Ocean Modelling 40 (2011) 52–71

Contents lists available at ScienceDirect

Ocean Modelling

journal homepage: www.elsevier.com/locate/ocemod

Reconstruction of vector fields for semi-Lagrangian advection on unstructured, staggered grids

B. Wang [a,*], G. Zhao [b], O.B. Fringer [a]

[a] Environmental Fluid Mechanics Laboratory, Department of Civil and Environmental Engineering, Stanford University, Stanford, CA 94305, USA
[b] Flow Science Incorporated, Pasadena, CA 91101, USA

ARTICLE INFO

Article history:
Received 1 November 2010
Received in revised form 12 June 2011
Accepted 27 June 2011
Available online 5 July 2011

Keywords:
Interpolation
Vector fields
Unstructured
Staggered grids
Semi-Lagrangian advection

ABSTRACT

Applying the semi-Lagrangian method to discretize the advection of momentum eliminates the Courant number constraint associated with explicit Eulerian momentum advection in coastal ocean models. Key steps of the semi-Lagrangian method include calculating trajectories and interpolating the velocity vectors at the end of trajectories. In this work, we follow the linear and quadratic interpolation methods proposed by Walters et al. (2007) for field-scale simulations on unstructured, staggered grids and compare their performance using a backward-facing step test case and field-scale estuarine simulations. A series of methods to approximate the nodal and tangential velocities needed for the interpolation are evaluated and it is found that the methods based on the low-order Raviart–Thomas vector basis functions are more robust with respect to grid quality than the methods from Perot (2000) while overall they obtain similar accuracy. Over the range of different nodal and tangential velocities, the quadratic interpolation methods consistently exhibit higher accuracy than the linear interpolation methods. For the quadratic interpolation, the overall accuracy depends on the approximation of the tangential velocity. The backward-facing step test case indicates that the quadratic interpolation behaves like Eulerian central differencing or first-order upwinding, depending on the tangential approximations. The field-scale estuarine flow test case also shows general improvement for the velocity predictions and sharper gradients in the velocity field with the quadratic interpolation. The quadratic interpolations add less than 15% to the total computational time, and parallel implementation is relatively straightforward in complex geometries.

1. Introduction

Stability and explicitness are the most attractive properties of the semi-Lagrangian schemes (also referred to as "trajectory" schemes) for coastal ocean models. Most applications of the semi-Lagrangian method use large time steps for horizontal advection with horizontal Courant numbers well beyond unity (Casulli and Walters, 2000; Hanert et al., 2005; Walters et al., 2007). Probably the most stringent constraint in estuarine modeling is the Courant number associated with vertical advection when vertical layer thicknesses go to zero during wetting and drying processes. Explicit Eulerian advection schemes may easily become unstable during wetting and drying, while the semi-Lagrangian scheme allows reasonable time step sizes and considerably improves model efficiency.

Using the semi-Lagrangian method to solve the total derivative of momentum consists of two main steps: (1) calculate backward-in-time trajectories of fluid particles that originate at each grid point where velocity is defined; and (2) interpolate the velocity vector at the trajectory end points. The total Lagrangian change in momentum is then the difference between the velocities at the trajectory end points. The original semi-Lagrangian method (Robert, 1981, 1982) evaluates the remaining terms in the momentum equations at the trajectory end points. For simplicity, we follow the localized Eulerian–Lagrangian approximation and evaluate the remaining terms at grid points. The differences are usually small (Walters et al., 2009) except when large gradients are present. Numerical details are explained in Section 2.

The overall accuracy and efficiency of the method strongly depend on the interpolation scheme and it has long been known that low-order interpolation can lead to substantial dissipation (Ritchie, 1986; McCalpin, 1988; Staniforth and Côté, 1991; Le Roux et al., 2000). Application of the semi-Lagrangian method on unstructured grids has been of recent interest due to the increasing popularity of unstructured-grid ocean models, e.g., Walters and Casulli (1998), Casulli and Walters (2000), Le Roux et al. (2000), Hanert et al. (2005), Ham et al. (2005), Fringer et al. (2006), and Walters et al. (2007). Unstructured triangular grids have attractive features because they are flexible for irregular coastlines and local mesh refinement. However, interpolating the velocity vectors on unstructured grids is not straightforward and high-order methods

* Corresponding author.
E-mail address: bingwang@stanford.edu (B. Wang).

1463-5003/$ - see front matter © 2011 Elsevier Ltd. All rights reserved.
doi:10.1016/j.ocemod.2011.06.003

are much more complex than equivalent methods on structured grids.

For unstructured, staggered grids, Walters et al. (2007) described several linear and quadratic interpolation schemes and showed that the quadratic schemes give better performance in general than the linear schemes. Vidović et al. (2004, 2009) proposed a method using a least-square fit of polynomial functions to approximate vector fields, and this method is also applicable for semi-Lagrangian advection. It incurs a higher computational cost for matrix inversion and special treatment of boundary and corner cells in complex geometry, both of which are not desirable for field-scale simulations. Hanert et al. (2005) described higher-order Kriging interpolation methods which can achieve higher accuracy but also incur an increased computational cost. Therefore, in this work, we follow the general guidelines for the interpolation methods by Walters et al. (2007) and focus on discussing the implementation procedures and evaluating the performance of the methods in greater detail.

In this paper we first examine the approximation of the velocity vectors at the grid nodes (Section 4.1) and the tangential velocity at the midpoints of edges (Section 4.2) that are needed for the interpolation. We focus on comparison of the interpolation methods from Perot (2000) to those based on low-order Raviart–Thomas (RT0) vector basis functions and two new least-square methods (Section 4.3), and analyze the convergence and the distribution of the errors. We include a brief discussion on the relative contribution from interpolation error and trajectory integration error in the overall semi-Lagrangian advection calculation in Section 4.4. Finally, we evaluate the performance of different interpolation methods for semi-Lagrangian advection using a backward-facing step test case (Section 5.1) and a field-scale estuarine simulation (Section 5.2).

2. Discretization with semi-Lagrangian advection

The interpolation methods are implemented in the SUNTANS model (Fringer et al., 2006). The model employs staggered prismatic cells that are triangular in the horizontal and structured in the vertical, as illustrated in Fig. 1. Fluxes are defined at face centers while all other variables are defined at cell centers, which are at the Voronoi points of the triangles and result in an orthogonal grid when the grid cells are acute triangles. This type of grid structure is found less dispersive than several other common grids and is free of spurious free-surface and pressure modes (Rostand and Le Roux, 2007), and is the same as what was used by Casulli and Walters (2000).

The governing equations are the three-dimensional, Reynolds-averaged Navier–Stokes equations with rotation under the Boussinesq approximation and hydrostatic assumption (in this paper, we do not employ the nonhydrostatic capability in SUNTANS), viz.

$$\frac{\partial u}{\partial t}+\nabla\cdot(\mathbf{u}u)-fv=-\frac{1}{\rho_0}\frac{\partial p}{\partial x}+\nabla_H\cdot(\nu_H\nabla_H u)+\frac{\partial}{\partial z}\left(\nu_V\frac{\partial u}{\partial z}\right), \tag{1}$$

$$\frac{\partial v}{\partial t}+\nabla\cdot(\mathbf{u}v)+fu=-\frac{1}{\rho_0}\frac{\partial p}{\partial y}+\nabla_H\cdot(\nu_H\nabla_H v)+\frac{\partial}{\partial z}\left(\nu_V\frac{\partial v}{\partial z}\right), \tag{2}$$

subject to incompressibility,

$$\nabla\cdot\mathbf{u}=0, \tag{3}$$

where $\nabla_H=\frac{\partial}{\partial x}\mathbf{e}_x+\frac{\partial}{\partial y}\mathbf{e}_y$ is the horizontal gradient operator, $u(x,y,z,t)$, $v(x,y,z,t)$ and $w(x,y,z,t)$ are the Cartesian components of the velocity vector in the x, y and z directions, and $\mathbf{u}$ and $\mathbf{u}_H$ are the three-dimensional and horizontal velocity vectors, respectively. The vertical momentum equation is not present because w is solved using incompressibility. When the free surface is solved, it evolves according to the depth-averaged continuity equation

$$\frac{\partial h}{\partial t}+\frac{\partial}{\partial x}\left(\int_{-d}^{h}u\,dz\right)+\frac{\partial}{\partial y}\left(\int_{-d}^{h}v\,dz\right)=0. \tag{4}$$

The definitions of the other terms are listed in Table 1.

We apply the semi-Lagrangian method to discretize the total derivative (unsteady and advection terms) explicitly. Rotation and horizontal diffusion are also solved explicitly while the other terms are discretized with a semi-implicit method to maintain stability of surface gravity waves as in the original SUNTANS model (described in Fringer et al. (2006), also Casulli (1990) and Casulli and Walters (2000)). We write the horizontal momentum Eqs. (1) and (2) as

$$\frac{D\mathbf{u}_H}{Dt}=\mathbf{R}_H, \tag{5}$$

where $D/Dt=\partial/\partial t+\mathbf{u}\cdot\nabla$ is the total derivative and $\mathbf{R}_H$ contains the Coriolis terms and the right-hand sides of the horizontal momentum equations. The semi-Lagrangian discretization of (5) to compute the evolution of the normal velocity, U_j, on edge j with SUNTANS is given by

$$U_j^{n+1}=U_j^{-}+\Delta t R_j, \tag{6}$$

where $R_j=\mathbf{n}_j\cdot\mathbf{R}_H$ is computed using the methods described in Fringer et al. (2006), and $\mathbf{n}_j$ is the normal vector corresponding to edge j (see Fig. 2). The second term on the right-hand side is discretized at the edge midpoints which is an Eulerian–Lagrangian simplification of the traditional semi-Lagrangian method. The methods we discuss here are for the calculations of U_j^{-} and are based on the traditional semi-Lagrangian approach. Among the right-hand side terms, the free-surface gradient and vertical diffusion terms are treated semi-implicitly to remove their time step constraints while the other terms are discretized explicitly. The time step of our model is most limited by the wetting and drying condition which is explained in Wang et al. (2009). For each time step, the normal

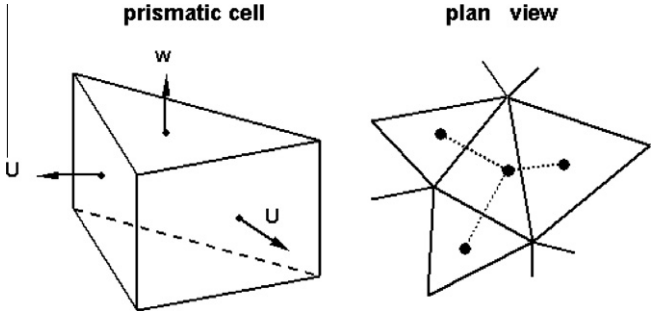

Fig. 1. The staggered prismatic grid cell with normal velocity components on the faces and cell centers defined at Voronoi points to ensure orthogonality for all acute triangle cells. Orthogonality requires that the Voronoi edges (dashed lines in the plan view) are perpendicular to the Delaunay edges (the triangle edges).

Table 1
Nomenclature of the terms in the momentum equations.

u, v, w	Cartesian components of the velocity vector, m s^{-1}
$\mathbf{u}$	Velocity vector, m s^{-1}
h	Free surface, m
d	Depth, m
t	Time, s
g	Gravitational acceleration, m s^{-2}
ρ_0	Reference density, kg m^{-3}
p	Pressure, N m^{-2}
f	Coriolis coefficient, m^{-1}
ν_H, ν_V	Horizontal and vertical eddy-viscosities, m^2 s^{-1}

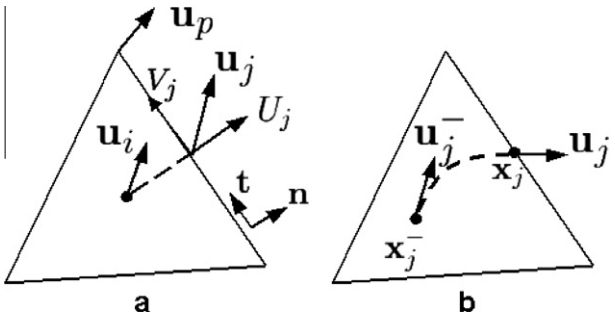

Fig. 2. (a) Definition of the velocity vectors and (b) illustration of the trajectory that departs from $\mathbf{x}_j$ and arrives at $\mathbf{x}_j^-$. $\mathbf{u}_p$ is the velocity vector at node p, $\mathbf{u}_i$ is the velocity vector at cell center i, and $\mathbf{u}_j$ is the velocity vector at the midpoint of edge j. U_j and V_j are the normal and tangential components of $\mathbf{u}_j$ at edge j, and $\mathbf{n}$ and $\mathbf{t}$ are the unit normal and tangential vectors, respectively.

component of the velocity at the end of the traceback, U_j^-, is obtained via interpolation at location $\mathbf{x}_j^-$. As depicted in Fig. 2, $\mathbf{x}_j^-$ is the arrival location of the Lagrangian traceback obtained with the backward-in-time integration

$$\mathbf{x}_j^- = \mathbf{x}_j - \int_{t^n}^{t^n+\Delta t} \mathbf{u}^n(\mathbf{x}(t))\mathrm{d}t, \tag{7}$$

where the time step superscript is ignored on $\mathbf{x}_j$ since this location is fixed in time. We employ the first-order Euler integration

$$\mathbf{x}_j^- = \mathbf{x}_j - \Delta t \mathbf{u}_j^n, \tag{8}$$

where $\mathbf{u}_j^n = \mathbf{u}^n(\mathbf{x}_j)$ is the velocity vector at edge j and time n. Therefore, the velocity field at time step n is used both for calculating $\mathbf{x}_j^-$ and interpolating $\mathbf{u}_j^-$ at $\mathbf{x}_j^-$, and the discretization is first-order accurate in time. If higher-order time accuracy is desired, multiple time levels can be used instead (Staniforth and Côté, 1991; Hortal, 2002; Lauritzen et al., 2006). The trajectory calculation relies on the interpolation to provide $\mathbf{u}^n(\mathbf{x})$ regardless of the numerical methods, and taking substeps requires interpolation at every substep. $\mathbf{u}_j^-$ is calculated with the interpolation after $\mathbf{x}_j^-$ is obtained. Since interpolation is required for both the trajectory calculation and computation of $\mathbf{u}_j^-$, accurate interpolation is a crucial component of the semi-Lagrangian method. The descriptions on the interpolation methods for the horizontal staggered grid in Walters et al. (2007) are general and hence can be implemented in many different ways. In this paper, we describe and compare several of the most efficient options. Vertical interpolation employs second-order linear interpolation (we always refer to the local order of accuracy) on the Cartesian z-level grid, which does not limit the accuracy of our calculations and so it is not discussed in this paper. In summary, the steps required for the semi-Lagrangian method are given by

1. Approximate the velocity vector at edge j to obtain $\mathbf{u}_j^n$. Since the discrete governing equations advance U_j^n, the tangential component of the velocity at edge j, V_j^n, must be approximated to obtain $\mathbf{u}_j^n = U_j^n\mathbf{n}_j + V_j^n\mathbf{t}_j$ (see Fig. 2). Methods to approximate the tangential velocity are discussed in Section 4.2.
2. Compute the location of the traceback, $\mathbf{x}_j^-$. If one step is employed, then the location can be computed directly with Eq. (8). However, if multiple steps are employed, then the velocity vector must be interpolated at each substep.
3. Interpolate to obtain the components of the velocity vector at location $\mathbf{x}_j^-$, i.e., $\mathbf{u}_j^-$, using the methods outlined in Section 4.3. The normal component for use in Eq. (6) is then obtained with $U_j^- = \mathbf{n}_j \cdot \mathbf{u}_j^-$.

3. Test flow field and meshes

We use a parabolic flow field (Fig. 3) and two triangular meshes (Fig. 4) to evaluate the interpolation. For ease of discussion, the test flow field is unidirectional, constant in the streamwise direction and varying laterally with $u = -6y^2 + 6y$ in a rectangular domain defined by $0 \leqslant x \leqslant 2.5$ and $0 \leqslant y \leqslant 1$ (the velocity field and domain are both dimensionless). This simple velocity field is found to be representative for general cases including non-unidirectional fields that vary quadratically in both the x and y directions. Therefore, the properties discussed here can be considered general for quadratic flow fields unless otherwise noted. Among the two test meshes, one is equilateral and the other is slightly skewed and is generated using GAMBIT (Fluent, Inc., Lebanon, NH). The average grid spacing in terms of average edge length is 0.052 for the equilateral grid and 0.051 for the skewed grid. The equilateral mesh consists of equilateral triangles of the same size, and the left and right boundaries of the domain are jagged to coincide with the triangle edges and ensure no angle skewness, which we define as the maximum deviation from 60° among the three angles of a cell. The skewed mesh has an average angle skewness of 4.5° which is a typical value for the grids we generate with GAMBIT for our simulations. In order to analyze spatial convergence, similar sets of equilateral and skewed meshes are generated for a series of grid spacings with average edge lengths 0.1, 0.025, and 0.0125 and average angle

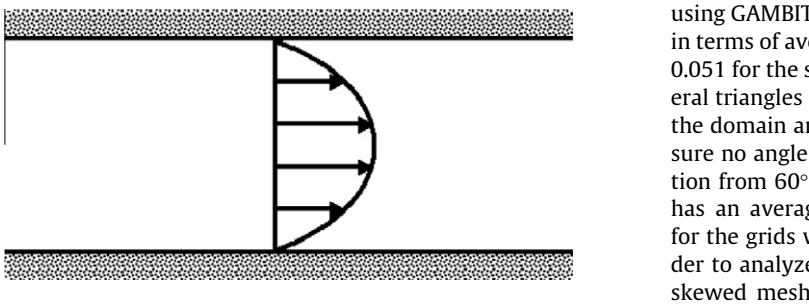

Fig. 3. Unidirectional parabolic velocity profile, $u = -6y^2 + 6y$.

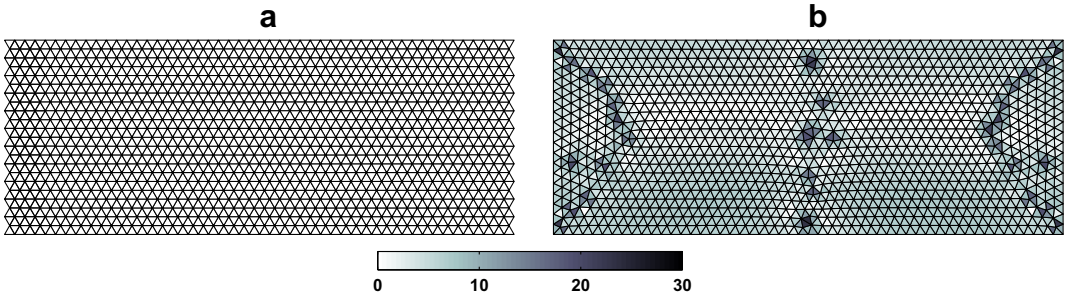

Fig. 4. The equilateral (a) and lightly skewed (b) meshes with grid spacing 0.05. Grayscale depicts the distribution of angle skewness, in degrees.

skewness 5.1°, 4.8° and 4.6°, respectively. To evaluate the interpolation methods, normal velocities at the edge centers (U_j) are first computed from the prescribed parabolic field. These are used to interpolate the velocities at various locations following different methods, and finally errors are calculated as the difference between the interpolated and true values.

4. Interpolation methods

On unstructured, staggered grids, there are numerous ways to approximate the nodal ($\mathbf{u}_p$) and tangential (V_j) velocities from the normal velocities (U_j). The methods we discuss are variations of two fundamental approaches: (1) the flux-based methods from Perot (2000) described in Appendix A, and (2) low-order Raviart–Thomas (RT0) vector basis functions described in Appendix B. Perot (2000) suggested methods to obtain both cell-centered and nodal velocities, while the RT0 basis functions are mainly used for obtaining nodal velocities.

4.1. Nodal velocities

Nodal velocities must be calculated for both linear and quadratic interpolations. We consider the following five methods for obtaining the vector $\mathbf{u}_p$ at node p (Fig. 5):

1. (nP1) area-weighted average of the cell-centered velocities calculated with Perot's method (Appendix A) in the cells surrounding node p, i.e., cell $i_1, i_2, \ldots, i_6$ in Fig. 5(a).
2. (nP2) Perot's method for nodal velocities (Appendix A) that uses the normal velocities on all edges surrounding node p, i.e., edges $j_1, j_2, \ldots, j_6$ in Fig. 5(b).
3. (nRT1) nodal velocities calculated using RT0 basis functions (Appendix B) for node p using the adjacent edges j_1 and j_2 in cell i (Fig. 5(c)). These nodal velocities differ when calculated in different cells for the same node and thus are considered "local" nodal velocities (Walters et al., 2007).
4. (nRT2) area-weighted average of the local nodal velocities calculated with method nRT1 in the cells surrounding node p, i.e., cells $i_1, i_2, \ldots, i_6$ in Fig. 5(d). This represents a least-square fit of the approximated velocity nRT1 obtained in the adjacent cells to standard linear basis functions at the node.
5. (nLS) least-square solution based on RT0 basis functions that assumes piecewise-constant normal velocity distribution along each edge. This method is a variation of method nRT2, and it minimizes the difference between the projection of the nodal velocity $\mathbf{u}_p$ to the edge normal and the normal velocities on all edges surrounding node p, as depicted in Fig. 5(e). The equation for each edge, e.g., edge j_1, connected to node p is given by

$$\mathbf{n}_{j1} \cdot \mathbf{u}_p = U_{j1},$$

where $\mathbf{u}_p$ is the velocity vector at node p, the unknown being solved, and $\mathbf{n}_j$ is the unit normal vector corresponding to edge j as illustrated in Fig. 2. The system is usually over-determined because the number of edges exceeds two. At a sharp corner node that only has one cell (i.e., two edges), the method automatically reduces to method nRT1. This is the simplest form of the least-square method for nodal velocities. Higher-order methods can be formulated by fitting a linear velocity field which incurs an enlarged stencil (see Vidović et al., 2004) and requires special treatment at boundary nodes.

Method nRT1 requires the least information, while the other methods use considerably larger stencils. However, the stencils are limited to the cells in the vicinity of the node, so no parallel overhead associated with interprocessor communication is incurred beyond that required in most standard parallel implementations.

Of the above methods, nP1, nP2, nRT2 and nLS are exact at interior nodes on equilateral grids for velocity fields that vary linearly in space. Boundary nodes are less accurate because the interpolation stencil is no longer symmetric. The local nodal velocities solved with method nRT1, which is only accurate for constant velocity fields, have relatively large errors of different signs depending on the location of the nodes with respect to the velocity gradient. Fig. 6 shows the errors in the streamwise velocity for the parabolic flow field (Fig. 3) on the test meshes (Fig. 4). Errors are defined as the difference between the interpolated and prescribed

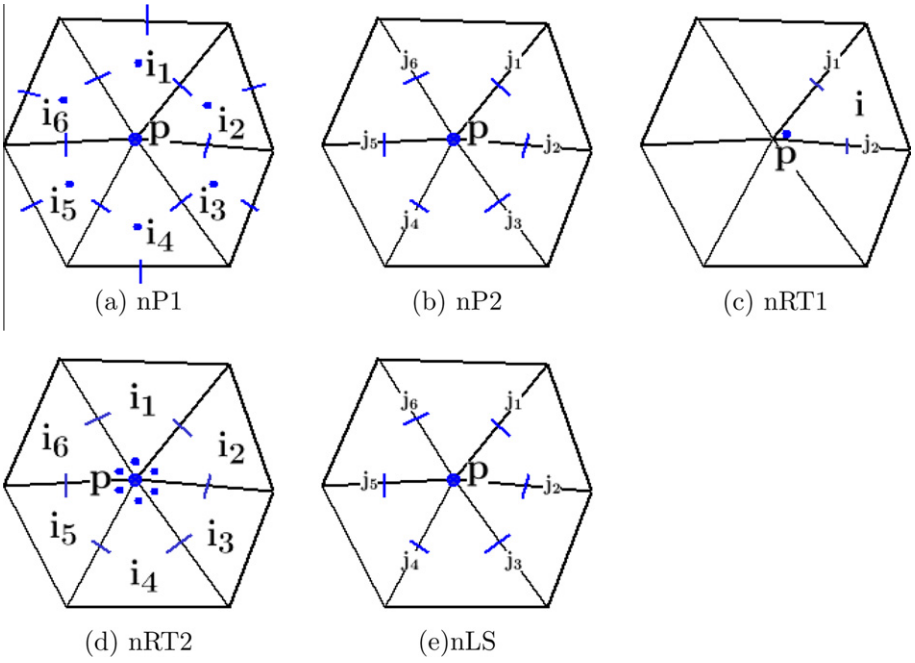

Fig. 5. Stencils for calculating the nodal velocity at node p. Short blue lines indicate the normal velocities used by each method. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

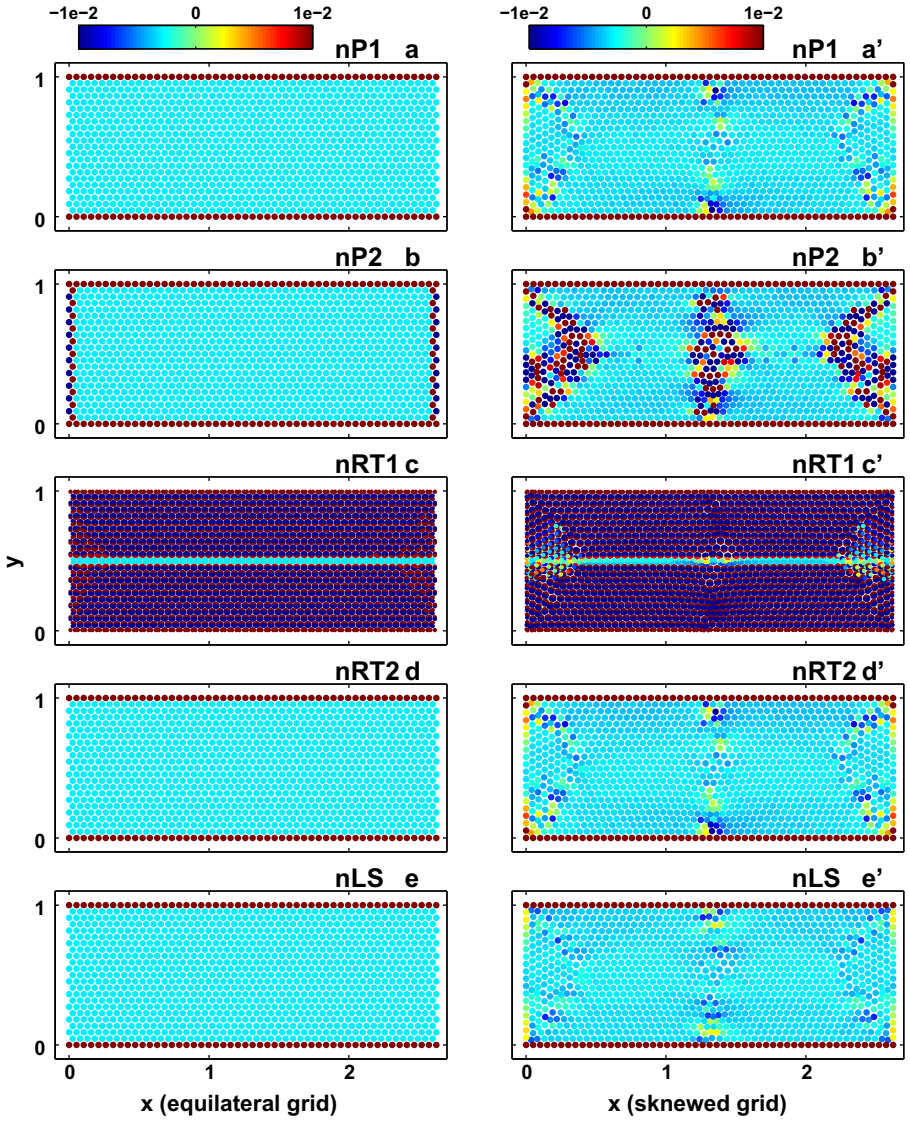

Fig. 6. Errors in the approximated nodal velocities for the parabolic velocity field on the equilateral mesh (left) and skewed mesh (right) using method nP1 (a and a′), nP2 (b and b′), nRT1 (c and c′), nRT2 (d and d′) and nLS (e and e′).

values at the nodes. Methods nP1, nP2, nRT2 and nLS result in a constant error of approximately -3.1×10^{-3} throughout the interior nodes of the domain on the equilateral mesh.

On the skewed mesh, errors increase considerably in the vicinity of the skewed cells and are of similar magnitude for methods nP1, nRT2 and nLS, while method nP2 is more sensitive to the grid quality. The local nodal velocities from method nRT1 are consistently the least accurate, with errors on the order of 10^{-1}. These errors are reduced when averaged over the cells around the node in the calculation of method nRT2. Similarly, method nP1 achieves higher accuracy through averaging because the cell-centered velocities from Perot's method are also much less accurate. The methods do not perform as well for boundary nodes because the stencil becomes smaller and asymmetric. Although the accuracy of boundary nodes can be improved with modifications, we do not specifically discuss these modifications in this manuscript.

Fig. 7 depicts the spatial convergence of the methods for obtaining the nodal velocities of the parabolic field on the test meshes. The y-axis is the absolute value of the error in the streamwise velocity averaged over the nodes within the rectangular area that excludes the boundaries and is defined by $x_{max}/8 \leqslant x \leqslant 7x_{max}/8$ and $y_{max}/8 \leqslant y \leqslant 7y_{max}/8$, where $x_{max} = 2.5$ and $y_{max} = 1$. Four levels of grid spacing, i.e., 0.1, 0.05, 0.025 and 0.0125, are used. Methods nP1, nP2, nRT2 and nLS on the equilateral mesh are the most accurate. The results for these methods coincide and the error decays with a slope of two with the grid spacing implying second-order accuracy. On the skewed grid, methods nP1, nRT2 and nLS retain their accuracy, while method nP2 has deteriorated significantly with errors one order of magnitude greater than those on the equilateral mesh. Method nRT1 is first-order and the least accurate, as expected. It has an error that is one to two orders of magnitude greater than those of the other methods, and the error is not affected by the mesh skewness.

The error for the second-order methods in the streamwise velocity u on an equilateral mesh is given by

$$u^e = \alpha u_{xx}\Delta x^2 + \beta u_{yy}\Delta x^2 - \gamma v_{xy}\Delta x^2 + H.O.T.,$$

where coefficients $\alpha = \frac{1}{24}\sum_k \sin\left(\frac{\pi}{3}k\right)^2 \cos\left(\frac{\pi}{3}k\right)^2$, $\beta = \frac{1}{24}\sum_k \sin\left(\frac{\pi}{3}k\right)^4$, and $\gamma = \frac{1}{12}\sum_k \sin\left(\frac{\pi}{3}k\right)^2 \cos\left(\frac{\pi}{3}k\right)^2$, for $k \in 0,1,2,3,4,5$. We obtain $\alpha = 0.03125$, $\beta = 0.09375$, and $\gamma = 0.0625$, and these are validated with numerical tests. Similarly, the error in the cross-stream velocity, v, is given by

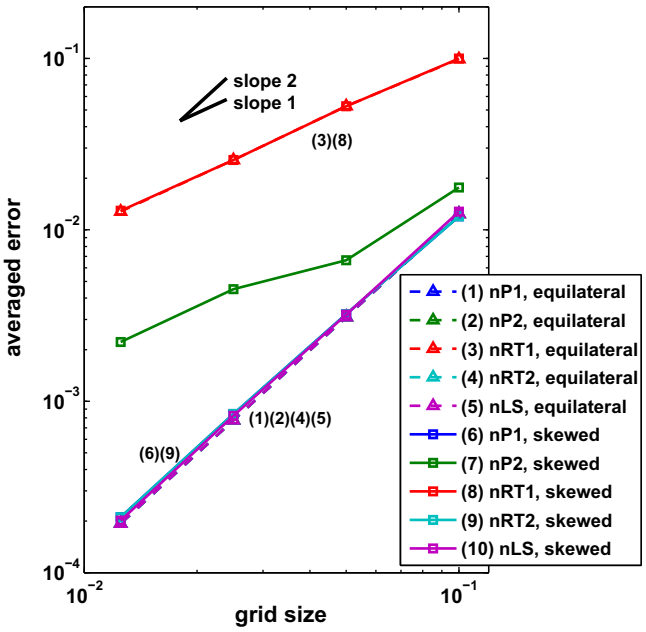

Fig. 7. Spatial convergence of methods nP1, nP2, nRT1, nRT2 and nLS in interpolating the quadratic velocity field. The y-axis is the average absolute value of the errors averaged over the edges within the region $x_{max}/8 \leqslant x \leqslant 7x_{max}/8$ and $y_{max}/8 \leqslant y \leqslant 7y_{max}/8$. Lines (1), (2), (4) and (5) are indistinguishable, and (6) and (9) are indistinguishable.

$$v^e = \beta v_{xx}\Delta x^2 + \alpha v_{yy}\Delta x^2 - \gamma u_{xy}\Delta x^2 + H.O.T.$$

This is consistent with the results in Fig. 7 which show $u^e \sim 0.1u_{yy}\Delta x^2$.

4.2. Tangential velocities

Velocity vectors at the midpoints of edges are needed explicitly for the quadratic interpolation. The key is to obtain the tangential velocity and then combine it with the normal component that is already known for each edge. We consider the following five methods for obtaining the midpoint velocity vector for edge j from which the tangential component is extracted (the interpolated normal component is discarded), and the methods are illustrated in Fig. 8.

1. (tP1) area-weighted average from cell-centered velocities calculated with Perot's method (similar to method nP1) in the two cells i_1 and i_2 neighboring edge j (Fig. 8(a)). We note that the weighting factor can be different. For instance, Ham et al. (2007) did not apply a weighting factor to obtain tangential velocities, which leads to an antisymmetric interpolation operator in the calculation of the Coriolis term.
2. (tP2) averaged from nodal velocities calculated with method nP2 at the two end nodes p_1 and p_2 corresponding to edge j (Fig. 8(b)).
3. (tRT1) area-weighted average from local nodal velocities calculated with method nRT1 at the two end nodes p_1 and p_2 in the neighboring cells i_1 and i_2 (Fig. 8(c)). Compared to nodal velocity nRT2, this can be considered a similar type of weighted approximation based on nodal velocity nRT1.
4. (tRT2) area-weighted average from the global nodal velocities calculated with method nRT2 at the two end nodes p_1 and p_2 corresponding to edge j (Fig. 8(d)). Because it uses averaged nodal velocities, this method is expected to obtain smoother results than method tRT1.
5. (tLS) least-square solution for a linear velocity field in the neighboring two cells i_1 and i_2 using four known normal velocity components (4 equations) on edges j_1, j_2, j_3 and j_4, and two approximated nodal velocity vectors (4 equations) at nodes p_1 and p_2. The normal velocity at edge j is not used because it has no contribution to the tangential component. The result is a system of 8 equations for 6 coefficients that describe a linear velocity field, i.e., $u = a_u x + b_u y + c_u$ and $v = a_v x + b_v y + c_v$. This implies that, e.g., for a node at (x_{p1}, y_{p1}) with velocity (u_{p1}, v_{p1}),

$$\begin{aligned} a_u x_{p1} + b_u y_{p1} + c_u &= u_{p1}, \\ a_v x_{p1} + b_v y_{p1} + c_v &= v_{p1}. \end{aligned} \tag{9}$$

For an edge with midpoint at (x_{j1}, y_{j1}) and normal velocity U_{j1} in direction (n_{1j1}, n_{2j1}),

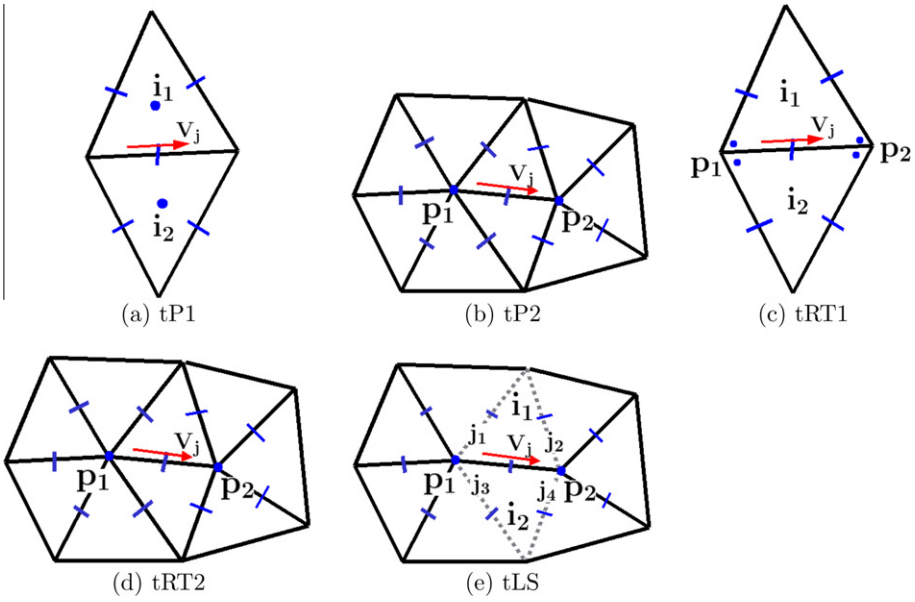

Fig. 8. Stencils for calculating the tangential velocity V_j. The dotted lines indicate the edges used for the least-squares solution. The short blue lines indicate the normal velocities used by each method. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

$$n_{1,j1}(a_u x_{j1} + b_u y_{j1} + c_u) + n_{2,j1}(a_v x_{j1} + b_v y_{j1} + c_v) = U_{j1}. \quad (10)$$

At a boundary edge, there is one adjacent cell which provides 6 equations from 2 edges and 2 nodes. The problem is well-constrained for 6 unknowns and does not require special treatment. This least-square method requires solving a 6 by 6 linear system for every edge and thus its computational cost is not trivial. There are a variety of similar least-square methods, and we only include one here as an example for the discussion of accuracy and efficiency. Compared to the least-square methods in Vidović et al. (2004), by including the approximated nodal velocities, the calculation here uses a much smaller stencil and performs more robustly for complex geometries, while the limitation is that overall accuracy is constrained by the accuracy of the nodal velocities. However, we find that, in general, fitting high-order polynomials can easily lead to ill-conditioning of the least-square methods.

As depicted in Fig. 8, methods tP1 and tRT1 use only the two cells neighboring the edge while the other methods need information from all the cells surrounding the two end nodes of the edge.

All five methods described above accurately reproduce the tangential velocities at interior edges on an equilateral mesh for velocity fields that vary linearly in space. Fig. 9 shows the errors in the approximated tangential velocities (difference between the approximated and true tangential velocities) for the parabolic velocity field (Fig. 3) on the test meshes (Fig. 4). Methods tP1 and tRT1 perform similarly, both of which result in alternating large and small errors with changing edge orientation. On the equilateral mesh, the edges parallel to the velocity have errors of roughly -3×10^{-3} while the results for nonparallel edges are accurate. These methods also have similar sensitivity to mesh quality and the maximum errors (on interior edges) reach 10^{-1} on the skewed mesh, which significantly exceeds the errors on the equilateral mesh. Methods tP2 and tRT2 are similar on the equilateral mesh, and they have errors of roughly -3×10^{-3} for interior edges. However, these methods respond quite differently to mesh quality. On the skewed mesh, the results of method tP2 produce the largest errors among the five methods, while method tRT2 is only slightly affected with errors (on interior edges) under 10^{-2}. Although the average error for method tRT2 is greater than that for methods tP1 and tRT1, it is distributed uniformly. Similar to method tP1

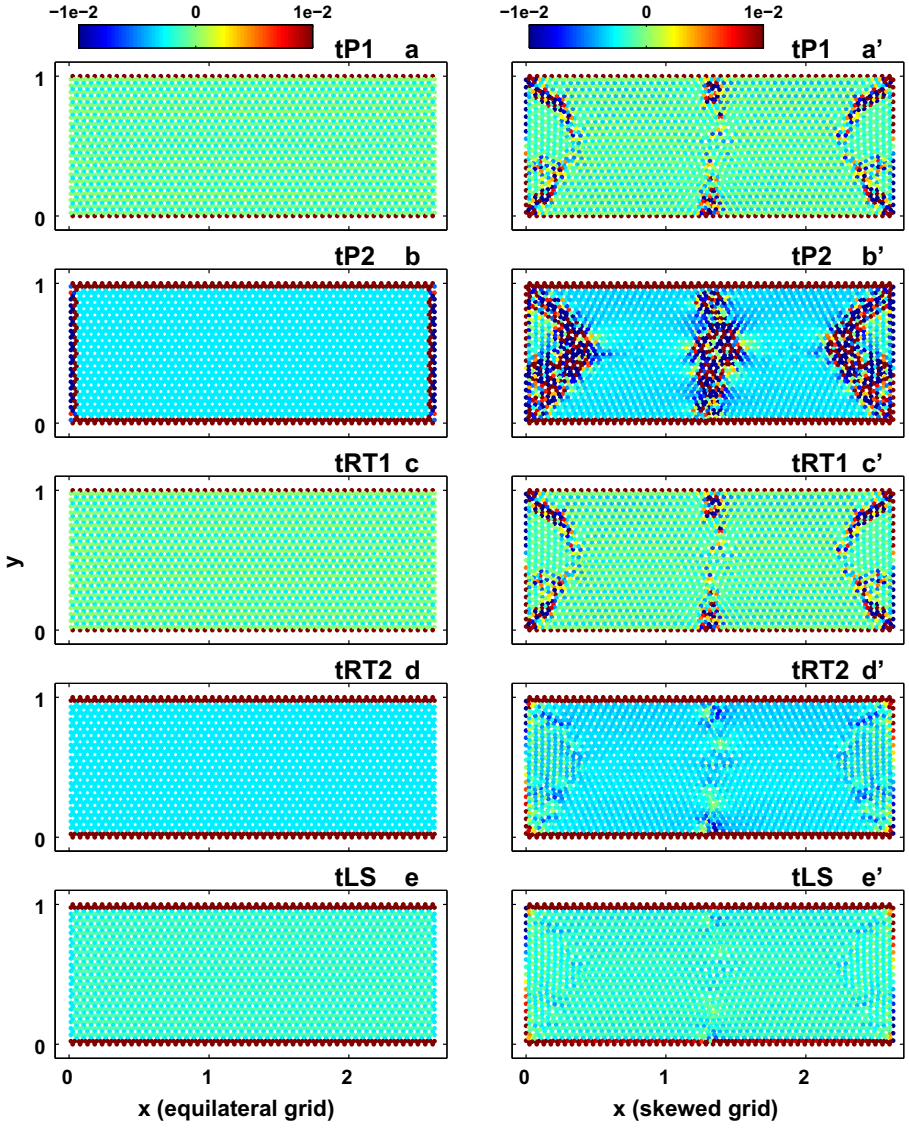

Fig. 9. Errors in the approximated tangential velocities for the quadratic velocity field on the equilateral mesh (left) and skewed mesh (right) for methods tP1 (a and a′), tP2 (b and b′), tRT1 (c and c′), tRT2 (d and d′), and tLS (e and e′).

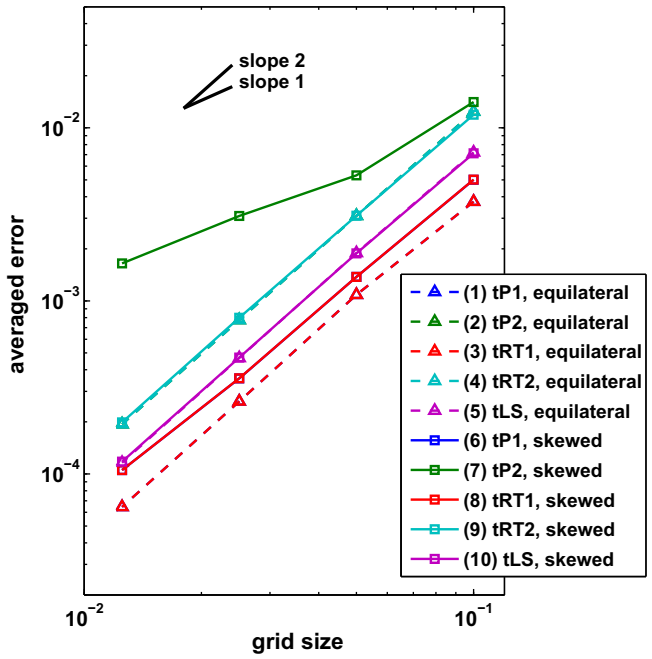

Fig. 10. Spatial convergence of methods tP1, tP2, tRT1, tRT2 and tLS in interpolating the tangential velocities for the parabolic velocity field. The y-axis is the average absolute value of the errors averaged over the edges within the region $x_{max}/8 \leqslant x \leqslant 7x_{max}/8$ and $y_{max}/8 \leqslant y \leqslant 7y_{max}/8$. Lines (1) and (3) are indistinguishable, (6) and (8) are indistinguishable, and (2) and (4) are indistinguishable.

and tRT1, the least-square method tLS has alternating errors of -3×10^{-3} for edges aligned in the x direction, and -1×10^{-3} for non-aligned edges on the equilateral mesh. Method tLS works robustly on the skewed mesh, as indicated by the errors that are not significantly affected near the skewed cells.

Fig. 10 depicts the spatial convergence of the methods for the approximation of tangential velocities for the parabolic velocity field on the test meshes. The y-axis is the absolute value of the error averaged over the edges within the rectangle defined by $x_{max}/8 \leqslant x \leqslant 7x_{max}/8$ and $y_{max}/8 \leqslant y \leqslant 7y_{max}/8$. The pairs of equilateral and skewed meshes with grid spacing, 0.1, 0.05, 0.025 and 0.0125, are used. All the methods converge at a slope of two although method tP2 does not converge as quickly on the finer skewed meshes. On the equilateral meshes, the error for methods tRT2 and tP2 is greater than that for method tLS, which is greater than that for methods tP1 and tRT1. However, methods tRT1 and tP1 are more sensitive to mesh quality. These tangential velocities are second-order accurate on equilateral meshes, and they have errors of similar magnitudes to those for the second-order accurate methods for nodal velocities, i.e., nP1, nRT2 and nLS. However, the differences between these methods of tangential velocities are more considerable than those between the methods for the nodal velocities. This implies that the method for calculating tangential velocities is more critical to the overall accuracy of the interpolation. It is also worth mentioning that for method tRT1 the averaged error is relatively small although it suffers from larger local noise due to skewed cells (Fig. 9). This indicates that a different measure may be needed to represent the local errors.

4.3. Interpolation for arbitrary locations

In Sections 4.1 and 4.2, we outlined methods for obtaining the nodal and tangential velocities, which are defined at specific locations on the mesh. In this section, we describe methods to interpolate the velocity vectors at any location in the domain, which is essential to obtain the trajectories and $\mathbf{u}^-$ for semi-Lagrangian advection. The interpolation techniques follow the linear and quadratic interpolation methods by Walters et al. (2007). The velocity vector at an arbitrary point $\mathbf{x}$ located in cell i can be obtained by linearly combining the nodal velocities of cell i or quadratically combining the velocities at the nodes and edge midpoints in cell i. We also include a four-cell based linear interpolation that is not discussed by Walters et al. (2007).

We require that the interpolation methods smoothly recover the normal velocity component U at edge midpoints. This is necessary for temporal convergence at small Courant numbers, which may occur in simulations with grid spacing varying by orders of magnitude and the time step is chosen for the stability and/or accuracy on the finest grid. The condition is also important in order to avoid erroneous acceleration and predict correct structures such as recirculation, stagnation point, etc. As a result, only the low-order local nodal velocities obtained with method nRT1 can be used for the linear interpolation. The quadratic interpolation, on the other hand, naturally converges because it uses the midpoint velocity vector that has the exact normal component. Because the nodal velocities from methods nP1, nRT2 and nLS are very similar (Fig. 7), in the following discussion we use method nRT2 for the nodal velocities and employ methods tRT1, tRT2 and tLS for the tangential velocities. Methods based on tP1 and tP2 are not employed because they perform similarly and tend to be more sensitive to mesh quality. Therefore, the following five methods are considered for interpolating the velocity vector at location $\mathbf{x}$, and their respective stencils are illustrated in Fig. 11.

1. (L1nRT1) Trilinear interpolation in one cell using the local nodal velocities obtained with method nRT1. Using the notation shown in Fig. 11(a), the velocity vector at point $\mathbf{x}$ in cell i can be written as (Huebner, 1975)

$$\mathbf{u}(\mathbf{x}) = A_{23}\mathbf{u}_{p,1} + A_{13}\mathbf{u}_{p,2} + A_{12}\mathbf{u}_{p,3}, \tag{11}$$

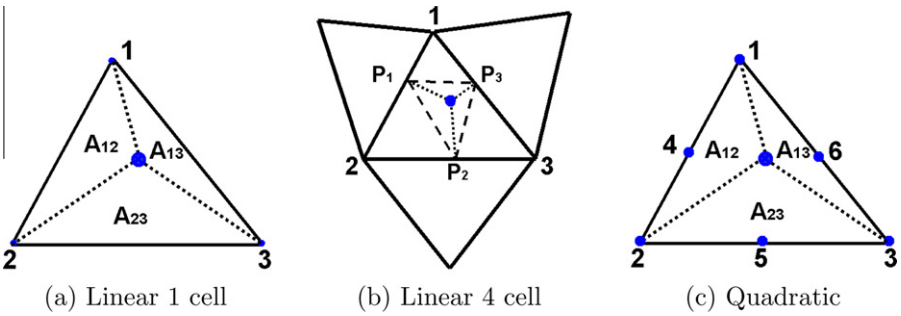

Fig. 11. Stencils for the linear and quadratic interpolation for point $\mathbf{x}$ in cell i, denoted by the filled circle in each triangle. A_{mn} is the area of the triangle formed by point $\mathbf{x}$ and two nodes normalized by the total area of cell i. Points P_1, P_2 and P_3 are the perpendicular feet from the point $\mathbf{x}$ to the edges of cell i.

where $\mathbf{u}_{p,1}$, $\mathbf{u}_{p,2}$, and $\mathbf{u}_{p,3}$ are the velocity vectors at the three nodes of the triangle, and A_{mn} is the area of the triangle formed by point $\mathbf{x}$ and the mth and nth node of the cell i, and normalized by the area of cell i (Fig. 11(a)). Variables A_{mn} are area coordinates, and the interpolating function of order n contains a complete polynomial of order n that is a Lagrange polynomial from a Pascal triangle.

2. (L4nRT1) Linear interpolation using the local nodal velocities in a four-cell stencil. As illustrated in Fig. 11(b), the method first requires the calculation of the perpendicular feet from point $\mathbf{x}$ to each edge of cell i, i.e., P_1, P_2 and P_3. The velocity vectors at perpendicular feet are computed with method L1nRT1, which are then interpolated to point $\mathbf{x}$ using trilinear interpolation in the triangle formed by P_1, P_2 and P_3 (Fig. 11(b)). This method guarantees convergence to the normal velocity when $\mathbf{x}$ coincides with the midpoint of an edge.
3. (QnRT2tRT1) Quadratic interpolation using global nodal velocities from method nRT2 and tangential velocities from method tRT1. The interpolation coefficients can be found explicitly with the area coordinates and thus do not necessarily require matrix inversion. Using the notation in Fig. 11(c), the quadratic interpolation for point $\mathbf{x}$ in cell i can be written as (Huebner, 1975)

$$\begin{aligned}\mathbf{u}(x,y) = {} & (2A_{23}-1)A_{23}\mathbf{u}_{p,1} + (2A_{13}-1)A_{13}\mathbf{u}_{p,2} \\ & + (2A_{12}-1)A_{12}\mathbf{u}_{p,3} + 4A_{13}A_{23}\mathbf{u}_{e,4} + 4A_{12}A_{13}\mathbf{u}_{e,5} \\ & + 4A_{12}A_{23}\mathbf{u}_{e,6}, \end{aligned} \tag{12}$$

where $\mathbf{u}_{p,1}$, $\mathbf{u}_{p,2}$, and $\mathbf{u}_{p,3}$ are the nodal velocity vectors and $\mathbf{u}_{e,4}$, $\mathbf{u}_{e,5}$, and $\mathbf{u}_{e,6}$ are the midpoint velocity vectors on the edges calculated with $\mathbf{u}_e = U\mathbf{n} + V\mathbf{t}$, where U is known and V is interpolated. Walters et al. (2007) implemented this particular method.
4. (QnRT2tRT2) Quadratic interpolation using Eq. (12) and method nRT2 for nodal velocities and method tRT2 for tangential velocities (stencil in Fig. 11(c)).
5. (QnRT2tLS) Quadratic interpolation using Eq. (12) and method nRT2 for nodal velocities and method tLS for tangential velocities (stencil in Fig. 11(c)).

We apply these methods to interpolate the velocity vectors onto an 80×40 Cartesian array of points uniformly distributed over the unstructured meshes. Among the five methods above, the quadratic methods are accurate for linear velocity fields on the equilateral mesh at points away from the boundaries. The linear methods L1nRT1 and L4nRT1 are only accurate for constant velocity fields, and for linear velocity fields the error for method L4nRT1 can be one order of magnitude smaller than that for method L1nRT1. Fig. 12 illustrates the errors in the streamwise velocity component for the parabolic flow field (Fig. 3) on the test meshes (Fig. 4). For a more quantitative comparison, the error is averaged in the x-direction to obtain an average error as a function of y on the equilateral mesh as shown in Fig. 13. The errors for method L1nRT1 reach 10^{-1} and alternate in the y direction with relatively small variation in the x direction. Method L4nRT1 has errors of similar magnitude which are, however, more randomly distributed, and thus the x-averaged error is reduced (below 10^{-2}) as shown in Fig. 13. The quadratic methods have errors on the order 10^{-3} that are mostly negative except for the points in the boundary cells. The negative sign corresponds to positive numerical diffusion when the interpolation is used to obtain $\mathbf{u}^-$ in semi-Lagrangian advection (Eq. (6)), and the errors alternate in the y direction with relatively small variation in x. Method QnRT2tRT2 has larger but more uniform errors than method QnRT2tRT1, and method QnRT2tLS gives intermediate accuracy and smoothness.

On the skewed mesh, none of the results are severely impacted because we have chosen robust methods to calculate the nodal and tangential velocities. However, it is evident that the error varies spatially depending on cell alignment. Among the quadratic methods, method QnRT2tRT1 is relatively more sensitive to the mesh quality and the skewed cells induce positive errors with magnitudes of up to 10^{-2}.

Spatial convergence for the interpolation methods using the parabolic velocity field is shown in Fig. 14. The y-axis is the average absolute value of the error in the streamwise velocity evaluated on the Cartesian array of points within the rectangle defined by $x_{max}/8 \leqslant x \leqslant 7x_{max}/8$ and $y_{max}/8 \leqslant y \leqslant 7y_{max}/8$. As expected, the linear interpolation methods converge with a slope of one and the quadratic methods converge with a slope of two with mesh refinement, and thus are first- and second-order accurate, respectively, in space. Although method L4RT1 is slightly more accurate than method L1RT1, the errors for the linear methods are one to two orders of magnitude larger than those for the quadratic methods. The accuracy of the quadratic methods is consistent with the accuracy of their tangential velocities. Method QnRT2tRT1 shows higher sensitivity to mesh quality at finer resolutions, while the error for method QnRT2tRT2 roughly follows $0.1u_{yy}\Delta x^2$.

4.4. Combined trajectory and interpolation errors

For semi-Lagrangian advection, to obtain the location of the traceback, $\mathbf{x}_j^-$, trajectories are calculated by integrating velocity vectors backward in time from the midpoint of edge j, where the normal velocity, U_j, is defined. Errors arise from two sources, namely (1) the temporal integration of the trajectory (Eq. (7)) and (2) the interpolation of the velocity vectors. The relative contribution of the two sources of error depends on the particular integration and interpolation methods, velocity field, resolution and the time step size. To compare different methods, we prescribe the velocity field $u = 6y - 6y^2$ and $v = 1.5x - 0.5x^2$ on the equilateral mesh shown in Fig. 4, which produces curved trajectories. For different time step sizes, we use the equilateral mesh and compute trajectories that originate at the edges, and then interpolate the velocity field at the end of each trajectory, $\mathbf{u}_j^-$.

The exact trajectories can be calculated using Eq. (7) with the prescribed velocity field, and we integrate these numerically using RK4 with a tolerance of 10^{-10}. Two types of trajectory integration methods are employed, namely, the explicit Euler (EE) method (Eq. (8)), and what we refer to as the analytical method. These are then combined with different interpolation methods and with and without substepping. In our tests, we only consider using the same interpolation to calculate the trajectory and the final U_j^- at the end point, although it is possible to vary the interpolation methods in these two steps. For the EE trajectory integration, we compare the results of using Eq. (8) with one step versus with five substeps. The velocity vector is interpolated (the exact velocity is used in one case for comparison) at the end of each substep, and used for the integration of the next substep. Analytical trajectories eliminate time-integration errors by integrating the interpolated velocity field exactly, and they differ from the "exact trajectories" only because the latter is integrated using the true velocity field. To compute the analytical trajectories, the area coordinates for the linear and quadratic interpolation (Eqs. (11) and (12)) are converted into regular polynomial form and time-integration of Eq. (7) is performed using RK4 with a tolerance of 10^{-10} with the velocity field described by these polynomials. When substepping is used with the EE method, the velocity vector at the end of the substep is always interpolated using the cell in which the trajectory is located at that substep. Therefore, if the trajectories cross over triangle edges into a different cell, the velocity vector is obtained in this new cell to use for the next substep. For the analytical integration, we evaluate two similar scenarios: (1) the velocity vector is interpolated using the coefficients from the triangle abutting the original departure edge j and remains the same for the integration over

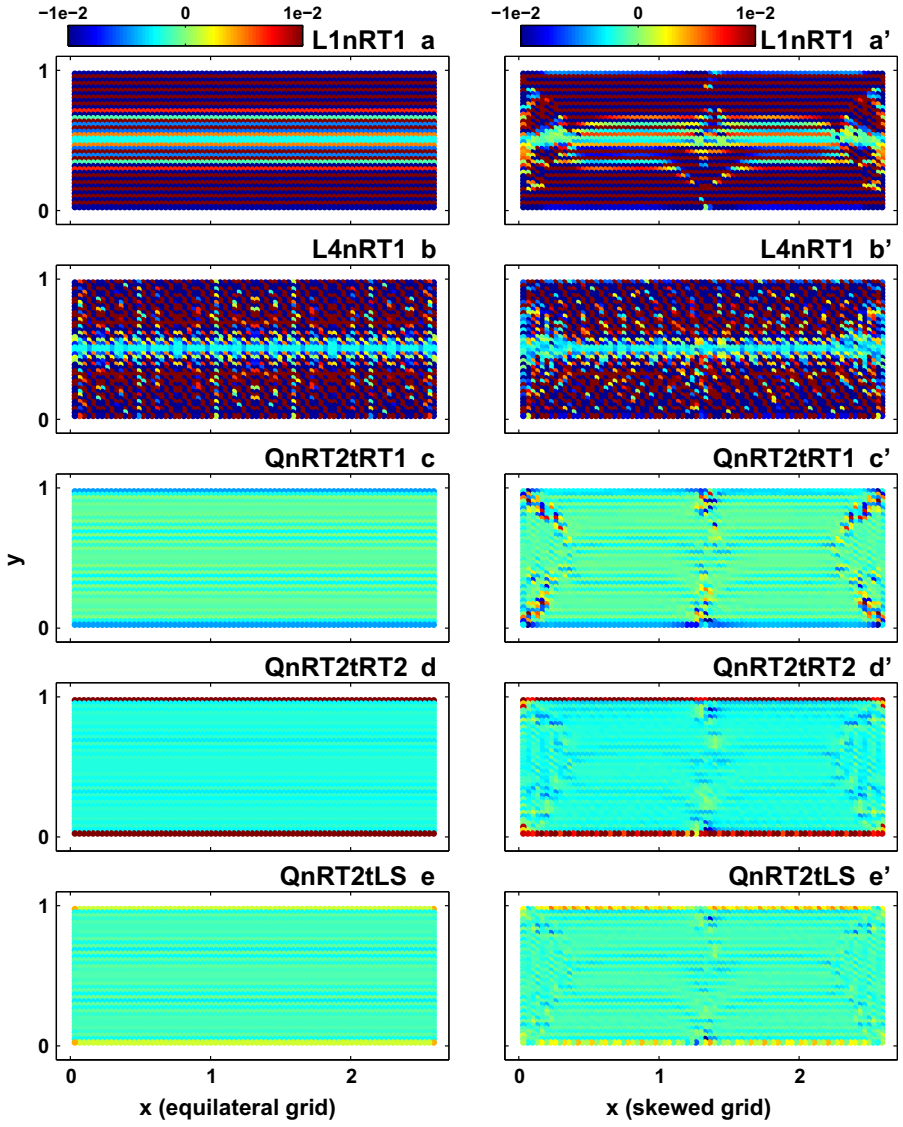

Fig. 12. Errors in the interpolated velocities for the quadratic velocity field on the equilateral mesh (left) and skewed mesh (right) for methods L1nRT1, L4nRT1, QnRT2tRT1, QnRT2tRT2 and QnRT2tLS.

the entire time step, and (2) to test the effect of using the local velocity field (different coefficients), we compute analytical trajectories over five intervals, and the coefficients are updated at the end of each interval to ensure that they correspond to the local coefficients. Here we use the term "interval" to emphasize that the analytical trajectory method computes curved trajectories during each substep. A trajectory for the analytical method will be smooth from one interval to the next unless it crosses over an edge, and if it crosses over an edge, it is still continuous but there will be a discontinuity in direction due to a change in the interpolated velocity field. We combine the integration methods with two interpolation methods L4nRT1 and QnRT2tRT2, and a total of nine different combinations of trajectory integration/interpolation methods are employed, as outlined in Table 2.

Fig. 15 illustrates the errors as a function of the Courant number in $\mathbf{x}_j^-$ and $\mathbf{u}_j^-$ using the combinations of methods listed in Table 2. Errors are evaluated as the average of the errors associated with trajectories that originate at edges within the rectangle defined by $x_{max}/8 \leqslant x \leqslant 7x_{max}/8$ and $y_{max}/8 \leqslant y \leqslant 7y_{max}/8$ and remain in the domain after the time interval Δt. The results of $\mathbf{x}_j^-$ are depicted in Fig. 15(a), which shows that the error associated with integration using EE in one step with the exact velocity (Combinations 2 and 5 in Table 2) converges as a straight line with a slope of two. This is consistent with its theoretical leading error of $O(\Delta t^2)$ in one time step. However, when interpolated velocities are used with EE integration in one step (Combinations 3 and 6 in Table 2), results converge at a slower rate at small Courant numbers and the error with interpolation method L4nRT1 (Combination 3) is considerably larger than that with method QnRT2tRT2 (Combination 4). The error due to interpolation of the velocity field, $\mathbf{u}^e(\mathbf{x})$, contributes errors in calculating $\mathbf{x}_j^-$ of the form $\mathbf{u}^e\Delta t + H.O.T.$, which converges with a slope of unity at small Δt. This error in the trajectory associated with using EE in one step is considerably reduced by using the analytical trajectory integration with the quadratic coefficients (Combination 7). However, when the Courant number, defined by $u_{max}\Delta t/\Delta x$, exceeds 0.7, the error increases rapidly because some of the tracebacks enter neighboring cells where the original quadratic coefficients for the analytical integration are no longer valid. The coefficients obtained from the quadratic interpolation are not necessarily close to the prescribed velocity field and can differ significantly from one cell to another. When the traceback is within or very close to the cell

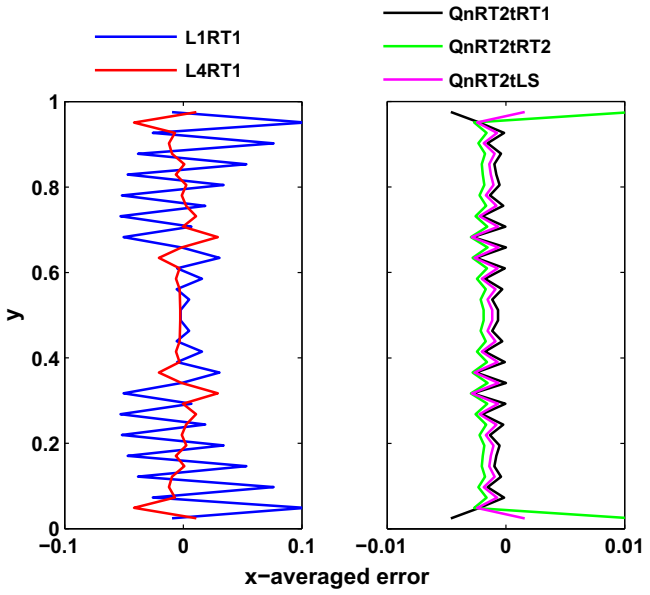

Fig. 13. Errors in the interpolated velocities averaged in the x direction for the quadratic velocity field on the equilateral mesh for methods L1nRT1, L4nRT1, QnRT2tRT1 and QnRT2tRT2.

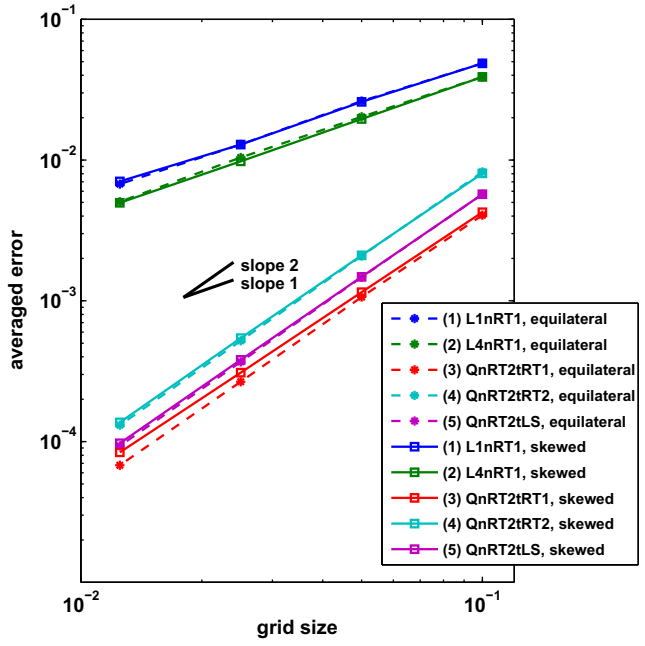

Fig. 14. Spatial convergence of the interpolation methods for the quadratic velocity field. The y-axis is the absolute value of the errors averaged over the edges within the region $x_{max}/8 \leqslant x \leqslant 7x_{max}/8$ and $y_{max}/8 \leqslant y \leqslant 7y_{max}/8$.

containing the edge from which the traceback departed, the interpolation error is bounded. Otherwise, interpolation error grows quickly with distance from the cell. Therefore, it is important to interpolate the velocity field using coefficients from the cell that encompasses the traceback. This can be done either every time the traceback crosses an edge or with intervals at a sufficient frequency. We choose the latter and use 5 intervals for comparison to the errors associated with the EE method that uses 5 substeps. As shown in Fig. 15(a), substepping produces higher accuracy when the quadratic interpolation is used for both the EE integration (Combination 8) and the analytical integration with substeps (Combination 9).

Table 2
Combinations of interpolation and trajectory calculation methods.

Combination	Interpolation	Trajectory method	
		Integration	Velocity field
1	L4nRT1	Exact	Exact
2	L4nRT1	EE one step	Exact
3	L4nRT1	EE one step	Interpolated
4	QnRT2tRT2	Exact	Exact
5	QnRT2tRT2	EE one step	Exact
6	QnRT2tRT2	EE one step	Interpolated
7	QnRT2tRT2	Analytical one interval	Departure cell coefficients
8	QnRT2tRT2	EE five substeps	Interpolated
9	QnRT2tRT2	Analytical five intervals	Local coefficients

Fig. 15(b) shows the errors in the final values of U_j^- (the normal component of $\mathbf{u}_j^-$) interpolated at location $\mathbf{x}_j^-$. In general, since decreasing the time step size leads to a traceback location that is closer to the departure point $\mathbf{x}_j$, the normal component $U_j^- = \mathbf{n}_j \cdot \mathbf{u}_j^-$ converges to the exact value with decreased time step because both the linear and quadratic interpolation methods give the exact value of the normal velocity when $\mathbf{x}_j^- = \mathbf{x}_j$. Although the tangential component $V_j^- = \mathbf{t}_j \cdot \mathbf{u}_j^-$ converges to the interpolated value rather than the exact value which is not unknown in general, this does not affect the results since V_j^- is not used in the remaining calculations.

Different trajectory integration methods do not affect the accuracy of Combinations 1–3 because the error due to the linear interpolation method L4nRT1 dominates over the error in the trajectory integration. This is in contrast to the errors in Combinations 4–9 which employ quadratic interpolation. For Combinations 4–9, when the trajectories are integrated with substeps or intervals (Combinations 8–9), errors are similar to the error that arises when using the exact $\mathbf{x}_j^-$ (Combination 4). When intervals or substepping is not employed, the errors grow rapidly for Courant numbers greater than unity (Combinations 5–7). At small Courant numbers, the results converge at a slightly sub-linear rate (i.e., $<O(\Delta t)$) for the linear interpolation Combinations (1–3) and a slightly super-linear rate for the quadratic interpolation Combinations (4–9). For large Courant numbers, interpolation errors are generally bounded unless the mesh is highly skewed. This is key to the stability of the semi-Lagrangian method. However, unlike structured grids, interpolation errors rarely vanish at integer Courant numbers because it is unlikely that the traceback coincides with an edge midpoint that has the same orientation as edge j. On the other hand, trajectory integration errors may continue to grow with increased time step size at large Courant numbers beyond the range that is discussed here, and thus are likely to play a more important role. This is consistent with the findings by Xiu and Karniadakis (2001) which indicate the dominance of interpolation error for small time step sizes.

5. Test cases

We compare the different interpolation methods with two test cases. The first test case consists of flow over a backward-facing step with idealized geometry, which is a well-documented benchmark test case for momentum advection schemes. The second test case consists of a field-scale simulation of estuarine flow in a shallow, macro-tidal estuary. For both cases the trajectory is integrated using the EE method with three substeps for simplicity and efficiency. More sophisticated trajectory integration methods can be

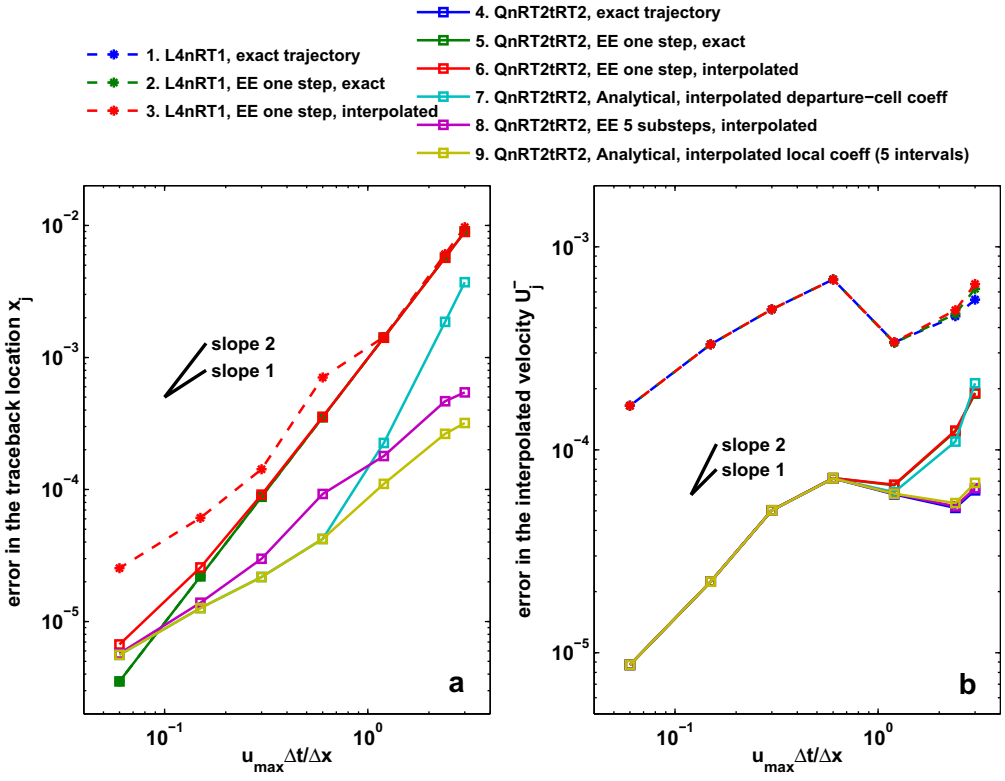

Fig. 15. Errors in computing the traceback location $\mathbf{x}_j^-$ (a) and the normal velocity at the traceback, U_j^- (b), as a function of the Courant number on the equilateral mesh (Fig. 4(a)). Combinations 1–9 are described in Table 2.

found in Walters et al. (2007) and Ham et al. (2006). To ensure temporal accuracy in resolving advection of momentum, the maximum horizontal Courant number we use is typically around 0.1–0.3.

5.1. Backward-facing step

The rectangular domain for the backward-facing step test case is shown in Fig. 16. The domain is $L = 40$ m long and $2H = 2$ m high with no-slip upper and lower walls, and is chosen to be long enough to minimize possible influence of the outflow boundary conditions. The inlet and the step height, H, are both 1 m, and, following the configuration of Gartling (1990), the inlet does not include an extended inflow channel which helps avoid skewed cells at the step corner. Two-dimensional rigid-lid simulations were performed using SUNTANS, and the unstructured mesh shown in Fig. 16 is generated with GAMBIT (Fluent, Inc., Lebanon, NH) and has an average angle skewness of roughly 4°. The resolution is 0.05 m near the inlet and is gradually stretched to roughly 0.06 m at $x = 20$ m and 0.1 m at the outlet boundary, which is similar to the resolutions used by other authors (Gartling, 1990; Vidović et al., 2004). Multiple resolutions were tested to ensure convergence, although these results are omitted here. At the inlet boundary, a parabolic velocity profile is specified, such that $u(x = 0, z, t = 0) = -6u_{ave}(1 - z/H)(2 - z/H)$ for $H \leqslant z \leqslant 2H$, which gives an average inflow velocity of $u_{ave} = 1$ m s^{-1}. With a constant viscosity $\nu = 2.5 \times 10^{-3}$ m^2 s^{-1}, the Reynolds number based on the channel width is given by $Re = 2u_{ave}H/\nu = 800$. The Reynolds

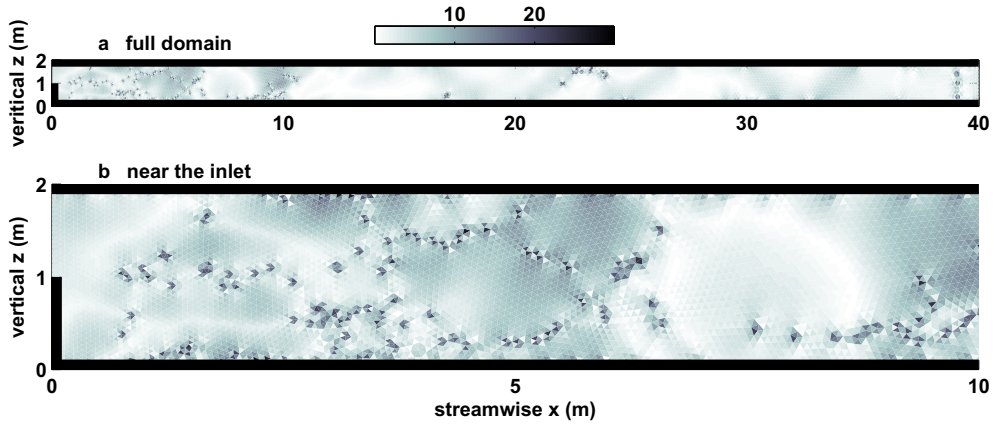

Fig. 16. Computational mesh for the backward-facing step test case. The gray scale shows the angle skewness in degrees.

number definition and problem specifications follow Armaly et al. (1983) and Gartling (1990).

We examine the backward-face step flow with four interpolation methods for semi-Lagrangian advection, namely L1nRT1, L4nRT1, QnRT2tRT1, and QnRT2tRT2, which are discussed in Section 4.3. Method QnRT2tLS is no longer included due to its low cost-effectiveness. Two Eulerian schemes, namely, first-order upwind and central differencing, are also included in the comparison. These Eulerian schemes are based on the momentum-conservative methods from Perot (2000), and are the original advection methods in the SUNTANS model. The stationary velocity fields obtained from the simulations are illustrated in Fig. 17 in order of decreasing primary reattachment length at the lower wall, which we refer to as x_r. In these results, the final velocity field on the staggered grid was interpolated to vertical transects using interpolation method QnRT2tRT1. The semi-Lagrangian method with QnRT2tRT1 interpolation predicts a reattachment length of roughly 12 step-heights downstream of the step at the lower wall, and an upper wall eddy extending between 10 and 21.5 step-heights from the step, which are close to the results of Gartling (1990), who found $x_r \approx 12H$. The results of the Eulerian central-differencing scheme are similar but the recirculation cells are slightly weaker. Although Eulerian central-differencing on Cartesian grids is non-diffusive in theory, mesh skewness may introduce diffusive errors on unstructured grids.

Overall, method QnRT2tRT1 and Eulerian central-differencing predict reasonable flow fields. Fig. 18 depicts the velocity and pressure variations from these two methods at transects located at $x/x_r = 14/12$ and $x/x_r = 30/12$, which correspond to transects at $x/H = 14$ and $x/H = 30$, locations of data published by Gartling (1990). We normalize the distance with the reattachment length here to facilitate interpretation of results. The comparison shows that the general structure of the flow is captured reasonably well, although errors in the vertical velocity are relatively large. We found that the profile of vertical velocity is quite sensitive to the horizontal location in the streamwise direction and is difficult to reproduce (also see Vidović et al., 2004).

The other methods have substantial errors and do not predict correct flow structures. They predict much weaker (or no) primary lower-wall circulation and upper-wall eddy. The semi-Lagrangian scheme with the QnRT2tRT2 interpolation gives a similar velocity field as the first-order upwind Eulerian scheme and appears to be slightly more diffusive. The marked difference between the semi-Lagrangian schemes with QnRT2tRT2 and QnRT2tRT1 implies that the choice of the tangential velocity in the interpolation is important. The former is more diffusive as expected because its tangential estimate is more dissipative. The semi-Lagrangian method with linear interpolation L4nRT1 predicts very weak recirculation at the lower wall while interpolation L1nRT1 gives no recirculation and the flow field is similar to that for the no-advection case (no momentum advection).

Predicted streamwise distributions of the pressure at the lower wall within 20 step-heights from the step are depicted in Fig. 19

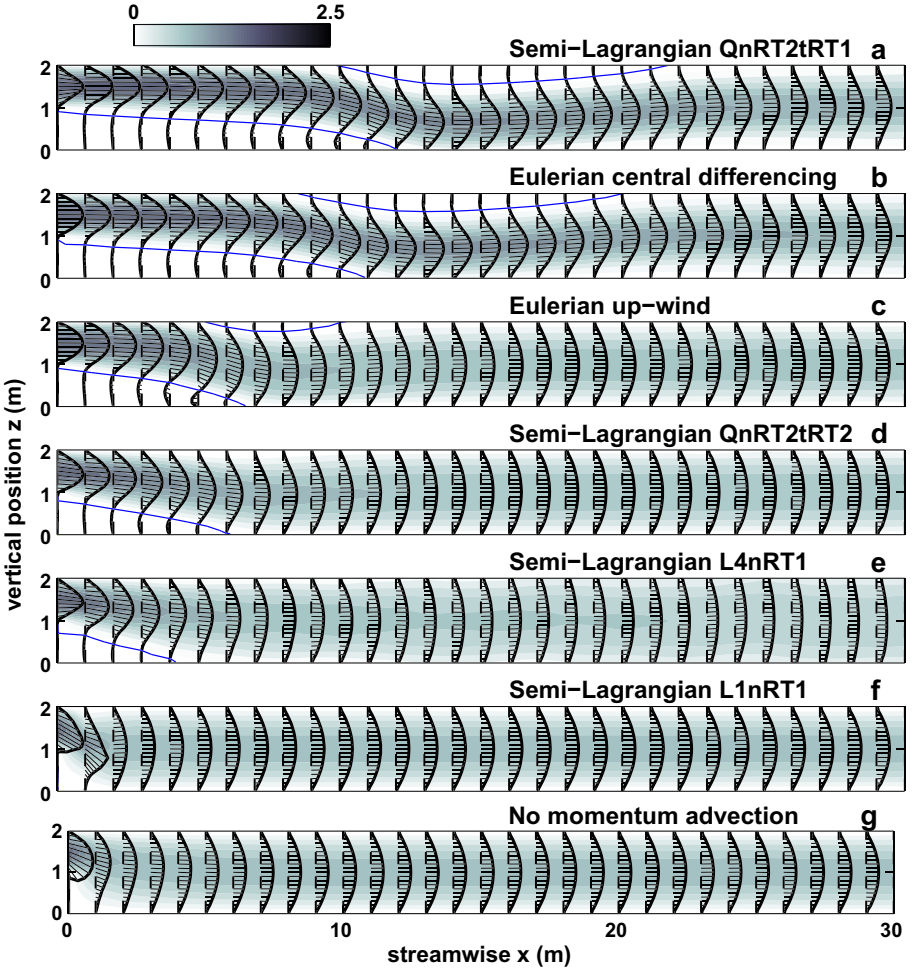

Fig. 17. Velocity profiles (in m s^{-1}) and recirculation zones predicted with different methods, in order of decreasing reattachment distance. The Reynolds number based on the step height is $Re = 800$.

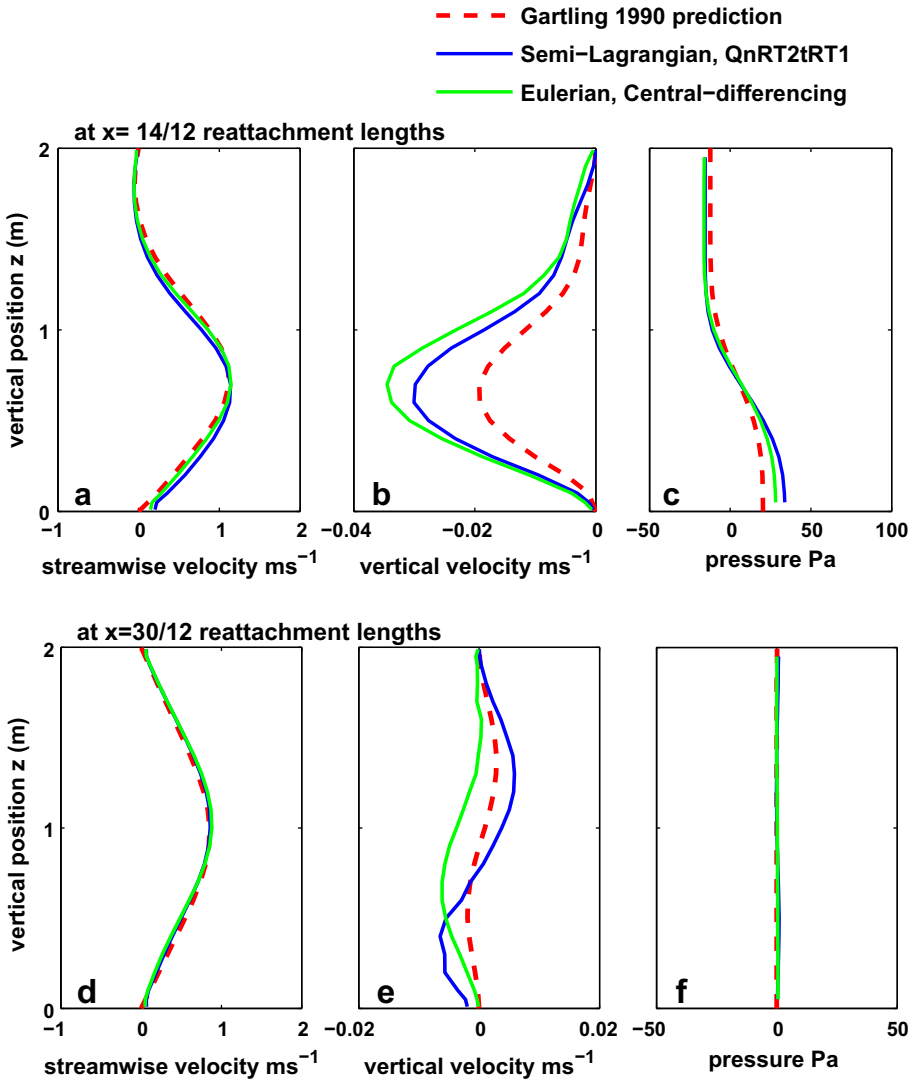

Fig. 18. Streamwise velocity and pressure profiles at $x/x_r = 14/12$ and $x/x_r = 30/12$ with the Eulerian central-differencing scheme and the semi-Lagrangian method with interpolation method QnRT2tRT1. The pressure is relative to the mean at each transect to emphasize vertical variability.

and compared to the results of Gartling (1990). Physically, pressure should increase slowly as the flow decelerates over the step and it should experience a local maximum at the lower-wall reattachment point. The results of the semi-Lagrangian scheme with QnRT2tRT1 and the Eulerian central-differencing are roughly correct. However, for the other cases, numerical errors cause significant momentum loss which leads to a steady pressure drop with distance down the channel. This is most significant for the semi-Lagrangian method with L4nRT1 interpolation in which the velocity profile in the downstream section is considerably flatter than the expected parabolic profile. The semi-Lagrangian method with L1nRT1 interpolation produces a pressure distribution that is similar to that for the no-advection case. Although incorrect, the no-advection case is non-dissipative, a result that was also shown by Walters et al. (2007).

5.2. Field-scale estuarine test case

We compare the advection schemes with field-scale simulations of the flows in the Snohomish River estuary, WA, to assess their ability to predict both large-scale tidal flows and local-scale flow-bathymetry interaction at an abrupt sill. The Snohomish River estuary is the lower mainstem of the river defined by Jetty Island to the west and the city of Everett to the east (see Fig. 20). The estuary

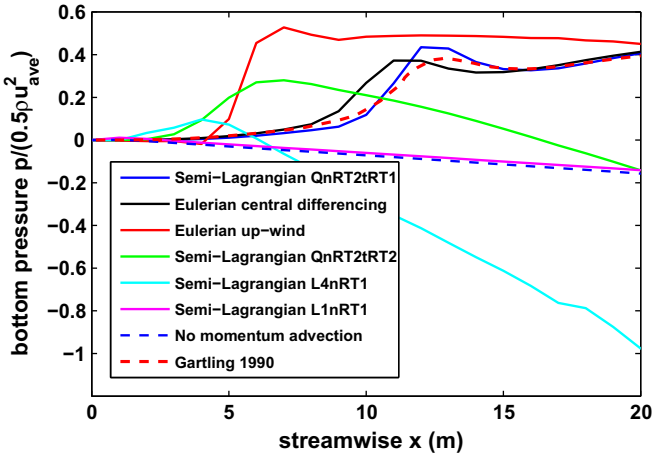

Fig. 19. Streamwise pressure distribution at the lower wall predicted with different methods. Pressure is relative to the pressure at the lower corner of the step at $x = 0$ and normalized by $0.5\rho u_{ave}^2$, where $u_{ave} = 1$ m s^{-1}.

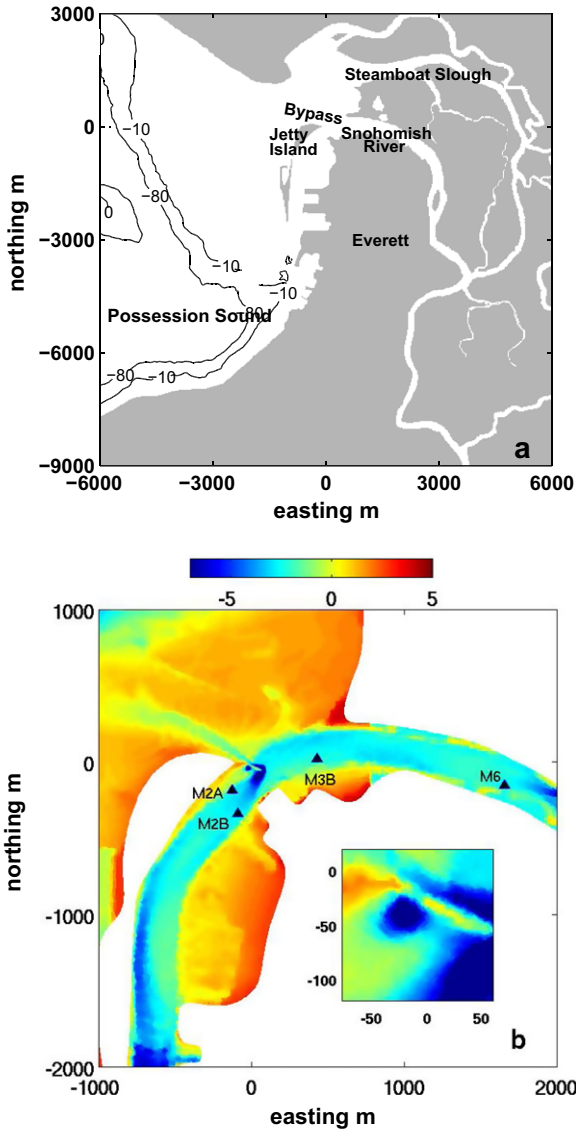

Fig. 20. (a) Map of the Snohomish River estuary, and (b) the bathymetry in m above MLLW (mean low–low water). The bathymetry is shown as a zoomed-in view around the northern tip of Jetty Island. The inset plot in (b) shows the steep man-made sill that causes flow separation and recirculation. Mooring locations are indicated by the dark filled triangles in (b).

is a shallow tidal river with less than 5 m mean depth and extensive intertidal mudflats, and it is influenced by strong tides with over 4 m tidal range. The river discharges to Possession Sound offshore, which is one of the basins in Puget Sound. A bypass connection between the main channel and Possession Sound exists at the north end of Jetty Island, which consists of intertidal mudflats that are exposed at low tide. The tidal flows interact with the complex geometry and result in strong spatio-temporal variability in the tidal flows during each tidal cycle. Furthermore, a highly dynamic local flow field is present in the vicinity of a small (50 m × 10 m), abrupt man-made sill and the scourholes at the tip of Jetty Island (see the inset plot in Fig. 20(b)).

The computational domain includes the main river channel and a simplified tributary channel, and the offshore region encompasses Possession Sound to the west and Port Susan Bay to the north. Two meshes with different horizontal resolutions (which we refer to as coarse and fine) are applied to evaluate the predictions of large-scale tidal flows and local-scale bathymetry-induced flows. The coarse mesh has a horizontal resolution of 8–35 m (edge length) to resolve flow in the main river channels, while stretching is employing to expand the resolution to 300 m in Possession Sound. Vertical layers are 0.2–0.3 m for the top 20 m that cover the estuary and are stretched at a rate of 1.19 (ratio of neighboring layer thickness) in the deeper Possession Sound regions. The vertical grid consists of structured z-levels, and linear interpolation is used in the vertical for the semi-Lagrangian advection, while the different horizontal interpolation methods discussed in this paper are tested. Tidal fluxes are forced at the west inlets of Possession Sound and are calibrated to give the correct offshore water level. Bottom drag is parameterized with a quadratic drag law using a roughness height of 2.5×10^{-4} m as inferred from observations. All simulations use a zero horizontal eddy-viscosity and are run with a time step of 1 s and a maximum horizontal Courant number of roughly 0.1 (based on edge length) for resolving advection and wetting and drying. A two-week long simulation is performed with the coarse grid to validate the model predicted tidal dynamics for a spring-neap tidal cycle (although in this paper we only discuss the results over one day). Specific details of bathymetry, boundary conditions, initial conditions and validation can be found in Wang et al. (2009) (in which method L4nRt1 was employed). It is worth mentioning that the bathymetry is improved by incorporating recent survey data for a 6-km stretch of the main channel upstream at the junction of the tributary. This improved bathymetry effectively damps out the peak flows and oscillation in the no-advection case shown in Wang et al. (2009). In what follows, we compare the predictions from three different interpolation schemes for semi-Lagrangian advection, namely method L4nRT1, QnRT2tRT1, and QnRt2tRT2, along with the no-advection case. The Eulerian schemes are not suitable for this test case because of wetting and drying, and the test case with interpolation method L1nRT1 is also excluded because it can be unstable and cannot be stabilized

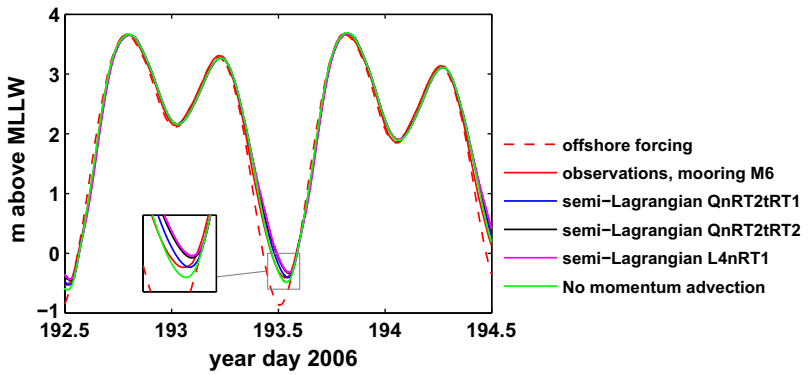

Fig. 21. Predicted and measured free surface, relative to MLLW (mean low–low water), at mooring location M6. Observations are from Giddings et al. (2011).

with horizontal viscosity (the instability may arise from local grid skewness which we have not specifically analyzed).

The tidal forcing consists of strong diurnal and semi-diurnal constituents, and thus there is a strong ebb/flood period followed by a weak ebb/flood period. Fig. 21 depicts the free-surface predictions at mooring site M6, the location of which is shown in Fig. 20(b). The predictions from the simulations are similar with noticeable differences near low–low water, when the water level measured at mooring site M6 is higher than that offshore in the Sound due to inertial and frictional effects. Because numerical diffusion associated with momentum advection can lead to increased dissipation, methods with more numerical diffusion will lead to a higher water level. Fig. 21 shows that the predictions from methods L4nRT1 and QnRT2tRT2 overpredict the water level during low–low water, and thus the tidal waves are slightly over-damped. The result of method QnRT2tRT1, on the other hand, has roughly the correct amplitude and so it likely has the least numerical diffusion. The non-advective case is the only case that the water level is underpredicted during low–low water due to a lack of inertial and frictional effects associated with momentum advection.

Fig. 22 shows time series of depth-averaged along-channel velocity at the mooring sites in the main channel (Fig. 20(b)) from the simulations in comparison with field observations. Positive velocity is defined in the upstream direction (flood tide direction). There are notable differences between the simulations, most prominently during peak ebb events with discrepancies up to 0.2 m s^{-1}, although the general pattern is similar. These peak ebb flows occur around the time of low–low water, when the predictions also deviate from the observations most significantly. This is the most challenging period to simulate accurately because the flow is very shallow (2 m) and fast (over 1.5 m s^{-1} near the free surface) and possesses shallow drying channel shoals, all of which lead to strong spatial gradients. Among the three semi-Lagrangian methods, interpolation L4nRT1 is always the most dissipative, QnRT2tRT1 is the least, and QnRT2tRT2 is intermediate, which is consistent with our earlier analysis. Overall, quadratic interpolation produces the best match between predictions and observations. Although details in the behavior of the results vary from location to location, it is evident that the non-advective case consistently over-predicts peak flows, implying the importance and spatial variability of momentum advection.

The near-surface velocity field during peak ebb (around low–low water) is illustrated in Fig. 23. The relative strength of the flow is generally consistent with what was found for the depth-averaged velocity. However, the horizontal variability in the velocity fields among the different methods is quite different. In the no-advection test, slow flow is limited to the shallow shoals and there are sharp lateral gradients between the shoals and the central

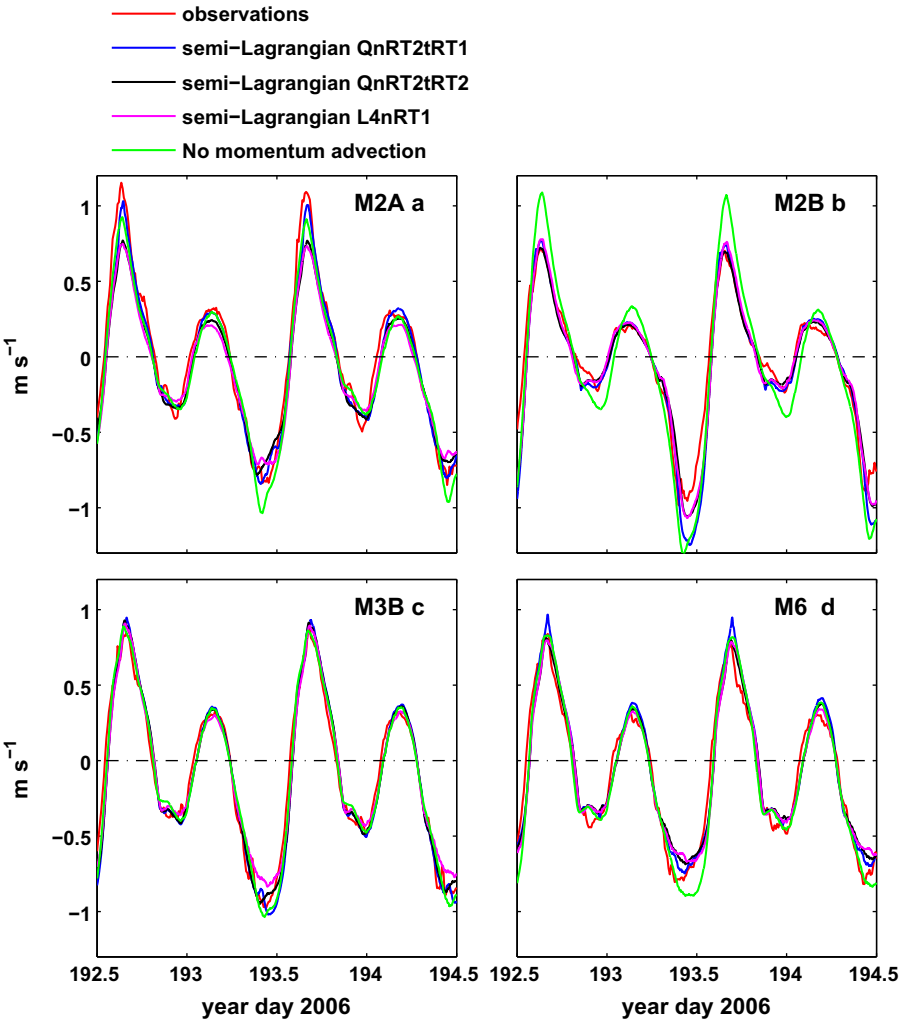

Fig. 22. Predicted and measured depth-averaged along-channel velocity at mooring sites (a) M2A, (b) M3A, (c) M3B and (d) M6. The mooring sites are shown in Fig. 20(b). Positive velocity is defined in the upstream direction (flood tide direction). Observations are from Giddings et al. (2011).

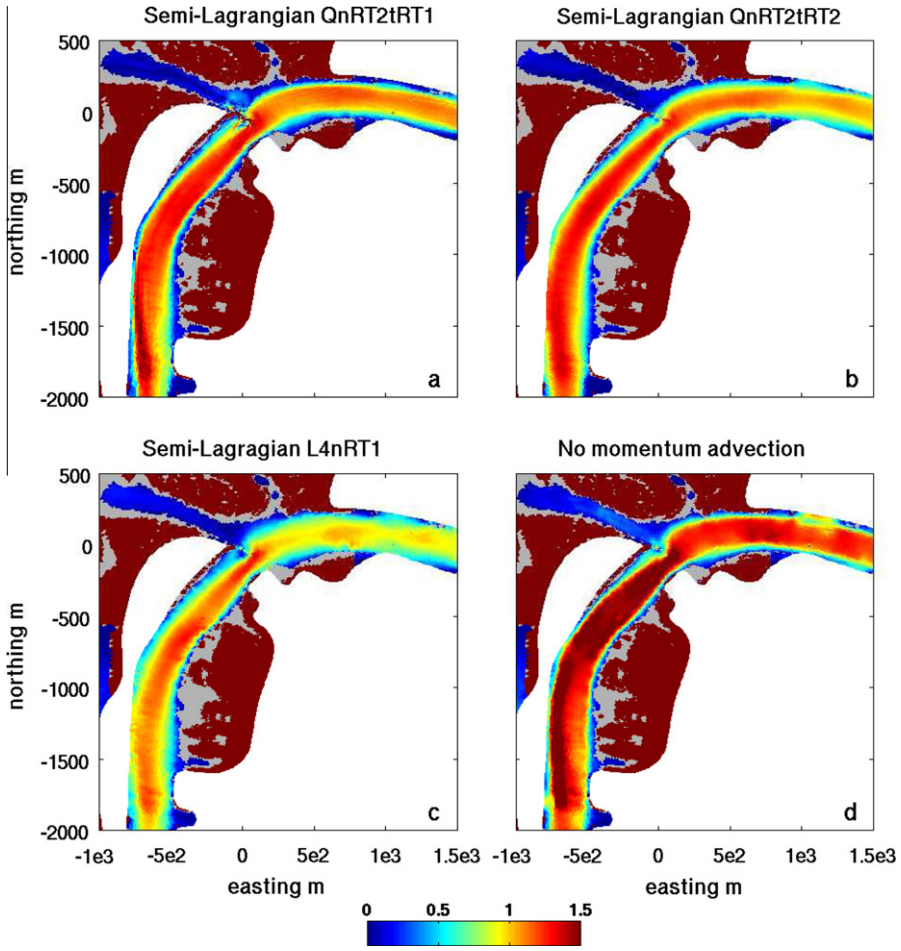

Fig. 23. Predicted magnitude of the near-surface velocity field (m s^{-1}) during peak ebb tide on the coarse mesh. Grey color indicates wet areas where the plotted elevation is below the bed. Dark red color indicates dry or exposed mudflats. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

channel. This flow field is well correlated with the bathymetry (Fig. 20(b)). The advective cases predict much smoother gradients between the shoals and the channel. This can be caused (1) physically by advection of momentum that brings slow fluid from the shoals to the channel, and (2) by the numerical diffusion of the advection scheme. It is evident that interpolation method L4nRT1 induces considerable numerical diffusion that can lead to significant energy loss in the shoals and over-damping of the tidal flow across the channel. The results from the two quadratic interpolations are similar, while method QnRT2tRT1 predicts slightly sharper lateral gradients. However, method QnRT2tRT1 tends to produce non-propagating spatially-oscillatory errors. For example, there are oscillations at the tip of Jetty Island in Fig. 23(a) that might be caused by unresolved sharp gradients.

The fine mesh is employed with 1–5 m (edge length) horizontal resolution near the sill to evaluate the advection schemes on predicting local-scale flow structures around the sill. With stretching the grid resolution expands to 300 m in the Sound, and the inset plot in Fig. 20(b) shows the resolved bathymetry on this grid. The same vertical resolution as the coarse mesh is used and in total the grid contains 12 million computational cells. With a time step size of 0.1 s, 864,000 time steps are required for a 24-h simulation period, which consumes 48,000 cpu-hours using 200 3.0 GHz Intel Woodcrest processors. Simulations were performed using interpolation method QRT2tRT2 until the second peak ebb tide, and then simulations were started from this same condition using different schemes and ran for another 10-min period. This short duration allows local flow features to develop while the adjustment in the large-scale condition is small.

The predicted near-surface velocity fields near the sill on the fine mesh are illustrated in Fig. 24. The crest of the sill is exposed due to the low water level and the flow separates as it moves around the sill. Recirculation eddies are predicted in the flow at the downstream side of the sill, and they are most pronounced with the quadratic interpolation cases QnRT2tRT1 and QnRT2tRT2. These methods predict similar velocity fields which are similar to what is observed in the field (method QRt2tRT1 was used for the model-observation comparison in Plant et al. (2009)). The linear interpolation method L4nRT1 predicts much weaker recirculation as well as slightly weaker ambient ebbing flow in the channel. In the no-advection case, the flow no longer recirculates, which is consistent with the results of the idealized backward-facing step test case. Although method QnRT2tRT1 predicts stronger recirculation, oscillations in the velocity field at the north side of the sill are also generated with this resolution. These are stable over the course of this ten-minute simulation although they become unstable over a longer period unless additional horizontal viscosity is applied. For the simulations on both coarse and fine meshes, the quadratic interpolation methods incur roughly 15% more computational time than the linear interpolation method L4nRT1.

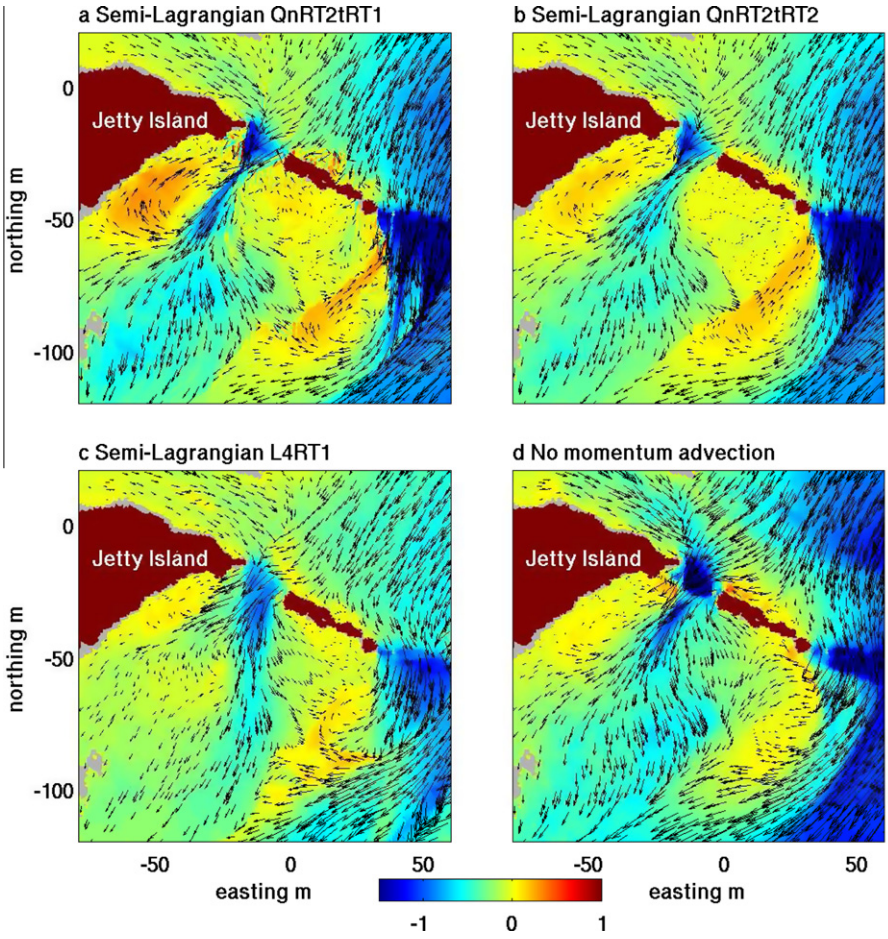

Fig. 24. Predicted velocity field around the sill during peak ebb using different interpolation schemes: (a) QnRT2tRT1, (b) QnRT2tRT2, (c) L4RT1, and (d) no advection of momentum. Color shows the magnitude of the northward velocity (m s^{-1}). Dark red regions are dry or exposed mudflats. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

6. Conclusions

Reconstruction of vector fields on unstructured, staggered, triangular grids is difficult because the governing equations are written for the normal velocity components and the tangential velocity components are not stored. As a result, methods must be developed to obtain the tangential velocity components from the known normal velocity components. In this paper we presented and analyzed five methods to approximate the nodal velocity vectors at cell vertices and five methods to approximate the tangential velocity components at the edge midpoints for use in semi-Lagrangian advection of momentum. The methods are based on RT0 basis functions and the methods of Perot (2000). The errors for each method are evaluated with a unidirectional parabolic velocity field on equilateral and skewed meshes in a rectangular domain. In summary, we found the following:

1. Methods nP1, nP2, nRT2 and nLS for nodal velocities produce similar results and they are second-order accurate in space on equilateral meshes, while method nRT1 is first-order accurate. Method nP2 is shown to be the most sensitive to mesh quality, while other methods are much less affected.
2. All five methods for the tangential velocity are second-order accurate in space on equilateral meshes. Method tP2 is the most sensitive to mesh quality and method tRT1 is moderately affected by the mesh.
3. For the interpolation at arbitrary locations, the quadratic methods QnRT2tRT1, QnRT2tRT2 and QnRT2tLS are second-order accurate in space and their averaged errors are one to two orders of magnitude smaller than the linear interpolation methods L1nRT1 and L4nRT1. For the quadratic methods, the errors are most sensitive to the particular method employed to reconstruct the tangential velocities.
4. Comparison of nine combinations of different interpolation and trajectory integration schemes shows that interpolation errors dominate over the trajectory integration errors. The quadratic interpolation methods produce the most accurate values of the normal velocity U_j^-, and convergence in time is slightly higher than first-order. Convergence in time for the linear interpolation methods is slightly lower than first-order.

The different interpolation methods are implemented for semi-Lagrangian advection (with the Eulerian–Lagrangian simplification) and evaluated with a backward-facing step test case and field-scale estuarine simulations. The quadratic interpolation schemes consistently produce better results for both large- and small-scale flow structures. Increased accuracy is achieved with an increase in computational overhead of only 15% in the field-scale simulations. The properties of the quadratic interpolation schemes depend to great extent on the method used to approximate the tangential velocities. Use of a method with a compact stencil to calculate the tangential velocities for method QnRT2tRT1

produces less numerical diffusion at the expense of the appearance of oscillations in the velocity field. For method QnRT2tRT2, the tangential velocities are averaged over a larger stencil, and this results in a moderate level of numerical diffusion which suppresses the oscillations which may not be desirable in general.

Acknowledgments

The authors acknowledge Dr. Robert Street for his invaluable and generous input on this work, and Dr. Roy Walters for explanations on the methods in Walters et al. (2007) through personal communication. We also thank members of the COHSTREX project for providing bathymetry and observational data. This research was supported by ONR Grant N00014-05-1-0177 (scientific officers: Dr. Thomas Drake, Dr. Terri Paluszkiewicz and Dr. C. Linwood Vincent) and the Stanford Graduate Fellowship (Wells Family Fellow). Simulations were performed on the JVN and MJM clusters at the ARL Major Shared Resource Center as part of a DOD Challenge Allocation.

Appendix A. Cell-centered and nodal velocities by Perot (2000)

Perot (2000) proposed a method to calculate cell-centered velocities that conserves momentum. It has been widely used in numerical models for solving the momentum advection and Coriolis terms (Fringer et al., 2006; Ham et al., 2007; Walters et al., 2009). $\mathbf{u}_i$ at the center of cell i is calculated with

$$Ac_i\mathbf{u}_i = \sum_{j\in\Psi_i} U_j L_j D_j \mathbf{n}_j, \tag{A.1}$$

where set Ψ_i includes the three edges of cell i and Ac_i is the area of cell i. The schematic of the cell is shown in Fig. A.25. U_j is the normal velocity on edge j, L_j is the length of the edge, D_j is the distance between the cell center and the edge center, and $\mathbf{n}_j$ is the unit vector in the normal direction of edge j, $\mathbf{n}_j = (n_{1j}, n_{2j})$ where $n_{1j} = \cos(\theta_j)$ and $n_{2j} = \sin(\theta_j)$. θ_j is the angle between the normal vector and the positive x axis (Fig. A.25). This method is exact for a constant velocity field on equilateral grids.

Perot (2000) also provides a method to calculate nodal velocities using the normal velocities at all the edges surrounding the node. The vector $\mathbf{u}_p$ at node p can be calculated with

$$Ad_p\mathbf{u}_p = \sum_{j\in\Psi_p} U_j W_j l_j \mathbf{n}_j, \tag{A.2}$$

where set Ψ_p includes edges surrounding node p and Ad_p is the area of the Voronoi diagram formed by the cell centers (shown by the dashed lines in Fig. A.26). W_j is the distance between two cell centers across edge j, and l_j is the distance from node p to the center of edge j (half of the edge length) as illustrated in Fig. A.26.

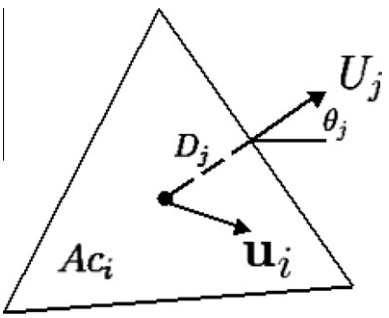

Fig. A.25. Cell-centered velocities by Perot (2000).

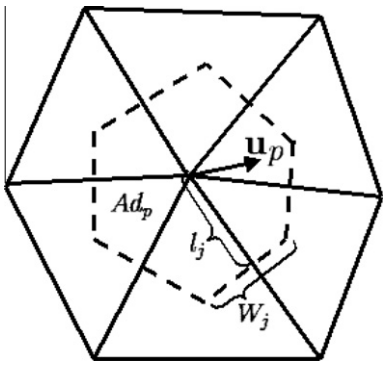

Fig. A.26. Nodal velocities by Perot (2000).

Appendix B. Nodal velocities based on RT0 approximation

Low-order Raviart–Thomas (RT0) vector basis functions assume a constant normal velocity distribution along the edges and linear variation of the velocity field over a cell. Walters et al. (2009) discussed the properties of this method, e.g., regarding the solution of the Coriolis terms, in comparison to the methods of Perot (2000). Following Hanert et al. (2003) and Walters et al. (2009), the velocity at location $\mathbf{x}$ in cell i is

$$\mathbf{u}(\mathbf{x}) = \sum_{j=1}^{N_e} U_j \mathbf{\Phi_j}, \tag{B.1}$$

where N_e is the number of element sides. $\mathbf{\Phi_j}$ are the basis functions given by

$$\mathbf{\Phi_j} = \frac{\mathbf{x} - \mathbf{x_{js}}}{2Ac_i/L_j}, \tag{B.2}$$

where Ac_i is the area of the cell, $\mathbf{x}_{js}$ is the location of the node opposing edge j and L_j is the length of edge j. This is equivalent to a geometric construction as follows. In cell i, vector $\mathbf{u}_p$ at node p can be solved using the normal velocities at edges $j1$ and $j2$ adjacent to node p in cell i (as shown in Fig. B.27), which gives the 2×2 system

$$\mathbf{n}_{j1} \cdot \mathbf{u}_p = U_{j1}, \tag{B.3}$$
$$\mathbf{n}_{j2} \cdot \mathbf{u}_p = U_{j2}. \tag{B.4}$$

This method is exact for constant velocity fields regardless of the mesh skewness while the method by Perot (2000) degrades relatively quickly with decreasing mesh quality. This method also correctly reproduces the linear velocity field $u = a_u + bx$ and $v = a_v + by$, although this velocity field rarely arises in practice.

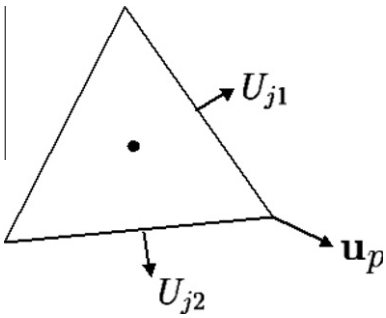

Fig. B.27. Nodal velocities using the RT0 method.

References

Armaly, B.F., Durst, F., Pereira, J.C.F., Schönung, B., 1983. Experimental and theoretical investigation of backward-facing step flow. Journal of Fluid Mechanics 127, 473–496.

Casulli, V., 1990. Semi-implicit finite difference methods for the two-dimensional shallow water equations. Journal of Computational Physics 86, 56–74.

Casulli, V., Walters, R.A., 2000. An unstructured grid, three-dimensional model based on the shallow water equations. International Journal of Numerical Methods in Fluids 32, 311–348.

Fringer, O.B., Gerritsen, M., Street, R.L., 2006. An unstructured-grid, finite-volume, nonhydrostatic parallel coastal ocean simulator. Ocean Modelling 14, 139–173.

Gartling, D., 1990. A test problem for outflow boundary conditions – flow over a backward facing step. International Journal of Numerical Methods in Fluids 11, 953–967.

Giddings, S.N., Fong, D.A., Monismith, S.G., 2011. The role of straining and advection in the intratidal evolution of stratification, vertical mixing, and longitudinal dispersion of a shallow, macrotidal, salt-wedge estuary. Journal of Geophysical Research 116.

Ham, D.A., Kramer, S.C., Stelling, G.S., Pietrzak, J., 2007. The symmetry and stability of unstructured mesh C-grid shallow water models under the influence of Coriolis. Ocean Modelling 16, 47–60.

Ham, D.A., Pietrzak, J.D., Stelling, G.S., 2005. A scalable unstructured grid 3-dimensional finite volume model for the shallow water equations. Ocean Modelling 10, 153–169.

Ham, D.A., Pietrzak, J.D., Stelling, G.S., 2006. Streamline tracking algorithm for semi-Lagrangian advection schemes based on the analytic integration of the velocity field. Journal of Computers and Applied Mathematics 192, 167–174.

Hanert, E., Le Roux, D.Y., Legat, V., Deleersnijder, E., 2005. An efficient Eulerian finite element method for the shallow water equations. Ocean Modelling 10, 115–136.

Hanert, E., Legat, V., Deleersnijder, E., 2003. A comparison of three finite elements to solve the linear shallow water equations. Ocean Modelling 5, 17–35.

Hortal, M., 2002. The development and testing of a new two-time-level semi-Lagrangian scheme (settls) in the ECMWF forecast model. Quarterly Journal of the Royal meteorological Society 128, 1671–1687.

Huebner, K.H., 1975. The Finite Element Method for Engineers. John Wiley and Sons.

Lauritzen, P.H., Kaas, E., Machenhauer, B., 2006. A mass-conservative semi-implicit semi-Lagrangian limited-area shallow-water model on the sphere. Monthly Weather Review 134, 1205–1221.

Le Roux, D.Y., Lin, C.A., Staniforth, A., 2000. A semi-implicit semi-Lagrangian finite-element shallow-water ocean model. Monthly Weather Review 128, 1384–1401.

McCalpin, J.D., 1988. A quantitative analysis of the dissipation inherent in semi-Lagrangian advection. Monthly Weather Review 116, 2330–2336.

Perot, B., 2000. Conservation properties of unstructured staggered mesh schemes. Journal of Computational Physics 159, 58–89.

Plant, W.J., Branch, R., Chatham, G., Chickadel, C.C., Hays, K., Hayworth, B., Horner-Devine, A., Jessup, A., Fong, D.A., Fringer, O.B., Giddings, S.N., Monismith, S.G., Wang, B., 2009. Remotely sensed river surface features compared with modeling and in-situ measurements. Journal of Geophysical Research 114, C11002.

Ritchie, H., 1986. Eliminating the interpolation associated with the semi-Lagrangian scheme. Monthly Weather Review 114, 388–394.

Robert, A., 1981. A stable numerical integration scheme for the primitive meteorological equations. Atmosphere-Ocean 19, 35–46.

Robert, A., 1982. A semi-Lagrangian and semi-implicit numerical integration scheme for the primitive meteorological equations. Journal of Meteorological Society, Japan 60, 319–325.

Rostand, V., Le Roux, D.Y., 2007. Raviart–Thomas and Brezzi–Douglas–Marini finite-element approximations of the shallow-water equations. International Journal for Numerical Methods in Fluids 57, 951–976.

Staniforth, A., Côté, J., 1991. Semi-Lagrangian integration schemes for atmospheric models – A review. Monthly Weather Report 119, 2206–2223.

Vidović, D., 2009. Polynomial reconstruction of staggered unstructured vector fields. Theoretical and Applied Mechanics, Belgrade 36, 85–99.

Vidović, D., Segal, A., Wesseling, P., 2004. A superlinearly convergent finite volume method for the incompressible Navier–Stokes equations on staggered unstructured grids. Journal of Computational Physics 198, 159–177.

Walters, R.A., Casulli, V., 1998. A robust, finite element model for hydrostatic surface water flows. Communications in Numerical Methods in Engineering 14, 931–940.

Walters, R.A., Hanert, E., Pietrzak, J., Le Roux, D.Y., 2009. Comparison of unstructured, staggered grid methods for the shallow water equations. Ocean Modelling 28, 106–117.

Walters, R.A., Lane, E.M., Henry, R.F., 2007. Semi-Lagrangian methods for a finite element coastal ocean model. Ocean Modelling 19, 112–124.

Wang, B., Fringer, O.B., Giddings, S.N., Fong, D.A., 2009. High-resolution simulations of a macrotidal estuary using SUNTANS. Ocean Modelling 26, 60–85.

Xiu, D., Karniadakis, G.E., 2001. A semi-Lagrangian high-order method for Navier–Stokes equations. Journal of Computational Physics 172, 658–684.