3G8*

[b] *Department of Mathematical Sciences, University of Alberta, Edmonton, Alta., Canada T6G 2G1*

Received 29 May 2001; received in revised form 21 June 2002; accepted 13 November 2002

Abstract

An algorithm based on operator splitting has been successfully implemented for solving unsteady, advection-dominated transport problems in 3-D. Specifically, the general operator-integration-factor splitting method of Maday et al. is applied to the unsteady advection–diffusion equation with source/sink terms. The algorithm incorporates a 3-D characteristic Galerkin scheme to treat advection, and a standard Galerkin treatment of the diffusion and source/sink terms. Up to third-order operator splitting was implemented and validated against several analytical solutions.

The algorithm showed the expected error behaviour and good performance in modeling advection-dominated transport problems. The practical utility and effectiveness of the proposed numerical scheme was further demonstrated by solving the Graetz–Nusselt problem, i.e. high Peclet number mass/heat transport in a fully developed pipe flow.

Keywords: Advection-dominated diffusion; Operator splitting; Finite element method; Method of characteristics; 3-D; Unstructured grid

1. Introduction

The dimensionless form of the governing equation for advective–diffusive transport is

$$\frac{\partial c}{\partial t} + \vec{V} \cdot \nabla c = \frac{1}{Pe} \nabla^2 c + f, \tag{1}$$

where $c(\vec{x}, t)$ is the transported scalar, and $\vec{V}(\vec{x}, t)$ is a known velocity field, $Pe = V_0 L_0 / \mathscr{D}$ is the Peclet number ($\mathscr{D}$ is the diffusion coefficient, and V_0 and L_0 are characteristic velocity and length scales), and f represents source or sink terms.

Numerical solution of Eq. (1) can be challenging, due in part to the mixed hyperbolic–parabolic nature of the equation. A successful numerical scheme must efficiently treat both the advection (hyperbolic) and

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

0045-7825/03/$ - see front matter
doi:10.1016/S0045-7825(02)00627-8

unsteady diffusion (parabolic) terms. For diffusion dominated cases, most conventional numerical schemes (e.g. the standard Galerkin finite element approach) behave reasonably well, and fairly reliable results are achieved. However, when advective effects dominate, the results of such schemes are unsatisfactory, for the reasons summarized in [20]. Advection-dominated transport phenomena occur in many fields of physics and engineering. Examples are heat transfer processes in rubber extrusion, plastic casting and mould filling [5,15,33], fluid flow and contaminant transport in porous media [7,12,30], atmospheric and ocean modeling [23,28,32], and cardiovascular mass transport processes [19,21].

The conventional Galerkin finite element method of spatial discretization, along with classical time discretization methods, fail to produce satisfactory solutions to 'near-hyperbolic' advection-dominated transport problems. Spurious oscillations are generated, particularly in regions of high gradients in the solution, which can corrupt the whole solution. These oscillations may in general 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. Numerous approaches have been proposed to preclude oscillations in advection-dominated transport problems, regardless of mesh or time step refinement. These schemes can be divided into three broad groups: Eulerian, Lagrangian, and Eulerian–Lagrangian. These three classes of schemes are reviewed in [20] in the context of finite element methodology. Briefly, 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 [4,17,22], Galerkin/Least-squares [16,29], and Taylor–Galerkin [9–11] 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 [8].

In the Lagrangian schemes (see for example [1]) the computational mesh moves along the characteristics (fluid trajectories). In practice, stretching and shearing of the original fluid particles distort the mesh after a few time steps, and hence these methods are rarely used.

In Eulerian–Lagrangian or characteristic methods, the advection step is treated using a Lagrangian tracking algorithm along characteristic lines while keeping the computational grid fixed. Points from within the Eulerian grid are tracked backward along the characteristics over the time step, thereby forming a Lagrangian virtual grid. Numerical information from the previous time level is projected from the background Eulerian grid onto the Lagrangian grid. A significant advantage of the characteristic 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. Several versions of the characteristic method have been proposed in the literature (see for example [3,18,26,31]). In the finite element context, the characteristic schemes combine the classical method of characteristics with a Galerkin finite element approximation.

An attractive approach to solution of the advection–diffusion equation across a wide range of Pe is to employ an operator splitting approach, which decouples the advection and diffusion operators by first advecting the scalar field (usually explicitly) and then diffusing it (usually implicitly). In this approach, the two operators of different mathematical nature are split and each is treated by a numerical scheme that best mimics the underlying physics of the respective operator.

The objective of the present work was to develop an efficient algorithm for solving the 3-D advection–diffusion transport equation using fully unstructured grids. In particular, the algorithm was required to work well for advection-dominated transport phenomena, i.e. with extremely high Pe values. To achieve this goal, we used the high-order operator-integration-factor splitting method of Maday et al. [24]. The advective terms in the unsteady advection–diffusion equation were treated by the characteristic Galerkin scheme briefly described in Section 2.2 (a detailed description and characterization of this scheme can be

found in [20]). The present paper shows the practical utility and effectiveness of the proposed numerical schemes.

2. Methods

2.1. Operator splitting for advection–diffusion equation

The advection–diffusion equation can be rewritten as:

$$\frac{\partial c}{\partial t} = \mathbf{D}c + \mathbf{C}c + f, \tag{2}$$

where

$$\mathbf{D} = \nabla \cdot \left(\frac{1}{Pe} \nabla \right)$$

is the diffusion operator, and $\mathbf{C} = -\vec{V} \cdot \nabla$ is the advection operator. Following the operator-integration-factor splitting method of Maday et al. [24] (see also [3]), Eq. (2) is written in terms of an integration factor in $\mathbf{C}$

$$\frac{\partial}{\partial t}(\mathcal{Q}_C^{(t^*,t)} c(t)) = \mathcal{Q}_C^{(t^*,t)}(\mathbf{D}c + f), \tag{3}$$

where t^* is an arbitrary, but fixed, time. The integration factor $\mathcal{Q}_C^{(t^*,t)}$ is defined by

$$\frac{\partial}{\partial t}(\mathcal{Q}_C^{(t^*,t)}) = -\mathcal{Q}_C^{(t^*,t)}\mathbf{C} \tag{4}$$

and

$$\mathcal{Q}_C^{(t^*,t^*)} = \mathbf{I}, \tag{5}$$

where $\mathbf{I}$ is the identity operator. Eq. (3) can be viewed as a pure diffusion problem for the new variable $\mathcal{Q}_C^{(t^*,t)} c$, to be integrated in time by a suitable method for the diffusion operator $\mathbf{D}$. If $t^* = t^{n+1}$, application of a kth order Gear's backward differencing scheme with a time step $\Delta t = t^{n+1} - t^n$ to Eq. (3) gives the following semi-discrete system

$$\frac{\beta_0 c^{n+1} - \sum_{i=1}^{k} \beta_i \mathcal{Q}_C^{(t^{n+1},t^{n+1-i})} c^{n+1-i}}{\Delta t} = \mathbf{D}c^{n+1} + f^{n+1}. \tag{6}$$

The coefficients β_i of the various schemes are listed in [6].

Integration of the diffusion equation, Eq. (3), requires evaluating the integration terms of the form $\mathcal{Q}_C^{(t^{n+1},t^{n+1-i})} c^{n+1-i}$, $(i = 1, 2, \ldots, k)$. To avoid explicit construction of $\mathcal{Q}_C^{(t^{n+1},t^{n+1-i})}$, an auxiliary variable $\tilde{c}(s)$ is introduced that satisfies the following initial value problem

$$\begin{cases} \dfrac{\partial \tilde{c}(s)}{\partial s} = \mathbf{C}\tilde{c}(s), \quad 0 < s < i\Delta t \\ \tilde{c}(0) = c^{n+1-i}. \end{cases} \tag{7}$$

It then follows that

$$\mathcal{Q}_{\mathrm{C}}^{(t^{n+1},t^{n+1-i})} c^{n+1-i} = \tilde{c}(i\Delta t). \tag{8}$$

Eq. (7) accounts for the pure advection step, and can be solved using a suitable scheme with a time step Δs (which can be different from Δt). It is noteworthy that the integration factor $\mathcal{Q}_{\mathrm{C}}^{(t^{n+1},t^{n+1-i})}$ is never

constructed explicitly. Rather, its role is played through the solution of the corresponding advection problem (Eq. (7)). In the present work we solve Eq. (7) using the characteristic Galerkin method outlined in Section 2.2 (further details of this scheme and its chacaterization can be found in [20]). The diffusion step (Eq. (3)) was solved using a standard Galerkin method, as briefly described in Section 2.3.

2.2. Treatment of the advection step

Consider the linear advection equation

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

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

$$c(\vec{x}, t = 0) = c_0(\vec{x}), \tag{10}$$

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

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

$$\frac{\mathrm{d}\vec{X}}{\mathrm{d}t}(\vec{x}, s; t) = \vec{V}(\vec{X}(\vec{x}, s; t), t) \tag{11}$$

with

$$\vec{X}(\vec{x}, s; s) = \vec{x}. \tag{12}$$

Once the characteristics are known from Eq. (11), the solution to the advection equation is

$$c(\vec{X}(., t; t + \tau), t + \tau) = c(., t), \tag{13}$$

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

$$\vec{x} = \vec{X}(\vec{y}, t^{n+1}; t^n) \tag{14}$$

and

$$\vec{y} = \vec{X}(\vec{x}, t^n; t^{n+1}). \tag{15}$$

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).

When discretized in time, Eq. (13) results in

$$c^{n+1}(\vec{y}) = c^n(\vec{x}). \tag{16}$$

In line with the Galerkin formulation, Eq. (16) is next multiplied by a weighting function w and is integrated over the spatial domain (at time t^{n+1}) [2,25–27]. The weighting functions w are chosen as equal to the Eulerian basis functions (at time t^{n+1}):

$$\int_\Omega c^{n+1}(\vec{y}) w \,\mathrm{d}\Omega = \int_\Omega c^n(\vec{x}) w \,\mathrm{d}\Omega. \tag{17}$$

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

$$\sum_j C_j^{n+1} \int_\Omega \phi_j(\vec{y}) \phi_i(\vec{y}) \,\mathrm{d}\vec{y} = \sum_j C_j^n \int_\Omega \phi_j(\vec{x}) \phi_i(\vec{y}) \,\mathrm{d}\vec{y}. \tag{18}$$

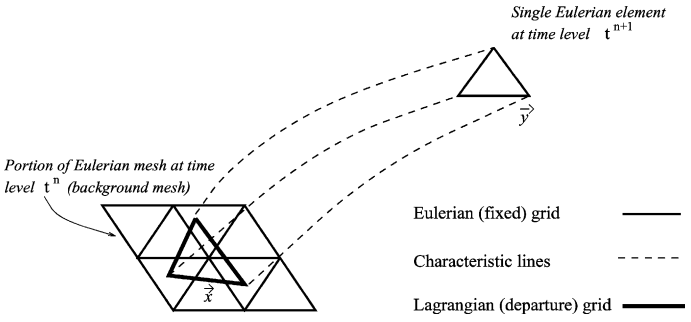

Fig. 1. Schematic of Eulerian and Lagrangian grids for a characteristic Galerkin method.

The left hand side of Eq. (18) 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. (18) becomes

$$\sum_{\text{ele}} \sum_{j=1}^{d} C_j^{n+1} \int_{\Delta} \phi_j(\vec{y}) \phi_i(\vec{y}) \, \mathrm{d}\vec{y} = \sum_{\text{ele}} \sum_{j=1}^{d} C_j^{n+1} \int_{\Delta} L_j L_i \, \mathrm{d}\vec{y} \quad i = 1, \ldots, d, \tag{19}$$

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 computation of the right hand side of Eq. (18) (''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.

In the present algorithm, 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:

$$\int_{\Omega} c^n(\vec{x}) \phi_i(\vec{y}) \, \mathrm{d}\vec{y} = \sum_{\text{ele}} \sum_{k} w_k c^n(\vec{x}_k) \phi_i(\vec{y}_k). \tag{20}$$

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.

Further specification of the characteristic Galerkin scheme, as well its characterization and practical implementation details, can be found in [20].

2.3. Treatment of the diffusion step

Consider the unsteady diffusion equation in non-dimensional form

$$\frac{\partial c}{\partial t} = \frac{1}{Pe} \nabla^2 c + f, \tag{21}$$

where the variables are as previously described. A standard Galerkin scheme is suitable for the solution of this equation, and yields the following semi-discrete form:

$$[M]\frac{\mathrm{d}\{c\}}{\mathrm{d}t} = [D]\{c\} + \{F\}, \tag{22}$$

where $[M]$ and $[D]$ are the mass and diffusion matrices defined as $M_{ij} = \int_\Omega \phi_i \phi_j \,\mathrm{d}\Omega$ and

$$D_{ij} = -\int_\Omega \left(\nabla\phi_i \cdot \frac{1}{Pe}\nabla\right)\phi_j \,\mathrm{d}\Omega,$$

respectively, $\{F\}$ is the source/sink vector defined as $F_i = \int_\Omega \phi_i f \,\mathrm{d}\Omega$, and ϕ_i is an Eulerian basis function. This scheme was implemented using tetrahedral finite elements with trilinear (volume coordinate) basis functions. The semi-discrete system was temporally discretized using Gear's implicit backward differencing method of order k, giving the following fully discrete system of equations:

$$(\beta_0[M] - \Delta t[D])\{c\}^{n+1} = \sum_{i=1}^{k} \beta_i [M]\{c\}^{n+1-i} + \{F\}^{n+1}. \tag{23}$$

3. Tests and results

The advection and diffusion solvers were first tested separately. The results of benchmarking the advection scheme are reported in the companion paper [20]. One of the test cases performed to benchmark the diffusion solver is described in Section 3.1. The overall operator splitting advection–diffusion solver was then tested against several benchmark problems as described in Section 3.2.

3.1. Test case for diffusion: algebraic solution testing

Simple analytic solutions to Eq. (21) were employed to test the diffusion solver. The analytic solutions were constructed using first-order polynomials in space, i.e. lying within the functional space spanned by the shape functions, ϕ_i. This allowed the temporal error performance of the numerical scheme to be examined even on a relatively coarse grid. The source term, $f(\vec{x}, t)$, was chosen to be $f(\vec{x}, t) = (p+1)t^p(x+y+z)$, in which case an exact solution is $c_{\text{exact}}(\vec{x}, t) = t^{p+1}(x+y+z)$, where f balances $\partial c/\partial t$. In order to test a time marching scheme of order k, it is necessary that p in the above solution satisfy $p \geqslant k$. In this study, we satisfied this requirement by simply choosing $p = k$.

The algorithm was tested against these analytical solutions on a very coarse grid as follows. The computational domain $\Omega = [-1, 1]^3$ was meshed with a $5 \times 5 \times 5$ grid. This grid was formed by uniformly distributing five points on each edge of the cube, and dividing the domain into smaller cubes based on these points. Each subcube was then subdivided into five tetrahedral elements. Dirichlet boundary conditions, based on the analytical solution, were imposed on the domain boundaries. Runs were performed with different time steps to a final time of $t = 2$.

The L_∞ norm of the error at the final time $t = 2$ is shown as a function of time step size in Fig. 2 for first-, second- and third-order versions of the above algorithm. Linear least-squares fitting of the log–log transformed data in Fig. 2 gave average slopes of 1.0, 1.9 and 2.8 for the first, second, and third-order schemes. This is close to the expected convergence rates, confirming the temporal performance of our implementation of the Gear's backward differencing scheme.

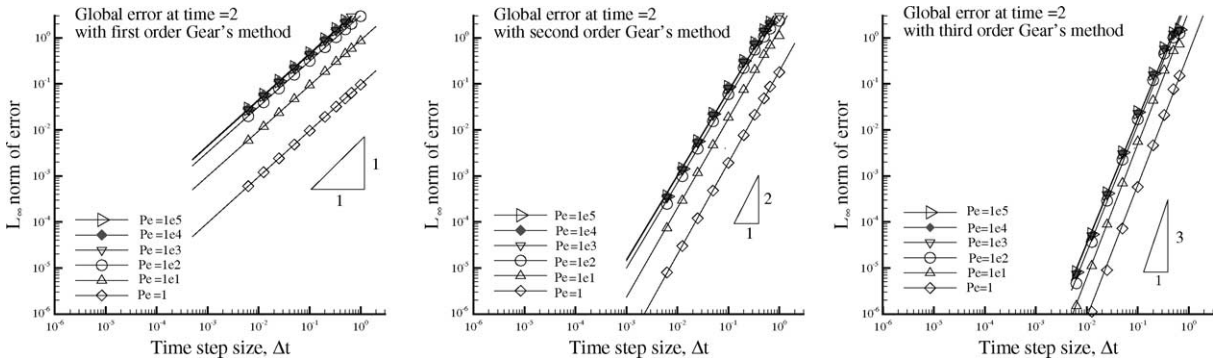

Fig. 2. Global error at time $t = 2$ versus Δt with (from left to right) first-, second- and third-order Gear's backward time stepping methods. The slope of error plots in log–log scale is indicative of the order of the scheme (test case: algebraic solution testing for diffusion).

As an aside, we note that the global error plotted in Fig. 2 shows a dependence on Pe for values of $Pe \leqslant 10^2$. This dependence is more explicitly demonstrated in Fig. 3, where the error is plotted against Pe for various time step sizes, Δt. At first sight, this behaviour may seem erroneous, since the analytic solutions are linear in $\vec{x}$, and as such $(1/Pe)\nabla^2 c$ (which is the only term involving Pe) should vanish identically and not make any contribution to the scheme's performance. This behavior can best be understood by considering a simple 1-D problem on a uniform mesh of spacing Δx. In this case, it is straightforward to derive the equivalent differential equation [13] for the scheme, and show that for the above form of c_{exact}, the error in the numerical solution $\epsilon = c_{\text{computed}} - c_{\text{exact}}$ must satisfy:

$$\frac{\partial \epsilon}{\partial t} - \frac{1}{Pe}\frac{\partial^2 \epsilon}{\partial x^2} = -\frac{\Delta t^k}{(k+1)!}\frac{\partial^{k+1} c_{\text{exact}}}{\partial t^{k+1}} + \text{higher-order terms} \tag{24}$$

for Gear's kth order backward differencing schemes. From this equation, it is clear that ϵ will be sensitive to Pe for small to moderate values of Pe, in which case the second term will dominate the first.

From Eq. (24) it is clear that the error should scale with Δt^k, be independent of Pe for large Pe, and decrease with decreasing Pe for small Pe. This is in fact what is observed (Fig. 4), where the dependence of error on Pe for $Pe \leqslant 100$ is clearly seen.

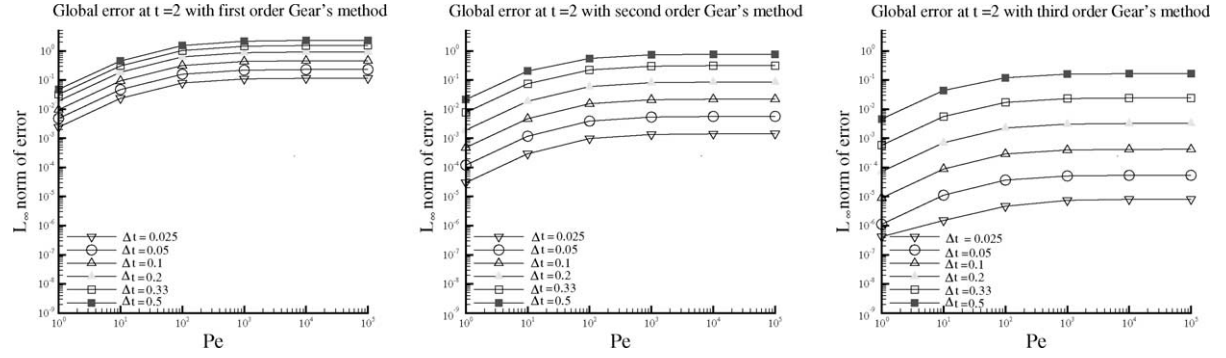

Fig. 3. Global error at time $t = 2$ versus Pe obtained with various time step sizes. The three panels represent results using (from left to right) first- through third-order Gear's backward differencing scheme (test case: algebraic solution testing for diffusion).

Fig. 4. Global error at time $t = 2$ normalized by Δt^k versus *Pe* obtained with various time step sizes. The three panels represent results using (from left to right) first- through third-order Gear's backward differencing scheme (test case: algebraic solution testing for diffusion).

3.2. Test cases for advection–diffusion

3.2.1. Test case 1: Algebraic solution testing

This test case was the same as that described in Section 3.1, except that here the velocity field was chosen $\vec{V}$ such that $\vec{V} \cdot \nabla c_{\text{exact}} = 0$. This again allowed testing of the temporal error performance of the overall advection–diffusion scheme on a relatively coarse mesh. More specifically, we chose f as above and $\vec{V} = (1/4, -1/4, 0)$, in which case the corresponding analytic solutions were $c(\vec{x}, t) = t^{p+1}(x + y + z)$. The exponent p was chosen as equal to the order of the scheme to be tested.

The schemes were tested against these analytical solutions on the same coarse mesh as used in Section 3.1. Dirichlet boundary conditions, based on the analytical solution, were imposed on the domain boundaries. Runs were performed with different time steps to a final time of $t = 2$. The L_∞ norm of the error at the final time $t = 2$ is shown as a function of time step size in Fig. 5 for these schemes.

Regression analysis on the log–log transformed data confirmed that the splitting scheme was working as expected, i.e. demonstrating consistent convergence behaviour. The dependence of global error on *Pe* seen in Section 3.1 was observed in this case as well.

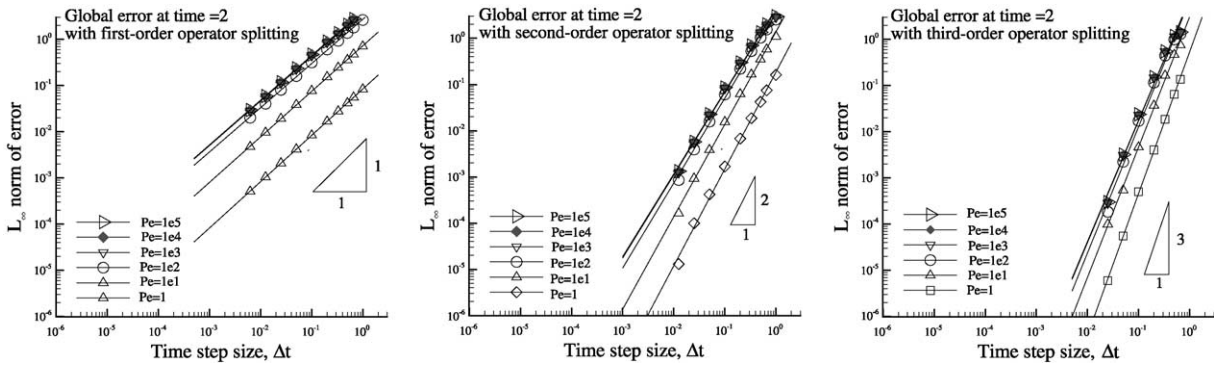

Fig. 5. Global error at time $t = 2$ versus Δt with (from left to right) first-, second- and third-order operator splitting methods (test case 1: algebraic solution testing for advection–diffusion).

3.2.2. Test case 2: Advection–diffusion of a Gaussian sphere

The overall performance of the scheme was next tested by simulating the advection and diffusion of an initially 3-D Gaussian distribution of tracer material. The velocity field $\vec{V}$ was assumed to be constant and uniform, in which case the exact solution has the form

$$c_{\text{exact}}(\vec{x},t) = \left(\frac{\sigma_0}{\sigma}\right)^3 \exp\left\{-\frac{1}{2\sigma^2}[\vec{x} - (\vec{x}_c + \vec{V}t)]^2\right\}, \tag{25}$$

where the time-dependent standard deviation is given by

$$\sigma(t) = \left(\sigma_0^2 + \frac{2t}{Pe}\right)^{1/2}. \tag{26}$$

σ_0 is the initial standard deviation of the Gaussian distribution, and $\vec{x}_c$ is the initial position of the center of the Gaussian distribution. Time-dependent Dirichlet boundary conditions, based on the analytical solution, were imposed on the domain boundaries.

The computational domain $\Omega = [-1,1]^3$ was meshed with a $30 \times 30 \times 30$ grid. This grid was formed by uniformly distributing 30 points on each edge of the cube, and dividing the domain into smaller cubes based on these points. Each subcube was then subdivided into five tetrahedral elements. We used the following parameters: $\vec{V} = (1/2, 1/2, 0)$ and $Pe = 10^6$. This corresponded to a grid Peclet number of approximately $Pe_\Delta = 3.3 \times 10^4$. The initial Gaussian sphere was centered at $\vec{x}_c = (-1/4, -1/4, 0)$ and had an initial standard deviation $\sigma_0 = 0.2$. To eliminate start-up transients in the calculation of the error field, the characteristics associated with the advective solver were allowed to back-track "out of the domain", as needed. This occurred, for example, when the third-order splitting scheme, which requires an advective update from time steps n, $n-1$ and $n-2$, was used to obtain the solution at timestep $(n+1)\Delta t$ for $n = 0$ and 1. In such cases, scalar information based on the analytic solution was assigned to the feet of the affected characteristics. This allowed all time steps in the simulation to be performed with a uniform order of splitting, which was important for these fairly short runs.

Figs. 6–8 show the computed scalar field and the error field at time $t = 1$ for the first- to third-order methods, respectively for a time step of 0.1 $\left(Cu = (|\vec{V}|\Delta t)/h = 1.1\right)$. A maximum local error of approximately 1% was observed in the $z = 0$ plane on this fairly coarse mesh and with this fairly large time step. The L_∞ norm of error at time $t = 1$ is shown in Table 1 for different values of Cu and different orders of splitting.

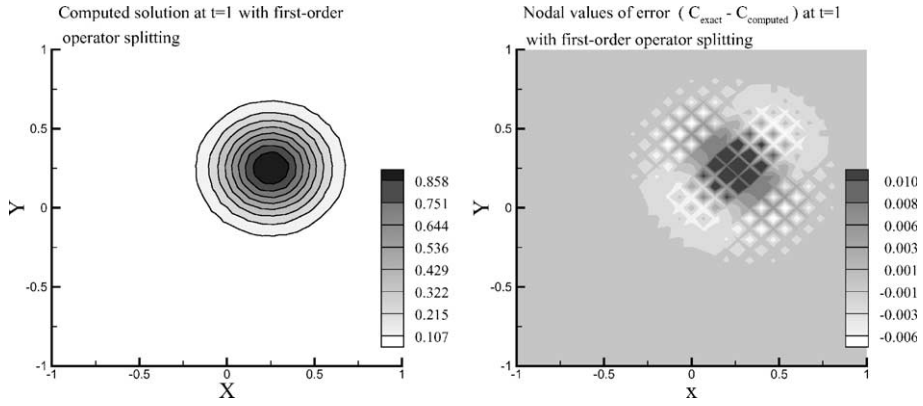

Fig. 6. Horizontal slice (on $z = 0$ plane) of the computed scalar field (left panel) and the error field (right panel) after 10 time steps of 0.1 in a $30 \times 30 \times 30$ grid with first-order operator splitting approach (test case 2: advection–diffusion of a Gaussian sphere).

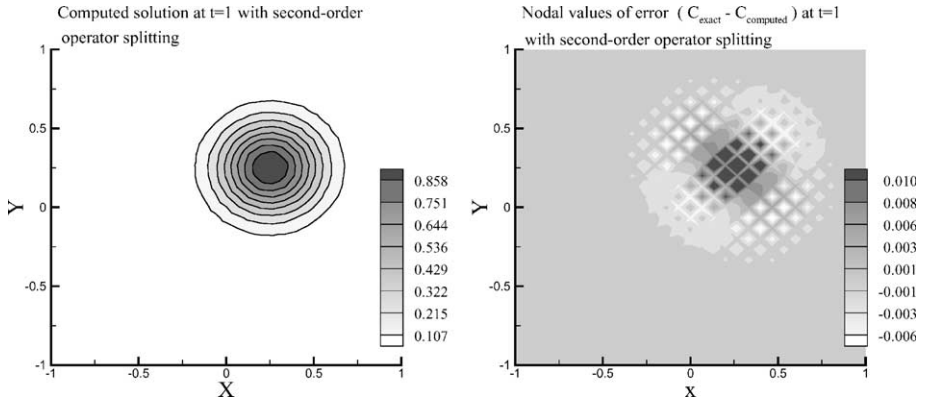

Fig. 7. Horizontal slice (on $z = 0$ plane) of the computed scalar field (left panel) and the error field (right panel) after 10 time steps of 0.1 in a $30 \times 30 \times 30$ grid with second-order operator splitting approach (test case 2: advection–diffusion of a Gaussian sphere).

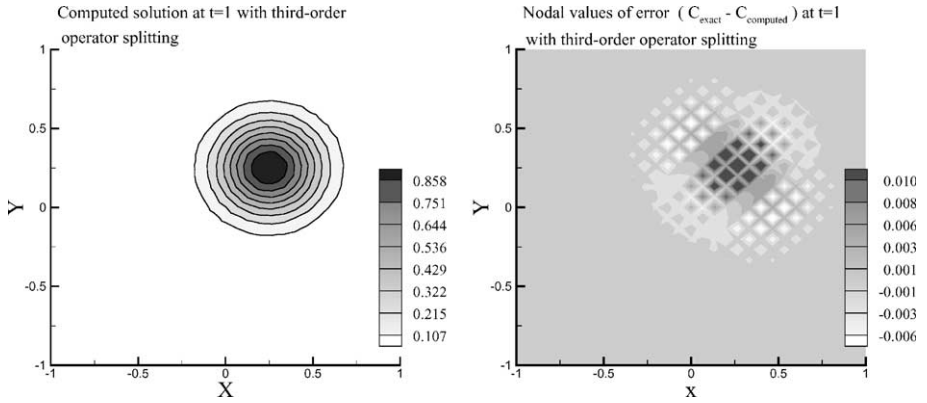

Fig. 8. Horizontal slice (on $z = 0$ plane) of the computed scalar field (left panel) and the error field (right panel) after 10 time steps of 0.1 in a $30 \times 30 \times 30$ grid with third-order operator splitting approach (test case 2: advection–diffusion of a Gaussian sphere).

Table 1
Dependence of error on Courant number and splitting order for test case 2: advection–diffusion of a Gaussian sphere

Δt	Cu	L_∞-norm of error		
		First order	Second order	Third order
0.025	0.275	0.0346	0.0479	0.0578
0.05	0.55	0.0151	0.0200	0.0255
0.1	1.1	0.0103	0.0101	0.0096
0.2	2.2	0.0086	0.0071	0.0060
0.33	3.7	0.0130	0.0084	0.0074

Several points are noteworthy. First, there is only a weak dependence of error on splitting order. This implies that other error sources are important. Two such sources exist: The error introduced by the advection solver (see [20]) and a spatial discretization error. Note that for this test case the equivalent differential equation contains an extra term, not present in Eq. (24), that is proportional to Δx^2. This means

that the spatial discretization is first-order accurate. For problems whose solution is characterized by regions of steep gradient, such lower-order methods are in general competitive with high-order methods [6,14]. For example, the most widely used schemes for shock capturing problems are first-order accurate. The higher the order of the spatial discretization for non-smooth solutions, the worse the oscillations in the approximation [14].

The spatial discretization error on this coarse mesh is reflected in pattern of square-like islands seen in the error field (Figs. 6–8). In order to confirm that the spatial discretization error decreases as the mesh is refined, a grid refinement study was performed using three sets of grids. The coarsest mesh ($20 \times 20 \times 20$ grid) consisted of 40,000 tetrahedral elements with 9261 corner nodes. The mesh with intermediate refinement ($30 \times 30 \times 30$ grid) had 135,000 elements with 29,791 nodes. The finest mesh ($40 \times 40 \times 40$ grid) contained 320,000 elements with 68,921 nodes. As shown in Fig. 9, the mesh refinement monotonically reduces the error between the numerical and exact solutions, hence confirming that the scheme is consistent.

The second noteworthy point is that error performance is not monotonically dependent on time step, Δt. This is understood by noting that the splitting error increases monotonically with Δt while the error in the advective solver varies non-monotonically with Δt over the Cu range of interest.

The third noteworthy point is that for small Cu, the total error actually increases with splitting order. This is due to a combination of two effects. First, for such small Cu, the error due to the advective solver dominates. Second, each time step in the kth order splitting scheme requires k independent advective solves (see Eq. (6)), thereby providing an opportunity for error "accumulation" when the advective error is large. From a practical viewpoint, this error behaviour complicates the application of this scheme to real-world problems, 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 pre-processing 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 (see [20]).

3.2.3. Test case 3: Graetz–Nusselt solution

The algorithm was further tested using the problem of a developing heat/mass transfer boundary layer in a hydrodynamically fully developed laminar flow in a cylindrical tube (the Graetz–Nusselt problem). This problem, whose analytical solution is available by neglecting axial diffusion, is a commonly used benchmark problem for transport problems in blood flow where the Peclet numbers are in the range of 10^6 or higher. Here, we assumed a $Pe = 2 \times 10^7$ based on the tube diameter. The boundary conditions were: $c = 1$ at the inlet, $c = 0$ on the walls, and $\nabla c \cdot \vec{n} = 0$ at the outlet.

3.2.3.1. Mesh construction. In highly advection-dominated transport problems, the boundary layer is very thin throughout the whole domain, except in regions of flow separation. To resolve such thin boundary layers, while keeping the mesh size within practical limits, stretched (high aspect ratio) elements are needed. Creating highly stretched tetrahedral elements for 3-D unstructured grids is not trivial. Most commercial mesh generation software packages do not allow this, since stretched elements are considered to be "bad quality" elements. Elsewhere [19] we describe a general strategy for generating stretched tetrahedral elements in boundary layer regions. Here, we simplified the mesh generation process by taking advantage of the regular domain geometry and the fact that the temperature/concentration profile is uniform outside the thin heat/mass transfer boundary layer. We therefore created a mesh that only filled a portion of the cylindrical tube. Specifically, we chose the computational domain to be a 15° sector of an annulus, with inner and outer boundaries at $r/R = 0.9$ and 1.0, respectively. The axial length of the computational domain was set to be $5R$. We imposed a homogeneous Neumann boundary condition on the inner boundary of the mesh.

To create the mesh, we first generated a structured quadrilateral mesh in a $[-1, 1]^3$ cube and then divided each quadrilateral into five tetrahedra. We then mapped the mesh onto the computational domain using a

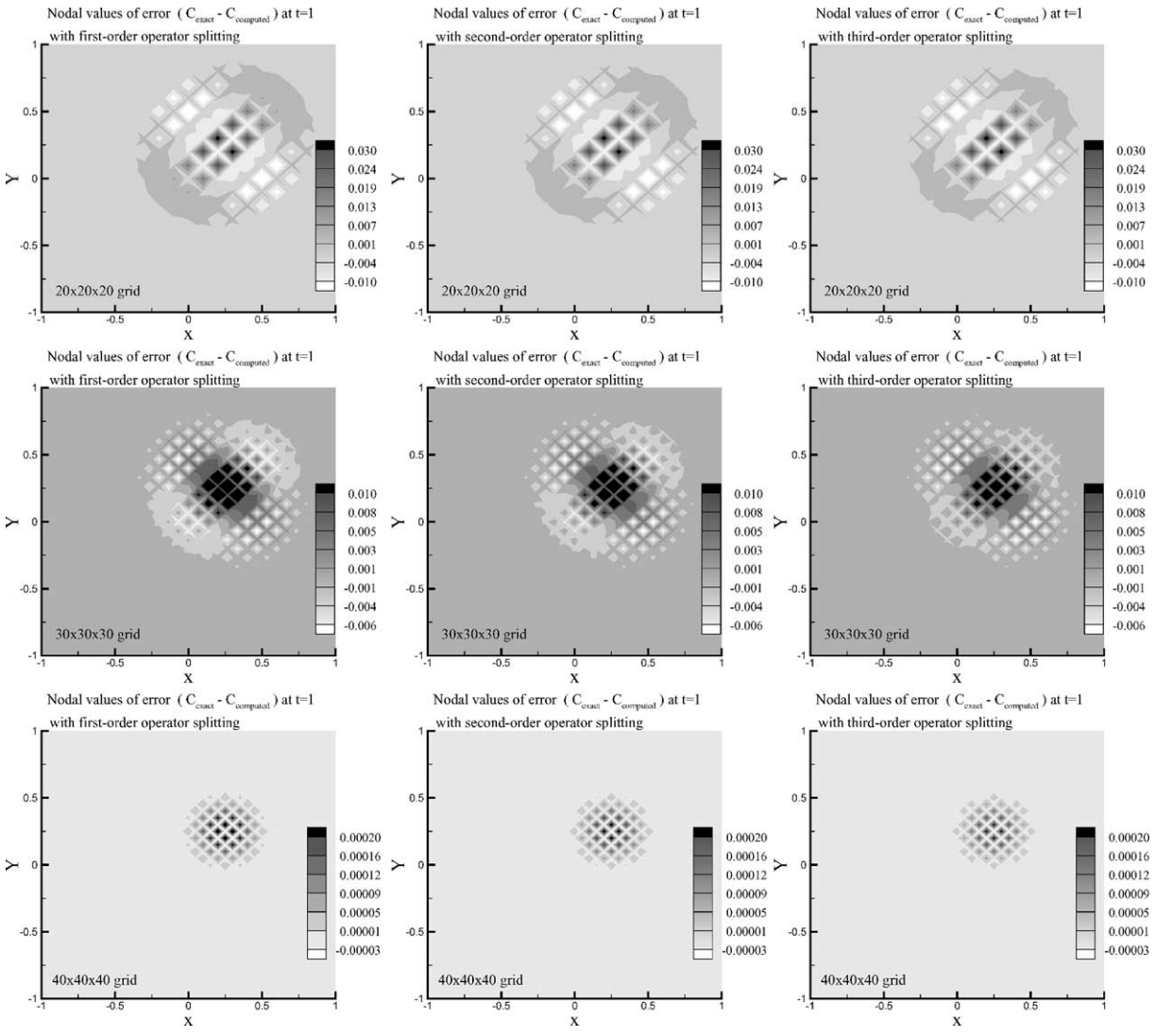

Fig. 9. Grid refinment study: Horizontal slice (on $z = 0$ plane) of the error field after 10 time steps of 0.1 in coarse $20 \times 20 \times 20$ grid (top row), intermediate $30 \times 30 \times 30$ grid (middle row), and fine $40 \times 40 \times 40$ grid (bottom row). Data with different orders of operator splitting, namely first order (left panels), second order (middle panels) and third order (right panels) are presented for each level of mesh refinement (test case 2: advection–diffusion of a Gaussian sphere).

linear one-to-one mapping. The mapping from a point P with coordinates (x, y, z) (where $x, y, z \in [-1, 1]$) to a point P' with coordinates (x', y', z') can be described as $x' = r \sin \alpha$, $y' = r \cos \alpha$ and $z' = (L/2)(z + 1)$. Here, $\alpha = (x/a)(\phi/2)$, and $r = R_c + y$, where the variables are defined in Fig. 10. L is the desired axial length of the annulus sector. Thus, for the present case we specified: $L = 5$, $R_1 = 0.9$, $R_2 = 1.0$, and $\phi = \pi/12$.

The nodes in the structured grid were arranged in the cube such that when they were mapped onto the annulus, the off-wall spacing was 0.0008, i.e. 1/2500th of the tube diameter. 17 nodes were distributed across the radial thickness of the annulus, 15 nodes along the arc, and 31 nodes along the axial distance (see Fig. 11). This resulted in a mesh with 37,800 tetrahedral elements (8,835 nodes).

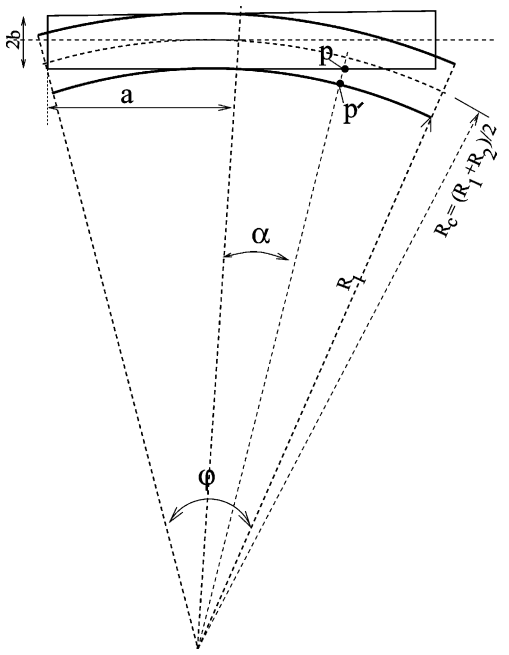

Fig. 10. A schematic of the mapping used to create the mesh in test case 3 (Graetz–Nusselt problem). Solid lines represent the box in which the structured mesh was generated. Heavy lines represent the 15-degree annulus sector used as the computational domain.

3.2.3.2. Convergence to steady state. The steady state solution was achieved by marching in time with time steps of $\Delta t = 0.5$ until the quantity $\|\Delta c\|_{L_\infty}$ reached 10^{-6} in single-precision computations, where Δc is the difference in the the solution between two consecutive steps (Fig. 12). A time step of $\Delta t = 0.5$ corresponds to a maximum Courant number over the computational domain of approximately $Cu = 14$, calculated elementwise and based on the minimum internodal spacing in each element and the local velocity at that node. The convergence history for $\|\Delta c\|_{L_\infty}$ is monotonic, but characterized by differing convergence rates throughout the calculation. This can be attributed to a complex interplay between advection of numerical information from the inlet throughout the computational domain and diffusion of the boundary information from the walls.

Fig. 13 compares the normalized concentration gradient at the wall, i.e. the Sherwood number, for the numerical and analytical solutions, versus the axial position (normalized by the tube radius, R). As the figure demonstrates, the scheme accurately predicts solution gradients for this highly advection-dominated situation. The discrepancy at the inlet ($Z/R \simeq 0$) is due to the singularity in the analytical solution at the inlet.

A "step" pattern is seen in the numerically computed curve. This pattern is an artifact of the coarseness of the grid in the axial direction and the fact that simulations use linear finite elements. Because the mesh is created by subdivision of a structured grid, each tetrahedral element has a twin which is aligned in the opposite direction. Although the solution gradient is the same in each pair, their centroidal positions are offset in the z direction. Fig. 13 shows the Sherwood number exhibited by each boundary element versus the axial position of the centroidal point of the element. As a result, a step pattern is created which is more pronounced near the inlet, i.e. at regions of high axial gradients in the solution. This step pattern implies that a more axially refined mesh is needed at areas where high axial gradients are expected in the solution. To further demonstrate this, a refined mesh was used which had 86,400 tetrahedral elements with 19,475

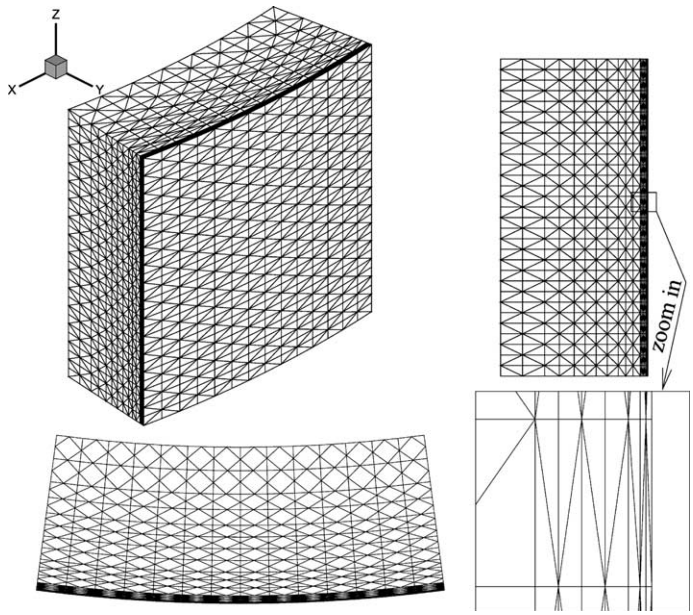

Fig. 11. A close-up of the mesh used for the test case 3 (Graetz–Nusselt problem). A 3-D view is shown at top left, a $z =$ constant slice is shown at bottom, and a $x =$ constant slice is shown at right.

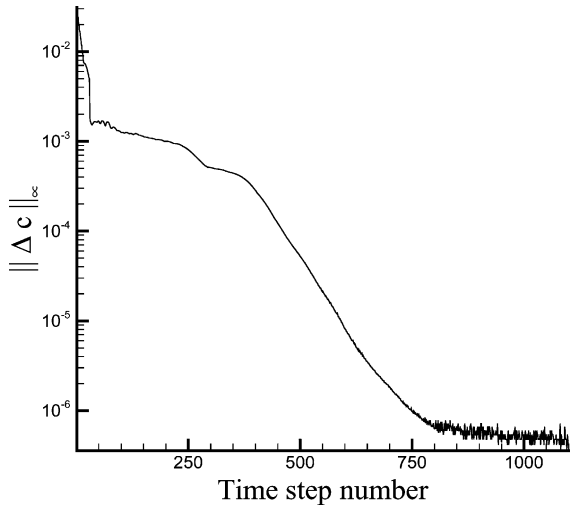

Fig. 12. Convergence history to the steady state solution with a time step of $\Delta t = 0.5$ (test case 3: Graetz–Nusselt solution).

corner nodes. The Sherwood number distribution calculated using the refined mesh (see Fig. 14) confirms that the step pattern decreases to a great extent and that the numerical scheme converges to the exact solution as the mesh is refined. It must be remarked that due to the physical singularity inherent in the problem (i.e. the thickness of the transport boundary layer is near zero at the inlet region), a full capture of the boundary layer at the entrance region is impossible.

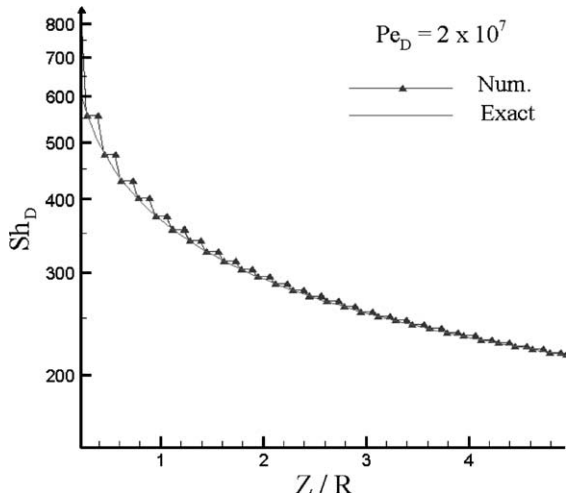

Fig. 13. Dimensionless gradient of the solution (Sherwood number Sh_D) on the wall versus axial position (Z/R) for test case 3: Graetz–Nusselt problem.

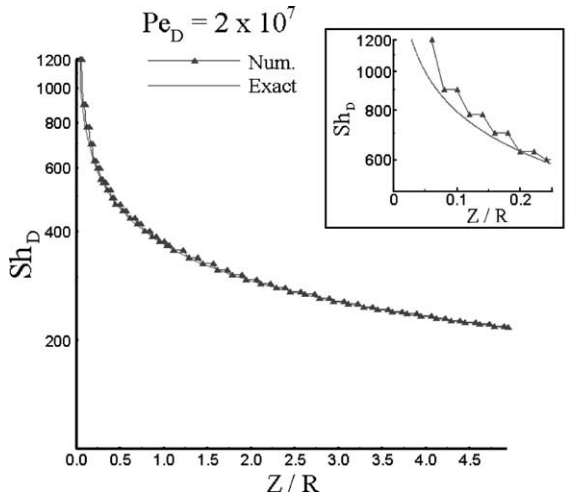

Fig. 14. Dimensionless gradient of the solution (Sherwood number Sh_D) on the wall versus axial position (Z/R) for test case 3: Graetz–Nusselt problem with the refined mesh. Comparison with Fig. 13 reveals that the step pattern is suppressed to a great extent as the mesh is refined. The inset shows a closer look at the inlet region, where the near zero boundary layer thickness prevents full capture of the solution gradient.

3.3. CPU profile of the scheme

The CPU breakdown for the present operator splitting algorithm with the characteristic Galerkin scheme for advection and the standard Galerkin scheme for diffusion is outlined in Table 2 for the first- to third-order splitting methods when applied to the Graetz–Nusselt Problem. The algorithm was implemented in C programming language, and run on a serial machine.

Table 2
CPU breakdown of the present algorithm, listed in descending order of CPU time

Task	CPU breakdown		
	First order	Second order	Third order
Conjugate-gradient solver	36%	50%	58%
Elemental searching for departure points	33%	29%	25%
Pre- and post-processing	17%	10%	7%
Interpolation of nodal values	9%	7%	6%
Trajectory approximation	5%	4%	4%

See text for detailed description of each item.

The CPU profile in Table 2 corresponds to 100 time steps of $\Delta t = 1$. The solver was a conjugate-gradient solver with Jacobi pre-conditioning used for solving the linear systems arising in the advective update as well as Eq. (6). The elemental search represents the location of departure points in the Eulerian grid, as required for the advective solver [20]. The pre-processing category includes the matrix and vector formations, data structure and memory allocations, setting up of the connectivity tables (i.e. node-to-node, element-to-element, and element-to-node tables), calculating the elemental volumes, determining wall (boundary) elements and their corresponding outward unit normal vector, etc. The post-processing task includes calculating the error norms, computing the solution gradients, outputting the results, etc. Interpolation of nodal values and trajectory approximation (including the variable time step method for trajectories that leave the domain near the inflow boundaries) is required for the advective solver.

Note that the conjugate-gradient solver's share of CPU cost increases markedly with the splitting order. This is due to the fact that splitting schemes of order 2 and above require the solution of additional linear systems for the quantities $\tilde{c}$. Note also that the above CPU breakdown is for a simulation in which the fluid trajectories are not stored (see [20]). In steady state simulations, fluid trajectory information can be computed and stored in a pre-processing step, and then used in subsequent time steps. This strategy effectively removes as much as a third of the total CPU cost, since the elemental search must be performed only once.

4. Summary and conclusions

An algorithm based on operator splitting was successfully developed for solving unsteady, advection-dominated transport problems. The algorithm incorporates a 3-D characteristic Galerkin scheme to treat advection terms, and a standard Galerkin treatment of the diffusion terms. Up to third-order operator splitting was implemented and validated against several analytical solutions. The algorithm showed consistent error performance by achieving its expected nominal convergence rate for all test cases. Overall, the present 3-D characteristic/FEM scheme exhibited good performance in modeling advection-dominated transport problems. We have further demonstrated the viability of this scheme through application of the present algorithm in the simulation of highly advection-dominated mass transport in an anatomically realistic human right coronary artery [21], and in physiologically relevant axisymmetric and asymmetric arterial stenosis [19].

The numerical tests and examples in the present work illustrate the effectiveness of the operator splitting method for the advection-dominated transport problems. In this method, the two mathematically and physically distinct operators of diffusion and advection are decoupled and each is treated by a numerical scheme that best mimics the respective underlying physics. This approach is particularly appealing in advection-dominated transport problems with extremely high *Pe* values. The semi-Lagrangian nature of the advection treatment allows stable and accurate solutions to these problems at fairly large CFL numbers, as compared to fully Eulerian methods.

Acknowledgements

M. Kaazempur-Mofrad gratefully acknowledges fruitful discussions with Drs. Andrew Priestley, Bill Morton and Olivier Pironneau. This work was supported by an NSERC Collaborative Project Grant CPG0163599 and by 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. Comput. Phys. 53 (2) (1984) 240–265.
[2] J. Benque, G. Labadie, J. Ronat, A new finite element method for the Navier–Stokes equations coupled with a temperature equation, in: Proc. 4th. Int. Conf. on Finite Element Methods in Flow Problems, 1982, pp. 295–301.
[3] K. Boukir, Y. Maday, B. Metivet, A high order characteristics method for incompressible Navier–Stokes equations, Comput. Methods Appl. Mech. Engrg. 116 (1994) 211–218.
[4] 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, Comput. Methods Appl. Mech. Engrg. 32 (1992) 199–259.
[5] G.C. Buscaglia, A finite element analysis of rubber coextursion using a power-law model, Int. J. Numer. Methods Engrg 36 (1993) 2143–2156.
[6] M. Celia, W. Gray, Numerical Methods for Differential Equations, Prentice Hall, New Jersey, 1992.
[7] 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.
[8] R. Courant, K. Friedrichs, H. Lewy, On partial differential equations of mathematical physics, IBM J. 11 (1967) 215–234.
[9] J. Donea, A Taylor–Galerkin method for convective transport problems, Int. J. Numer. Methods Engrg. 20 (1984) 101–119.
[10] J. Donea, S. Giuliani, H. Laval, L. Quartapelle, Time accurate solution of advection–diffusion problems by finite elements, Comput. Methods Appl. Mech. Engrg. 45 (1985) 123–145.
[11] 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.
[12] R.E. Ewing, Simulation of multiphase flows in porous media, Transport Porous Media 6 (1991) 479–499.
[13] C.A.J. Fletcher, in: Computational Techniques for Fluid Dynamics, vol. 1, Springer Verlag, 1990.
[14] P.M. Gresho, R.L. Sani, in: Incompressible Flow and the Finite Element Method, vol. 1, John Wiley and Sons Ltd., Chichester, NY, 1998.
[15] 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. Numer. Methods Fluids 28 (1998) 1355–1369.
[16] 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, Comput. Methods Appl. Mech. Engrg. 73 (1989) 173–189.
[17] 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, Comput. Methods Appl. Mech. Engrg. 63 (1987) 97–112.
[18] C. Johnson, A new approach to algorithms for convection problems which are based on exact transport + projection, Comput. Methods Appl. Mech. Engrg. 100 (1992) 45–62.
[19] M.R. Kaazempur-Mofrad, A Characteristic/Finite Element Algorithm for 3-D Unsteady Advection-dominated Transport Phenomena Using Unstructured Grids, PhD thesis, Department of Mechanical and Industrial Engineering, University of Toronto, 1999.
[20] M.R. Kaazempur-Mofrad, C.R. Ethier, An efficient characteristic Galerkin scheme for the advection equation in 3-D, Comput. Methods Appl. Mech. Engrg. 191 (46) (2002) 5345–5363.
[21] M.R. Kaazempur-Mofrad, C.R. Ethier, Mass transport in an anatomically realistic human right coronary artery, Ann. Biomed. Engrg. 29 (2001) 121–127.
[22] D.W. Kelly, S. Nakazawa, O.C. Zienkiewicz, J.C. Heinrich, A note of upwinding and anisotropic balancing dissipation in finite element approximation to onvective diffusion problems, Int. J. Numer. Methods Engrg 15 (1980) 1705–1711.
[23] D.Y. LeRoux, C.A. Lin, A. Staniforth, A semi-implicit semi-Lagrangian finite-element shallow-water ocean model, Monthly Weather Rev. 128 (5) (2000) 1384–1401.
[24] Y. Maday, A. Patera, E. Rønqusit, An operator-integration-factor splitting method for time-dependent problems: application to incompressible fluid flow, J. Sci. Comput. 5 (1990) 263–292.
[25] 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.

[26] O. Pironneau, On the transport-diffusion algorithm and its applications to the Navier–Stokes equations, Numer. Math. 38 (1982) 309–332.
[27] A. Priestley, Exact projections and the Lagrange–Galerkin method: a realistic alternative to quadrature, J. Comput. Phys. 112 (1994) 316–333.
[28] R.B. Rood, Numerical advection algorithms and their role in atmospheric transport and chemistry models, Rev. Geophys. 25 (1) (1987) 71–100.
[29] 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, Comput. Methods Appl. Mech. Engrg. 87 (1991) 35–58.
[30] B.E. Sleep, J.F. Sykes, Computational simulation of groundwater contamination by organic compounds, Water Resour. Res. 29 (1993) 1697–1708.
[31] P.K. Smolarkiewicz, J.A. Pudykiewicz, A class of semi-Lagrangian approximations for fluids, J. Atmospheric Sci. 49 (1987) 2082–2096.
[32] A. Staniforth, J. Cote, Semi-Lagrangian integration schemes for atmospheric models—a review, Monthly Weather Rev. 119 (1991) 2206–2223.
[33] 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. Numer. Methods Engrg 35 (1992) 787–806.