INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS
Int. J. Numer. Meth. Fluids 2005; **47**:121–137
Published online 5 November 2004 in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/fld.799

Semi-analytical method for departure point determination

Nick Martin*,† and Steven M. Gorelick

Department of Geological and Environmental Sciences, Stanford University, Braun Hall Bldg. 320, 450 Serra Mall, Stanford, CA 94305-2115, U.S.A.

SUMMARY

A new method for departure point determination on Cartesian grids, the semi-analytical upwind path line tracing (SUT) method, is presented and compared to two typical departure point determination methods used in semi-Lagrangian advection schemes, the Euler method and the four-step Runge–Kutta method. Rigorous comparisons of the three methods were conducted for a severely curving hypothetical flow field and for advective transport in the rotation of a Gaussian concentration hill. The SUT method was shown to have equivalent accuracy to the Runge–Kutta method but with significantly improved computational efficiency. Depending on the case being simulated, the SUT method provides either far greater or equivalent computational efficiency and more certain accuracy than the Euler method.

KEY WORDS: semi-Lagrangian advection; departure point determination

1. INTRODUCTION

The ability to numerically simulate regional-scale fluid flow allows accurate weather forecasting and ocean modelling. The capacity to finely resolve complex fluid flow enables enhanced understanding of compressible and incompressible fluid dynamics. One of the main concerns in both numerical weather prediction and computational fluid dynamics is computational efficiency while maintaining accuracy. For example, maximum permissible model time step used to represent advection has been limited by stability concerns as well as model accuracy [1].

The traditional stability constraint on advection imposed by the Courant–Friedrichs–Lewy (CFL) restriction, $\mathrm{CFL} < 1$, limits the acceptable time step in Eulerian frameworks,

$$\mathrm{CFL} = u_i \frac{\Delta t}{\Delta x_i} \tag{1}$$

*Correspondence to: N. Martin, Department of Geological and Environmental Sciences, Stanford University, Braun Hall Bldg. 320, 450 Serra Mall, Stanford, CA 94305-2115, U.S.A.
†E-mail: nick.martin@stanfordalumni.org

Contract/grant sponsor: National Science Foundation; contract/grant number: EAR 0207177

Received 26 January 2004
Revised 1 September 2004

where u_i is the velocity component in the x_i direction, Δx_i is computational volume length in the x_i direction, and Δt is the model time step. An Eulerian model describes fluid flow from fixed points in space and provides computational simplicity because the locations of these points are known in advance. However, Lagrangian representations, which describe fluid flow from the vantage point of a moving fluid particle, are not limited by the CFL restriction on model time step. To circumvent the CFL restriction, numerical models combine Lagrangian and Eulerian techniques. A variety of slightly different combinations of Eulerian and Lagrangian representations exist.

In this paper, we are concerned with methods that employ Lagrangian representations to determine values for advected quantities at regularly spaced model grid points. We refer to these methods as semi-Lagrangian schemes [1] even though Eulerian–Lagrangian methods [2, 3] and the modified methods of characteristics [4] have been previously employed to describe similar algorithms.

1.1. Semi-Lagrangian schemes

In semi-Lagrangian schemes, advection processes are modelled by travelling with the fluid across an underlying Eulerian model grid. Any conservative quantity, like momentum or scalar concentration, will have a constant value along a trajectory of flow [5]. Equation (2) provides a description of the advection of passive concentration, C, and Equation (3) describes the self-advection of momentum in a constant density fluid. In Equations (2) and (3), t is time, and u_i is velocity in the x_i direction. The Lagrangian component of a semi-Lagrangian scheme involves tracing fluid trajectories. The Eulerian component interpolates between the known advection-related quantities (e.g. C or u) on the underlying model grid to provide values at points along a fluid trajectory.

$$\frac{\mathrm{D}C}{\mathrm{D}t} = \frac{\partial C}{\partial t} + u \cdot \nabla C = 0 \tag{2}$$

$$\frac{\mathrm{D}u}{\mathrm{D}t} = \frac{\partial u}{\partial t} + u \cdot \nabla u = 0 \tag{3}$$

A common method of semi-Lagrangian solution for Equation (2) or (3) is to solve the material derivative component, $\mathrm{D}C/\mathrm{D}t$ or $\mathrm{D}u/\mathrm{D}t$, for C or for u by following the path lines of fluid particles. This formulation takes advantage of the constant value of the advected quantity along the trajectory of flow and solves for the spatial displacement, given the underlying velocity field, that occurs in the time interval. As an example of this approach, Equation (4) provides an approximate integration of Equation (2) along a fluid trajectory, where x_M is the trajectory end point, and α is the displacement distance across the trajectory. Equation (4) gives a semi-Lagrangian representation for passive concentration, and the similar semi-Lagrangian relationship for self-advection of momentum would replace C with u.

$$\frac{C(x_M, t + \Delta t) - C(x_M - \alpha, t)}{\Delta t} = 0 \tag{4}$$

The application of a semi-Lagrangian path line tracing scheme is a two-step process [1, 3]. In the first step, a discrete set of particles that arrives at a regular set of grid points is traced backward over the current time step to find the particles' departure points [5] (e.g. $(x_M - \alpha)$

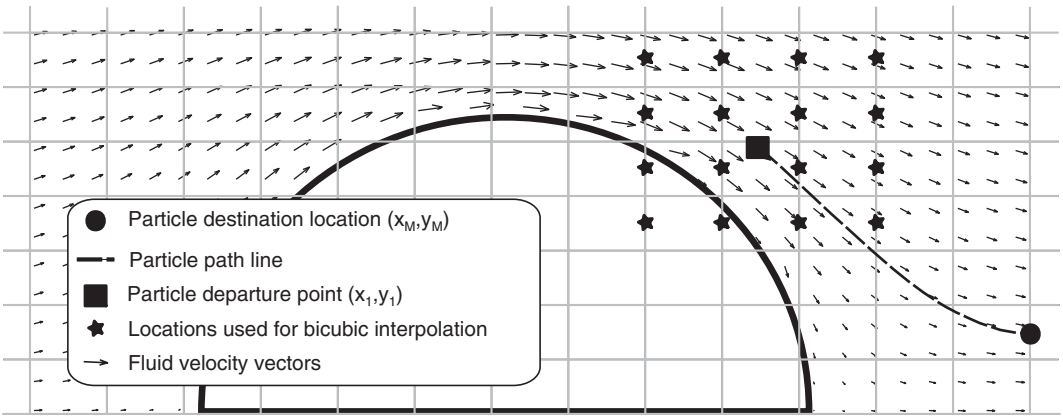

Figure 1. Two-step, semi-Lagrangian representation of advection for uniform flow around the top-half of a submerged cylinder. The first step is path line tracing from the particle destination point, (x_M, y_M), to the particle departure point, (x_1, y_1). Given the particle departure point, the value for the advective quantity is obtained by interpolation from the surrounding grid points. The stars provide the locations employed in bicubic Lagrange polynomial interpolation. The uniform flow around the top-half of a submerged cylinder scenario is examined in greater detail in Section 3.1.

in Equation (4)). The particles destination points are the initial, regular set of grid points. In terms of methods of characteristics, the characteristic ending at each grid point of interest is followed to the foot location [4] in the first step. The second step is the determination of the value of the advected quantity at the departure point, (e.g. $C(x_M - \alpha, t)$ in Equation (4)). Because the departure point will likely not coincide with a model grid point or other location where the advected quantity is defined, the value is interpolated from the surrounding model-grid points or is integrated across the surrounding elements.

Figure 1 displays this two-step process on a two-dimensional Cartesian grid; the first step is tracing the particle path line from the destination point (x_M, y_M) to the departure point (x_1, y_1). The second step, in Figure 1, is the interpolation of the quantity of interest from the surrounding known values from the computational grid. The overall accuracy of this type of semi-Lagrangian method is sensitive to the accuracy of both the departure point location and to the interpolation or integration method [5, 6]. The focus of the remainder of this paper is on step 1, departure point determination.

1.2. Departure point calculation methods

The first step in a semi-Lagrangian scheme is to determine departure points for particles with destination points located at a set of grid points. The departure points could be particle locations after a complete model time step, Δt, or the mean value theorem could be applied to permit the use of the midpoint of the particle trajectory as the average, advected value [7]. Finding the location of either the midpoint or the departure point involves tracing the path lines of the particles, originally located at the regular set of grid points, across all or part of the current time interval.

Although semi-Lagrangian schemes are not subject to the CFL < 1 restriction on model time step, Δt, the grid-based velocity values employed for path line tracing must correspond to the grid location of the particle. As the particle moves across the simulation domain the velocity values employed for path line tracing must be from the local neighborhood of the current particle location. As a result, the CFL < 1 criterion stipulates the duration of the partial time step, τ, that can be employed for path line tracing. Each partial time step corresponds to the travel distance of the particle so that the particle will not travel beyond the local area without updating the velocity values used to calculate the particle path line. For departure point determination when Δt provides CFL > 1, departure points are located by moving back a number of partial time steps, M, so that the CFL < 1 criterion is not exceeded for each partial time step, τ, and so that the sum of partial time steps is equal to the model time step.

$$\Delta t = M \times \tau \tag{5}$$

Departure point determination methods are characterized by the number of steps necessary to calculate particle positions at the end of a partial time step and by the time level, superscript n, of the velocity values employed to calculate the position (i.e. the method is implicit, $n+1$, or explicit, n). In the remainder of the equations in this paper, the superscript on velocity variables refers to the model time step. Superscript n is the current time level; $n+1$ denotes the current time level plus Δt. The subscript on velocity variables denotes spatial location.

Groundwater and surface flow semi-Lagrangian schemes generally employ explicit, linear, multi-step path line tracing methods or Runge–Kutta methods for departure point location. One-step, or Euler method, algorithms have been employed for path line tracing in semi-Lagrangian scalar transport and in the self-advection of momentum [2, 8, 9].

Modified one-step methods that employ the four corners of a computational volume have been employed for self-advection of momentum [10, 11]. Two-step linear methods have been employed for scalar advection [12]. Four-step Runge–Kutta methods have also been used in both scalar transport [13] and for the self-advection of momentum [14, 15]. Even fifth-order Runge–Kutta–Fehlberg integration has been employed in fluid dynamics calculations [4].

In numerical weather prediction, semi-Lagrangian schemes generally use implicit, linear, multi-step methods for path line tracing. Implicit methods improve the stability of the scheme but require an interpolation and extrapolation of velocity values to the new time step [16] and require iteration to obtain the final solution [17]. After the extrapolation of velocity values to the unknown time $(n+1)$, linear multi-step methods are employed in an iterative manner to obtain the implicit solution for particle location. To extrapolate the velocity values, both two [18, 19] and three time level schemes have been adopted [1, 19]. An implicit midpoint algorithm is commonly used for departure point determination in atmospheric applications [7, 19, 20] and the departure point is not the end point but the midpoint of the trajectory. Second order, implicit Runge–Kutta methods have also been used [7, 21]. In terms of the influence of departure point location on accuracy, use of the trajectory endpoint rather than the midpoint is more accurate in spherical geometries in some cases [22]. Also, departure point location with trajectory calculations was found to create a small error in comparison to a situation where the departure point was known exactly [23].

2. SEMI-ANALYTICAL PATH LINE TRACING FOR DEPARTURE POINT DETERMINATION

The purpose of this paper is to introduce a new path line tracing method for departure point determination in semi-Lagrangian schemes on Cartesian grids. The new method is a semi-analytic scheme obtained by modifying a forward-in-time path line tracing method employed in groundwater particle-tracking applications [24] that generally provides increased accuracy compared to linear, explicit, multi-step methods [25]. The underlying assumption of the semi-analytic method for forward-in-time particle tracking is that each directional velocity component varies linearly in its coordinate directions within each computational volume or cell. A linear variation in velocity in each direction within the volume allows the derivation of an analytical expression for the path line of a particle across a volume [24].

This semi-analytic solution, provided in Reference [24], can easily be modified to trace a particle path backward-in-time to the departure point for the first step of a semi-Lagrangian scheme. The result is the SUT method for departure point determination. Figure 2 provides the layout of the SUT scheme for tracing a path line, or tracking a particle, backward-in-time across a single computational volume in two dimensions, but the method can easily be extended to three dimensions [24, 25]. Equations (6) and (7) provide the analytic solution for the path line exit point (X_e, Y_e). In Equations (6) and (7), X_p and Y_p provide the particle current location on the volume boundary; U_{xp} and V_{yp} provide the x- and y-directions velocity components at (X_p, Y_p) which are calculated assuming a linear velocity variation from Equation (8); x_2 provides the x-co-ordinate of the upwind x-direction volume boundary;

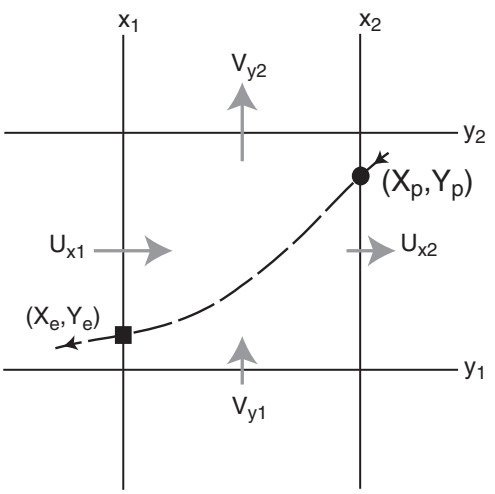

Figure 2. Schematic layout for the semi-analytical upstream path line tracing (SUT) method for backward-in-time particle tracking, (X_p, Y_p) is the particle entry point into the computational volume. (X_e, Y_e) is the calculated particle exit point from the computational volume. The downstream x-boundary of the computational volume is x_2; the upstream x-boundary is x_1. The downstream y-boundary of the computational volume is y_2; the upstream y-boundary is y_1. The downstream velocity component in the x-direction is U_{x2}, and the upstream x-direction velocity component is U_{x1}. The downstream velocity component in the y-direction is V_{y2}, and the upstream y-direction velocity component is V_{y1}.

y_2 gives the y-co-ordinate of the upwind y-direction volume boundary; U_{x2} is the velocity at the upwind x-direction boundary; V_{y2} is the velocity at the upwind y-direction boundary; A_x and A_y are the velocity gradients across the computational volume to be traversed in the x- and the y-directions, Equation (9), and τ_e is the partial time step duration necessary for a particular particle to travel across the current volume, given in Equation (10). In Equation (9), U_{x1} and V_{y1} are the velocities at the downwind x- and y-volume boundaries, respectively, and Δx and Δy are the distances across the computational volume in the x- and y-directions. Equations (6)–(10) are given in explicit form (velocity terms have superscript n). However, an implicit formulation could be conceived by employing extrapolated velocities in a manner similar to the modifications of the semi-analytical path line method employed for particle tracking in transient groundwater flow [26].

$$X_e = x_2 - \frac{1}{A_x}\left[U^n_{x_2} - \frac{U^n_{x_p}}{\exp(A_x\tau_e)}\right] \tag{6}$$

$$Y_e = y_2 - \frac{1}{A_y}\left[V^n_{y_2} - \frac{V^n_{y_p}}{\exp(A_y\tau_e)}\right] \tag{7}$$

$$U^n_{x_p} = U^n_{x_2} - A_x(x_2 - x_p), \quad V^n_{y_p} = V^n_{y_2} - A_y(y_2 - y_p) \tag{8}$$

$$A_x = \frac{U^n_{x_2} - U^n_{x_1}}{\Delta x}, \quad A_y = \frac{V^n_{y_2} - V^n_{y_1}}{\Delta y} \tag{9}$$

$$\tau_{e_{k,p}} = \min\left(\frac{1}{A_x}\ln\left[\frac{U^n_{x_p}}{U^n_{x_1}}\right], \frac{1}{A_y}\ln\left[\frac{V^n_{y_p}}{V^n_{y_1}}\right], \Delta t - \sum\tau_{e_{k,p}}\right) \tag{10}$$

Equation (10) enforces the CFL<1 restriction on particle travel distance with the semi-analytic method because it ensures that a particular particle, p, will travel completely across the local volume or will travel the distance stipulated by the model time step but will not travel past the local volume boundary. Remembering that particles are initially located at each grid point of interest for P total destination points, $p = 1, \ldots, P$, the time of traversal must be calculated for every particle travelling through each volume, for $k = 1, \ldots, K$ total volumes. When the sum of partial time steps would exceed the duration of the model time step, the final partial time step is cut short to ensure that the sum of partial time steps equals the model time step. As a result, Equation (11) gives the relation between partial time steps for a particular particle, $\tau_{e_{k,p}}$, and model time step, Δt.

$$\Delta t = \sum_{k=1}^{K} \tau_{e_{k,p}} \tag{11}$$

This semi-analytical method of SUT provides a one-step particle departure point solution that can be employed for departure point location in semi-Lagrangian schemes for both the self-advection of momentum and advective transport. This method is one-step because the particle location at the end of the partial time step in one-direction is calculated with the solution of one equation, Equation (6). In the SUT method, each particle can travel completely across a computational volume, provided that $\Delta t > \tau_e$, in each step. Because of

the semi-analytic nature of the method, SUT can provide a more accurate departure point determination than other one-step methods.

3. COMPARISON OF DEPARTURE POINT METHODS

The Runge–Kutta family of linear multi-step methods is commonly employed in groundwater, surface water, and numerical weather prediction applications for departure point determination. In this family of methods, improvement in location accuracy is obtained by increasing the number of steps. For example, the one-step Euler method is spatially first-order accurate while the four-step, classical Runge–Kutta method is spatially fourth-order accurate. To compare the SUT method to this family of methods, we examine representative end-members, the one-step Euler scheme and the four-step classical Runge–Kutta scheme, of the family.

The Euler scheme is one-step because a single calculation or step along the fluid trajectory can be taken as long as the fluid particle does not leave the local domain of influence during the step. The CFL criterion, equation (1), governs the extent of the local domain of influence. If the time interval for integration, the model time step Δt, will result in a departure point location outside of the local domain of influence, then multiple partial time steps, $s = M, M-1, \ldots, 1$, are employed to reach the departure point, (x_1, y_1), in Equations (12) and (13). Equation (14) provides the calculation for maximum partial time step duration, τ, that meets the CFL constraint for two-dimensional applications. This number of time steps could be modified to obtain the midpoint by multiplying Δt in Equation (5) by 0.5. Equation (14) presents the relationship between model time step, Δt, and partial time step, τ. In Equations (12)–(14), Δx is the model discretization length in the x-direction, Δy is the model discretization length in the y-direction, U is the x-direction velocity component, V is the y-direction velocity component, the superscript n gives an explicit velocity value, and the subscript k on the superscript n denotes bilinear interpolation from the surrounding model grid velocity values.

$$x_{s-1} = x_s - \tau U^{n_k}_{x_s, y_s} \tag{12}$$

$$y_{s-1} = y_s - \tau V^{n_k}_{x_s, y_s} \tag{13}$$

$$\tau \leqslant \min \left[\frac{\Delta x}{\max_{i,j} |U^n|}, \frac{\Delta y}{\max_{i,j} |V^n|} \right] \tag{14}$$

Spatially, fourth-order accurate, four-step, explicit Runge–Kutta schemes, see Equations (15) and (16), have been widely employed for path line tracing [25]. Equations (15) and (16) use the same symbolic notation as Equations (12) and (13) with the subscript representing location, with the superscript, n, denoting an explicit velocity value, and with the subscript, k, on n specifying bilinear interpolation. The same partial time step restrictions from Equation (14) govern the value of τ. Bilinear interpolation is employed for the velocity values because little difference has been found in trajectory location between using bilinear and cubic interpolation [1, 19]. Equations (12), (13), (15), and (16) are shown in explicit form; however, these same

path line tracing methods can be employed in implicit, iterative formulations.

$$
\begin{aligned}
x_{s-1} &= x_s - \frac{\tau}{6}\left(U^{n_k}_{x_s,y_s} + 2U^{n_k}_{x_{sp1},y_{sp1}} + 2U^{n_k}_{x_{sp2},y_{sp2}} + U^{n_k}_{x_{sp3},y_{sp3}}\right)\\
x_{sp1} &= x_s - U^{n_k}_{x_s,y_s}\frac{\tau}{2}\\
x_{sp2} &= x_s - U^{n_k}_{x_{sp1},y_{sp1}}\frac{\tau}{2}\\
x_{sp3} &= x_s - U^{n_k}_{x_{sp2},y_{sp2}}\tau
\end{aligned}
\tag{15}
$$

$$
\begin{aligned}
y_{s-1} &= y_s - \frac{\tau}{6}\left(V^{n_k}_{x_s,y_s} + 2V^{n_k}_{x_{sp1},y_{sp1}} + 2V^{n_k}_{x_{sp2},y_{sp2}} + V^{n_k}_{x_{sp3},y_{sp3}}\right)\\
y_{sp1} &= y_s - V^{n_k}_{x_s,y_s}\frac{\tau}{2}\\
y_{sp2} &= y_s - V^{n_k}_{x_{sp1},y_{sp1}}\frac{\tau}{2}\\
y_{sp3} &= y_s - V^{n_k}_{x_{sp2},y_{sp2}}\tau
\end{aligned}
\tag{16}
$$

Two different test scenarios were chosen to compare departure point determination using the SUT method, Equations (6)–(10), to the one-step explicit, Euler method, Equations (12)–(14), and the explicit four-step classical Runge–Kutta method, Equations (14)–(16). For the remainder of the paper, the three methods will be referred to as the SUT method, the Euler method, and the Runge–Kutta method, respectively. The first test case is path line tracing around the top-half of a submerged cylinder, and the second scenario is the application of semi-Lagrangian advective transport to the rotation of Gaussian concentration hill.

3.1. Flow around the top-half of a submerged cylinder

The first flow scenario, displayed schematically in Figure 1, employed to compare the three different methods is flow around the top half of a submerged cylinder centred on the origin [27]. An analytic solution in radial coordinates exists for the velocity field and for streamlines in this scenario. The stream function, radial component, and angular component of velocity are given by

$$v_r = U\left(1 - \frac{a^2}{r^2}\right)\cos\theta \tag{17}$$

$$v_\theta = -U\left(1 + \frac{a^2}{r^2}\right)\sin\theta \tag{18}$$

$$\psi = U r\left(1 - \frac{a^2}{r^2}\right)\sin\theta \tag{19}$$

where v_r is the radial velocity, v_θ is the angular velocity, r is the radial distance, a is the radius of the cylinder, U is the magnitude of uniform fluid velocity away from the cylinder, ψ is the stream function, and θ is the angle. In this scenario, a cylinder of radius 8 m is centred

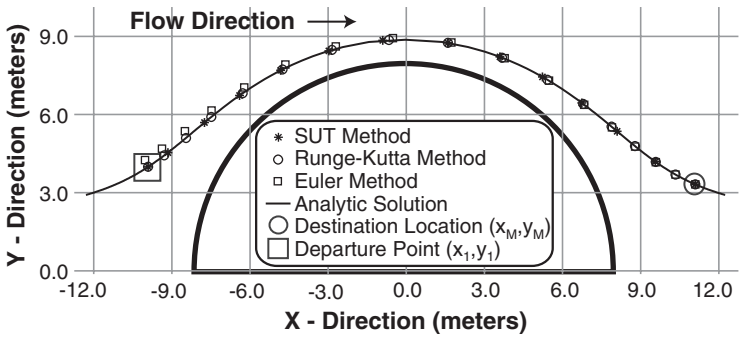

Figure 3. Path line tracing in uniform flow around a submerged half-cylinder with $\Delta x = \Delta y = 0.2500$ m. The destination point, or initial location (x_M, y_M), for all three methods is in the large circle on the right. The calculated departure point, (x_1, y_1) for all three methods is in the large rectangle on the left. The time interval, Δt, for path line tracing is 20 s. For each method, the destination location, $s = M$, the final location, $s = 1$, and every 10th location, $s = M - 10, M - 20, M - 30, \ldots, 1$, are marked.

on the origin in uniform flow of 1.0 m/s. The radial velocities were transformed into Cartesian component velocities located in the centre of the faces of square computational volumes. A streamline running through the x-direction velocity location chosen as the destination point was calculated with Equation (19) and transformed into Cartesian co-ordinates to compare the particle path lines generated with the three methods. The destination point was chosen close enough to the submerged cylinder so that the velocity variations in the traversed computational volumes would be non-linear. Two different size computational volumes, $\Delta x = \Delta y = 0.250$ m and $\Delta x = \Delta y = 0.0833$ m, were employed in separate sets of simulations for each method. A total time step, Δt, of 20.0 seconds was used to ensure that the tracked particles would travel completely past the cylinder. Using Equation (20) and $u_{\max}$, which is the maximum Cartesian directional velocity component, of 2.0 m/s, the grid Courant numbers, $Cr_{\max}$, for $\Delta x = 0.250$ and 0.0833 are 160 and 480, respectively.

$$Cr = u_{\max} \frac{\Delta t}{\Delta x} \tag{20}$$

Figure 3 displays the results obtained with the coarse resolution, $\Delta x = \Delta y = 0.250$ m. The initial location, or destination point, is on the right side of the figure and flow is from left to right. The calculated departure points are on the left side. In Figure 3 locations are shown for each method for the destination point, the departure point location, and for every tenth location, $s = M, M - 10, M - 20, M - 30, \ldots, 1$.

Table I displays the results for this scenario. Here, relative calculation time is the average calculation time for 10 identical simulations for a particular method divided by the average calculation time for the SUT method for ten equivalent simulations. Calculation time was determined with a CPU 'stopwatch timer.' Error in percentage of Δx is the distance from the calculated departure point to the actual departure point divided by Δx and multiplied by 100. In this scenario the 'true' departure location was obtained using the Runge–Kutta method with the finer grid resolution and by doubling the number of partial time steps, M, until the departure point location converged to eight significant digits.

Table I. Results for flow around the upper half of a cylinder for $\Delta x = \Delta y = 0.250$ m.

Method	M^*	$x_1^\dagger$	$y_1^\dagger$	Relative time‡	Error in % Δx§
SUT	128	−9.885	4.047	1.00	0.99
Euler	158	−9.945	4.307	0.47	113.47
	316	−9.914	4.179	0.91	57.90
	632	−9.899	4.115	1.77	29.27
	1264	−9.891	4.082	3.53	14.72
	2528	−9.886	4.065	6.92	7.38
	5056	−9.884	4.057	13.88	3.70
	10 112	−9.883	4.053	27.75	1.85
	20 224	−9.883	4.051	55.86	0.93
	40 448	−9.883	4.050	110.89	0.47
	80 896	−9.883	4.049	220.80	0.24
	161 792	−9.882	4.049	442.31	0.12
Runge–Kutta	158	−9.883	4.049	1.54	0.07
	316	−9.882	4.049	3.17	0.04

*M is the number of partial time steps necessary to trace the path line for the model time step of 20 s.

†(x, y) gives the x- and y-co-ordinates of the departure point location.

‡Relative time is the average computational time for a particular method across ten identical simulations divided by the average computational time for the SUT method across 10 identical simulations. Computational time is calculated with a stop watch timer.

§The error in % of Δx is the distance between the calculated (x, y) and the solution (x, y) divided by Δx and multiplied by 100. The solution (x, y) was obtained by doubling the number of partial time steps, M, for the Runge–Kutta method with the finer discretization, $\Delta x = \Delta y = 0.0833$ m, until (x, y) converged to eight significant figures.

From Table I, the Euler method and the Runge–Kutta method employ a minimum of 156 partial time steps, $M = 156$, of duration τ from Equation (11). The SUT method uses 128 partial time steps, $M = 128$, because the tracked particle traverses all or part of 128 computational cells. The duration of each partial time step, τ_e, for the SUT method is calculated with Equation (9).

Even though the velocity variations are non-linear in this case and the SUT method assumes linear velocity variation, the SUT method and the Runge–Kutta method provide similar departure point locations. The Runge–Kutta method provides a slightly more accurate final location because the velocity variation in each coordinate direction across each computational volume traversed is not linear. However, the Runge–Kutta method requires 50% more computational time than the SUT method to obtain a departure point location.

The departure point location obtained with the Euler method is displaced by more than one volume length, Δx, from the 'true' location when the partial time step duration from Equation (14) is employed. To compare an equivalent accuracy Euler method with the SUT method, the number of partial time steps employed with the Euler method were doubled until the Euler method departure point corresponds to three decimal places with the 'true' departure point at $M = 161\,792$. The first doubling of partial time steps, $M = 2528$, where the error in percentage Δx falls below 10% was chosen to compare computational efficiency of a more

Table II. Results for flow around the upper half of a cylinder with $\Delta x = \Delta y = 0.0833$ m.

Method	M^*	$x_1^\dagger$	$y_1^\dagger$	Relative time‡	Error in % Δx§
SUT	380	−9.882	4.049	1.00	0.97
Euler	479	−9.924	4.135	0.45	115.32
	958	−9.904	4.092	0.90	58.08
	1916	−9.893	4.071	1.80	29.15
	3832	−9.888	4.060	3.60	14.60
	7664	−9.885	4.054	7.03	7.31
	15 328	−9.884	4.051	14.25	3.66
	30 656	−9.883	4.050	28.33	1.83
	61 312	−9.883	4.049	56.41	0.91
	122 624	−9.883	4.049	104.25	0.46
	161 792	−9.883	4.049	133.79	0.35
	245 248	−9.882	4.049	200.02	0.23
Runge–Kutta	479	−9.882	4.049	1.59	0.02

*M is the number of partial time steps necessary to trace the path line for the model time step of 20 s.

†(x, y) gives the x- and y-co-ordinates of the departure point location.

‡Relative time is the average computational time for a particular method across 10 identical simulations divided by the average computational time for the SUT method across ten identical simulations. Computational time is calculated with a stop watch timer.

§The error in % of Δx is the distance between the calculated (x, y) and the solution (x, y) divided by Δx and multiplied by 100. The solution (x, y) was obtained by doubling the number of partial time steps, M, for the Runge–Kutta method with the finer discretization, $\Delta x = \Delta y = 0.0833$ m, until (x, y) converged to eight significant figures.

accurate Euler method with the SUT method. To obtain close to the same level of accuracy as the SUT method, the Euler method requires almost 7 times the computational time.

Table IIpresents the results for the finer discretization, $\Delta x = 0.0833$. Here, the Euler method and the Runge–Kutta method employ a minimum of 479 partial time steps, $M = 479$, of duration τ from Equation (11). The SUT method uses 380 partial time steps, $M = 380$, because the tracked particle traverses all or part of 380 computational cells. Both the SUT method and the Runge–Kutta method provide essentially the same departure point as the 'true' departure point, but the Runge–Kutta method requires 50% more computational time than the SUT method.

Again, the Euler method provides a departure point location greater than Δx away from the 'true' location. A departure point discrepancy of this magnitude can create a significant error in the interpolated value in the interpolation step of a semi-Lagrangian scheme when using cubic or higher order interpolation. The number of partial time steps for the Euler method was doubled until the Euler method departure point is equal to the 'true' solution to three decimal places. At $M = 7664$, the Euler method provides a departure point that is within $0.10\Delta x$ of the 'true' location. Just as in the coarse resolution scenario, the Euler method requires 7 times the computational time of the SUT method to obtain this level of accuracy.

3.2. Rotation of a Gaussian concentration hill

The next application employs the departure point determination schemes in a semi-Lagrangian representation of the advection of a passive scalar, Equation (2). The semi-Lagrangian algorithm employed here assumes that concentration is defined at the center of each computational volume. Advection of scalar concentration is simulated by finding the departure point for each volume center location for each time step and then by employing bicubic Lagrange polynomial interpolation (The stars in Figure 1 provide example of the bicubic Lagrange polynomial interpolation stencil for a two-dimensional Cartesian grid.) to estimate the concentration at the departure point. The estimated departure point concentration at time n provides the destination point concentration at time $n+1$. Although this semi-Lagrangian method is simple and non-conservative, the application of the three departure point determination schemes within the algorithm highlights the influence of departure point accuracy on the representation of advection.

This simple, semi-Lagrangian scheme is employed, using all three departure point determination methods, to simulate the advection of a Gaussian concentration hill in a rotational flow field. The concentration solution, Equation (21), and the flow field, Equation (22), were adapted from Reference [28].

$$C(x, y, t) = \exp(1) - \left\{ \frac{(x-\bar{x})^2}{2\sigma_0^2} + \frac{(y-\bar{y})^2}{2\sigma_0^2} \right\} \tag{21}$$

$$U = -\omega y, \quad V = \omega x \tag{22}$$

In these equations, C is concentration in kg/m^3; x is the x-co-ordinate of the centre of the concentration hill; y is the y-co-ordinate of the concentration hill centre; σ_0 is the standard deviation of the initial concentration hill, and ω is the angular velocity.

Figure 4 presents the rotation scenario employed in this application for comparison of the three departure point determination methods. The simulation domain is 101 rows by 101 columns with $\Delta x = \Delta y = 1.0$ m. The standard deviation of the concentration hill is 2.0 and an angular velocity of $1/18$ rad/s is employed to give the concentration hill a velocity of 1 m/s. The initial location of the concentration hill is $x_0 = 0.0$ and $y_0 = 18.0$ m. Maximum initial concentration is $\exp(1) \approx 2.72$ and minimum initial concentration is 0.0. Flow is counterclockwise, and each simulation lasts at least 90 s. If the time step duration, Δt, does not divide evenly into 90 s, then the next larger whole number is employed for total time so that the end of the simulation period will coincide with the end of a model time step.

Seven different simulations were completed with each departure point determination method with different time steps, Δt, to give $Cr_{\max}$ values ranging from 0.5 to 10.0. Figure 5 displays a mesh surface plot of the simulated concentration hill after 90 s using the Runge–Kutta method for departure point determination and employing $\Delta t = 10.0$ s; all three methods provide similar surface plots for all of the seven time steps. To examine the influence of departure point location on accuracy of the semi-Lagrangian representation of advection in this scenario, three different diagnostic error measurements were employed [28, 29]. The ℓ_1 measure, Equation (23), is the integral measure of error; ℓ_2, Equation (24), is the integral measure of squared

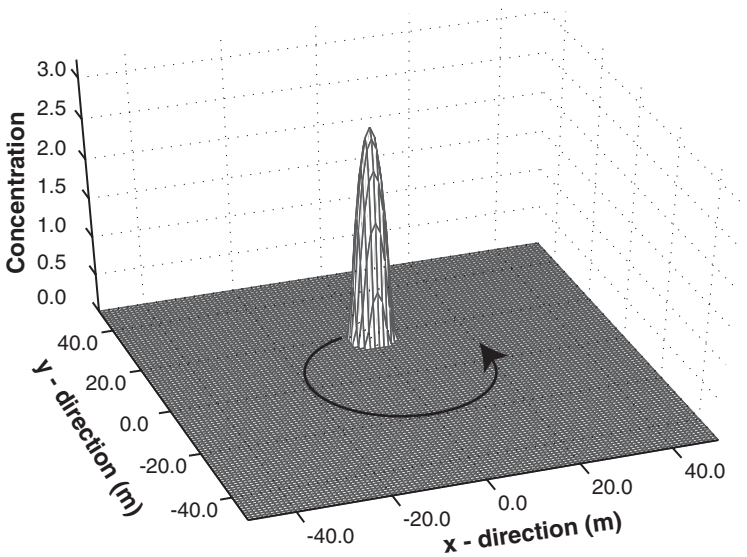

Figure 4. Layout of the advection of a Gaussian concentration hill in a rotational flow field scenario. The domain has 101 rows and 101 columns with $\Delta x = \Delta y = 1.0$ m. The concentration hill has a maximum initial concentration of approximately 2.72 kg/m and has a standard deviation of 2.0. The hill is allowed to travel for at least 90 s in the rotational flow field so that it traverses approximately $\frac{3}{4}$ of the simulation domain. The arrow displays the direction of fluid flow. During the movement of the hill, the same side of the hill faces the origin.

error, and ℓ_{inf} in Equation (25) provides the maximum local error.

$$\ell_1 = \frac{\sum_{i=1}^{N_r} \sum_{j=1}^{N_c} |C_{i,j} - \hat{C}_{i,j}|}{\sum_{i=1}^{N_r} \sum_{j=1}^{N_c} |\hat{C}_{i,j}|} \tag{23}$$

$$\ell_2 = \frac{\left[\sum_{i=1}^{N_r} \sum_{j=1}^{N_c} (C_{i,j} - \hat{C}_{i,j})^2\right]^{1/2}}{\left[\sum_{i=1}^{N_r} \sum_{j=1}^{N_c} \hat{C}_{i,j}^2\right]^{1/2}} \tag{24}$$

$$\ell_{\text{inf}} = \frac{\max_{\forall i,j} |C_{i,j} - \hat{C}_{i,j}|}{\max_{\forall i,j} |\hat{C}_{i,j}|} \tag{25}$$

In Equations (23)–(25), C is solution concentration at the end of the simulation; $\hat{C}$ is the exact solution; N_r provides the number of rows in the simulation domain, and N_c gives the number of columns.

Table III provides the results from the 21 simulations. The relative ℓ_1, ℓ_2, and ℓ_{inf} values are simply the error measurement for a particular method divided by the equivalent error measurement for the SUT method. As a result, a relative error value of 1.0 is equivalent to the SUT method; a value less than 1.0 provides a better match to the exact solution than the

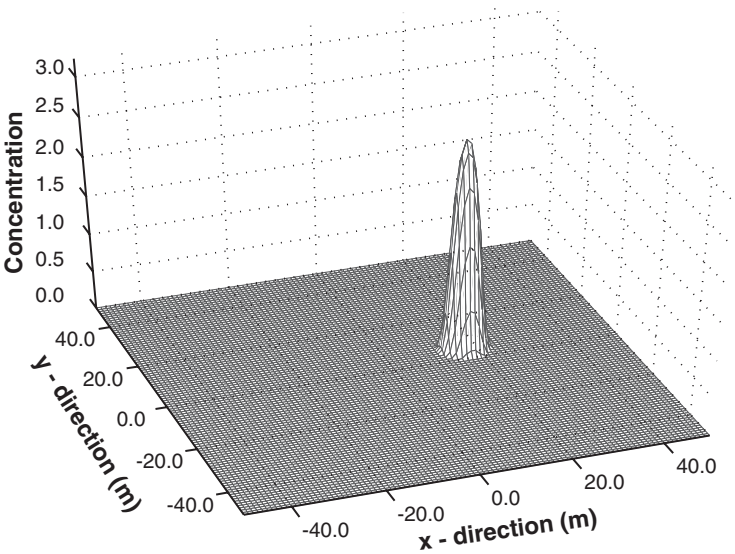

Figure 5. Representative mesh surface plot of the simulated concentration after 90 s. This simulation employed the Runge–Kutta method and a time step, Δt, of 10.0 s. The simulated concentrations with the SUT method and with the Euler method are similar in appearance.

SUT method, and a value greater than 1.0 is a worse match to the exact solution than the SUT method. The relative error for a particular method can decrease with increasing time step if that method's performance improves relative to the SUT method as time step increases. However, a decrease in relative error with time step increase does not require that method provide a better representation of the exact solution with increased time step. Relative time is calculated in the same manner as in Section 3.1 and is net of I/O.

The results in Table III for the departure point determination methods are similar to those in Section 3.1. For $Cr_{\max}$ values greater than 1.0, the SUT method provides increased computational efficiency relative to the Runge–Kutta method; the Runge–Kutta method requires as much as 18% more computational time. However, the SUT method is generally not as accurate as the Runge–Kutta method in this scenario. The Euler method has more than twice the relative error of the SUT method in some cases when $Cr_{\max}$ is greater than 1.0. In this scenario, the SUT method provides reasonable, and in some cases comparable, accuracy relative to the Runge–Kutta method and has significant relative computational efficiency. For $Cr_{\max}$ values above 1.0, the Euler method has error measures of up to twice the magnitude of the other two methods.

4. CONCLUSIONS

A new one-step, semi-analytic departure point determination algorithm, the SUT method, is presented for semi-Lagrangian advection schemes on Cartesian grids. Although the SUT method is presented in explicit form for endpoint calculations, it can be modified for implicit

Table III. Results from advection of a Gaussian concentration hill in a rotational flow field.

Method	$\Delta t[s]$	Cr^*_{max}	Total time [s]†	Max. C‡	Min. C‡	Relative ℓ_1§	Relative ℓ_2§	Relative ℓ_{inf}§	Relative time¶
SUT	0.5	0.5	90.0	2.24	0.00	1.00	1.00	1.00	1.00
	1.0	1.0	90.0	2.68	0.00	1.00	1.00	1.00	1.00
	2.0	2.0	90.0	2.75	0.00	1.00	1.00	1.00	1.00
	4.0	4.0	92.0	2.71	0.00	1.00	1.00	1.00	1.00
	6.0	6.0	90.0	2.72	0.00	1.00	1.00	1.00	1.00
	8.0	8.0	96.0	2.68	0.00	1.00	1.00	1.00	1.00
	10.0	10.0	90.0	2.71	0.00	1.00	1.00	1.00	1.00
Runge–Kutta	0.5	0.5	90.0	2.29	0.00	0.79	0.70	0.54	0.99
	1.0	1.0	90.0	2.65	0.00	0.87	0.83	0.69	0.99
	2.0	2.0	90.0	2.74	0.00	0.93	0.89	0.82	1.01
	4.0	4.0	92.0	2.72	0.00	0.98	0.96	0.86	1.05
	6.0	6.0	90.0	2.72	0.00	0.99	0.98	0.91	1.08
	8.0	8.0	96.0	2.68	0.00	0.94	0.92	0.80	1.10
	10.0	10.0	90.0	2.71	0.00	0.92	0.93	0.83	1.18
Euler	0.5	0.5	90.0	2.28	0.00	0.80	0.74	0.66	0.98
	1.0	1.0	90.0	2.66	0.00	0.96	0.99	0.83	0.95
	2.0	2.0	90.0	2.75	0.00	1.16	1.20	1.05	0.95
	4.0	4.0	92.0	2.74	0.00	1.52	1.55	1.36	0.90
	6.0	6.0	90.0	2.73	0.00	1.84	1.84	1.57	0.85
	8.0	8.0	96.0	2.71	0.00	1.87	1.86	1.55	0.81
	10.0	10.0	90.0	2.72	0.00	2.19	2.13	1.66	0.80

*Cr_{max} from Equation (19).

†Total simulation time in seconds is at least 90 s. The total simulation time is set to the next greater whole number from 90 s so that the end of a model time step, Δt, will coincide with the end of the simulation.

‡Maximum and minimum concentrations in kg/m^3 remaining at the end of the simulation. Maximum initial concentration is exp(1) or approximately $2.72\,kg/m^3$.

§Relative ℓ_1, ℓ_2, and ℓ_{inf} error measurements are the ℓ_1, ℓ_2, or ℓ_{inf} value from Equations (27)–(29) divided by the equivalent error measurement for the SUT method. A value greater than 1.0 stipulates an error value larger than the SUT value, and a value less than 1.0 denotes a smaller error value than provided by the SUT method.

¶Relative time is the average computational time across 10 identical simulations for a particular method divided by the average computational time across ten identical simulations for the SUT method. Computational time is calculated with a stopwatch timer and is net of I/O.

and for midpoint departure point determination. Two test cases were employed to compare the SUT method to two traditional multi-step methods, the Euler scheme and the Runge–Kutta scheme.

The first scenario, departure point location in a flow field around the top-half of a submerged cylinder, demonstrates the accuracy and efficiency of the SUT method. The SUT method still provides close to the same accuracy as the Runge–Kutta method, and the Runge–Kutta method required approximately 160% of the computational time. The Euler scheme is relatively inaccurate. To achieve the comparable accuracy of the SUT method, the Euler method requires seven times the computational time. Compared to traditional methods, the SUT method provides both accurate departure point location and computational efficiency.

The second application applied a semi-Lagrangian representation of advection to simulate the transport of a Gaussian concentration hill in a rotational flow field. In this case, the sharp concentration gradients provided by the Gaussian hill produced significant differences in the results obtained with the three methods. For Cr_{max} values greater than 1.0, the Runge–Kutta method again required the most computational time but provided the smallest error from the three error measures. The SUT method provided reasonable and in some cases equivalent accuracy to the Runge–Kutta method and required less computational time. In some cases, the SUT method required only 90% of the computational time of the Runge-Kutta method. The Euler method provided the largest error measurements with twice the magnitude of error of the SUT method in some simulations.

ACKNOWLEDGEMENTS

This material is based upon work supported by the National Science Foundation under Grant No. EAR-0207177. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation. This material is also based upon work supported under a Stanford Graduate Fellowship. In addition, we would like to thank three anonymous referees for their valuable comments and suggestions.

REFERENCES

1. Staniforth A, Cote J. Semi-Lagrangian integration schemes for atmospheric models—a review. *Monthly Weather Review* 1991; **119**:2206–2223.
2. Cheng RT, Casulli V, Milford SN. Eulerian–Lagrangian solution of the convection-dispersion equation in natural coordinates. *Water Resources Research* 1984; **20**:944–952.
3. Oliveira A, Baptista AM. A comparison of integration and interpolation Eulerian–Lagrangian methods. *International Journal for Numerical Methods in Fluids* 1995; **21**:183–204.
4. Roache PJ. A flux-based modified method of characteristics. *International Journal for Numerical Methods in Fluids* 1992; **15**:1259–1275.
5. Bermejo R. On the equivalence of semi-Lagrangian schemes and particle-in-cell finite element methods. *Monthly Weather Review* 1990; **118**:979–987.
6. Malevsky AV, Thomas SJ. Parallel algorithms for semi-Lagrangian advection. *International Journal for Numerical Methods in Fluids* 1997; **25**:455–473.
7. Smolarkiewicz PK, Pudykiewicz JA. A class of semi-Lagrangian approximations for fluids. *Journal of Atmospheric Science* 1992; **49**:2082–2096.
8. Casulli V, Cheng RT. Semi-implicit finite difference methods for three-dimensional shallow water flow. *International Journal for Numerical Methods in Fluids* 1992; **15**:629–648.
9. Manson JR, Wallis SG. A conservative, semi-Lagrangian fate and transport model for fluvial systems—I. Theoretical development. *Water Research* 2000; **34**:3769–3777.
10. Phillips TN, Williams AJ. Conservative semi-Lagrangian finite volume schemes. *Numerical Methods for Partial Differential Equations* 2001; **17**:403–425.
11. Phillips TN, Williams AJ. A semi-Lagrangian finite volume method for Newtonian contraction flows. *SIAM Journal on Scientific Computing* 2001; **22**:2152–2177.
12. Karpik SR, Crockett SR. Semi-Lagrangian algorithm for two-dimensional advection–diffusion equation on curvilinear coordinate meshes. *Journal of Hydraulic Engineering* (ASCE) 1997; **123**:389–401.
13. Glass J, Rodi W. A higher order numerical scheme for scalar transport. *Computer Methods in Applied Mathematics* 1982; **31**:337–358.
14. Galland J-C, Goutal N, Hervouet J-M. TELEMAC: A new numerical model for solving shallow water equations. *Advances in Water Resources* 1991; **14**:138–148.
15. Hervouet J-M, Haren Lv. Recent advances in numerical methods for fluid flows. In *Floodplain Processes*, Anderson MG, Walling DE, Bates PD (eds). Wiley: Chichester, UK, 1996; 183–214.
16. Durran DR. *Numerical Methods for Wave Equations in Geophysical Fluid Dynamics*. Springer: New York, 1999.
17. Fletcher CAJ. *Computational Techniques for Fluid Dynamics*, vol. I. Springer-Verlag: New York, 1991.
18. Robert A. A stable numerical integration scheme for the primitive meteorological equations. *Atmosphere and Oceans* 1981; **19**:35–46.

19. Temperton C, Staniforth A. An efficient two-time level, semi-Lagrangian semi-implicit integration scheme. *Quarterly Journal of the Royal Meteorological Society* 1987; **113**:1025–1039.
20. Ruge JW, Li Y, McCormick S, Brandt A, Bates JR. A nonlinear multigrid solver for a semi-Lagrangian potential vorticity-based shallow-water model on the sphere. *SIAM Journal on Scientific Computing* 2000; **21**:2381–2395.
21. Carfora MF. An unconditionally stable semi-Lagrangian method for the spherical atmospherical shallow water equations. *International Journal for Numerical Methods in Fluids* 2000; **34**:527–558.
22. Williamson DL, Rasch PJ. Two-dimensional semi-Lagrangian transport with shape-preserving interpolation. *Monthly Weather Review* 1989; **117**:102–129.
23. Behrens J. An adaptive semi-Lagrangian advection scheme and its parallelization. *Monthly Weather Review* 1996; **124**:2386–2395.
24. Pollock DW. Semianalytical computation of path lines for finite-difference models. *Ground Water* 1988; **26**:743–750.
25. Zheng C, Bennett GD. *Applied Contaminant Transport Modeling*. Wiley-Interscience: New York, 2002.
26. Lu N. A semianalytical method of path line computation for transient finite-difference groundwater flow models. *Water Resources Research* 1994; **30**:2449–2459.
27. Munson BR, Young DF, Okiishi TH. *Fundamentals of Fluid Mechanics*. Wiley: New York, 2002.
28. Baptista AM, Adams EE, Gresho P. Benchmarks for the transport equation: the convection–diffusion forum and beyond. In *Quantitative Skill Assessment for Coastal Ocean Models*, Lynch DR, Davies AM (eds). American Geophysical Union: Washington, DC, 1995; 241–268.
29. Gross ES, Koseff JR, Monismith SG. Evaluation of advective schemes for estuarine salinity. *Journal of Hydraulic Engineering* (ASCE) 1999; **125**:32–46.

